NPCA121 Series
Audio Enhancing Engine
Technical Reference Manual

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

www.nuvoton.com

TABLE OF CONTENTS

List of Figure

List of Tables

1 GENERAL DESCRIPTION

The NPCA121 series is a member Nuvoton sound enhancing family based on 32-bit ARM® Cortex®-M4F core with DSP extensions and a Floating Point Unit which run up to 200 MHz with 512 KB of flash memory and 192 KB of SRAM. It supports Bongiovi Digital Power Station (DPS) V3D™ virtual sound enhancing algorithms optimized for consumer products like gaming headset and audio headphone.

The NPCA121 supports plenty of audio peripherals such as I2S, DMIC and audio DPWM mpdulator. It also equipped with a variety of peripherals, such as Multi-Function Timers, Watchdog Timers, RTC, PDMA, UART, SPI, I^2C, PWM, GPIO, 12-bit ADC, USB1.1 Device, Low voltage reset and Brown-out Detector.

The NPCA121 series is suitable for a wide range of applications such as:

- Audio Enhancing Platform
- Consumer Audio Products
- Gaming Headset
- Audio Headphone

2 FEATURES

2.1 NPCA121 Series Features

- Core
  - ARM® Cortex®-M4F core running up to 200 MHz
  - Supports DSP extension with hardware divider
  - Supports IEEE 754 compliant Floating-point Unit (FPU)
  - Supports Memory Protection Unit (MPU)
  - One 24-bit system timer
  - Supports Low Power Sleepmode by WFI and WFE instructions
  - Single-cycle 32-bit hardware multiplier
  - Supports programmable 16 level priorities of Nested Vectored Interrupt Controller (NVIC)
  - Supports programmable mask-able interrupts
  - Supports Embedded Trace Macrocell
- Built-in LDO for wide operating voltage range
- Flash Memory
  - 512KB on-chip Application ROM (APROM)
  - Configurable program code/data allocation
  - 4 KB Flash for loader (LDROM)
  - Supports 2-wire ICP update through SWD/ICE interface
  - Supports In-system program (ISP), In application program (IAP) update
  - Supports 4 KB page erase for all embedded flash
  - Supports 4 KB two-way cache to reduce power consumption and improve performance.
  - Enhanced performance up to 3.4 Core Mark/MHz when running code in Flash with cache
  - Supports 2-wire ICP flash updating through SWD interface
  - Supports 32-bit/64-bit and multi-word flash programming function.
  - Supports fast flash programming verification by CRC function.
- SRAM
  - 192 KB embedded SRAM
  - 32 KB SRAM in bank 0 that supports hardware parity check and retention mode
  - Supports byte-, half-word- and word-access
  - Supports exception (NMI) generated once a parity check error occurs
  - Supports PDMA mode
- Clock Control
  - Built-in 48.0 MHz or 49.152 MHz selectable internal high speed RC oscillator (HIRC) for system operation
  - Built-in 10 kHz internal low speed RC oscillator (LIRC) for Watchdog Timer and wake-up operation
  - 4~24.576 MHz external high speed crystal oscillator (HXT) for precise timing operation
  - 32.768 kHz external low speed crystal oscillator (LXT) for RTC function and low-power system operation
  - Supports one PLL up to 500 MHz for high performance system operation, sourced from HIRC or HXT
  - Supports clock failure detection for high/low speed external crystal oscillator
  - Supports exception (NMI) generation once a clock failure detected
  - Supports clock output
- GPIO
  - Supports four I/O modes:
    - Quasi bi-direction

    - Push-Pull output
    - Open-Drain output
    - Input only with high impendence
  - TTL/Schmitt trigger input selectable
  - I/O pin configured as interrupt source with edge/level trigger setting
  - Supports high slew driver and high sink current I/O (up to 20mA at 3.3V)
  - Supports software selectable slew rate control
  - Supports 5V tolerance function on subset of GPIO except analog I/O
- PDMA (Peripheral DMA)
  - Supports 16 independent configurable channels for automatic data transfer between memories and peripherals
  - Supports stride function.
  - Channel 0, 1 supports time-out function for each channel.
  - Supports Basic and Scatter-Gather Transfer modes
  - Each channel supports circular buffer management using Scatter-Gather Transfer mode
  - Supports two types of priorities modes: Fixed-priority and Round-robin modes
  - Supports byte-, half-word- and word-access
  - Supports single and burst transfer type
  - Supports source and destination address can be increment or fixed.
  - DMA transfer count up to 65536.
- Multi-Function Timer (MFT, Timer + PWM)
  - TIMER mode
    - Supports 4 sets of 32-bit timers with 24-bit up-timer and 8-bit prescale counter, 24-bit up counter value is readable.
    - 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 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, EADC and DMA.
    - Supports Inter-Timer trigger mode
  - PWM mode
    - Supports four 16-bit PWM counters with 10-bit dead time generator
    - Supports 12-bit pre-scale for PWM.
    - Supports independent mode for PWM output channel
    - Supports 8 channel PWM outputs in complementary mode
    - Supports mask function and tri-state enable for each PWM pin
    - Supports interrupt on the following events:
      - PWM counter match zero, period value or compared value
    - Supports trigger EADC on the following events:
      - PWM counter match zero, period value or compared value
- PWM
  - Supports up to 6 independent PWM outputs with 16-bit resolution
  - Supports maximum clock frequency up to 200MHz
  - Supports 12-bit clock prescale
  - Supports dead time with maximum divided 12-bit prescale
  - Supports one-shot or auto-reload counter operation mode
  - Supports up, down or up-down PWM counter type
  - Supports synchronous function for phase control

- Supports counter synchronous start function
- Supports complementary mode for 3 complementary paired PWM output channel
- Supports brake function with auto recovery after brake condition removed
- Supports mask function and tri-state output for each PWM channel
- Supports trigger EADC to start conversion
- Supports up to 6 independent input capture channels with 16-bit resolution counter

- Watchdog Timer
  - 18-bit free running up counter for WDT time-out interval
  - Supports multiple clock sources from LIRC (default selection), HCLK/2048 and LXT
  - 8 selectable time-out period from 1.6ms ~ 26.0sec (depending on clock source)
  - Able to wake up from Power-down or Idle mode
  - Interrupt or reset selectable on watchdog time-out
  - Supports selectable WDT reset delay period, including 1026、130、18 or 3 WDT_CLK reset delay period
  - Configurable to force WDT enable after chip power-on or reset.
  - Supports WDT time-out wake-up function only if WDT clock source is selected as LIRC or LXT
- Window Watchdog Timer
  - Supports multiple clock sources from HCLK/2048 (default selection) and LIRC
  - Window set by 6-bit counter with 11-bit prescale
  - WWDT counter suspends in Idle/Power-down mode
- RTC
  - Supports software compensation by setting frequency compensate register (FCR),compensated clock accuracy reaches ±5ppm within 5 seconds
  - Supports RTC counter (second, minute, hour) and calendar counter (day, month, year)
  - Supports Alarm registers (second, minute, hour, day, month, year)
  - Selectable 12-hour or 24-hour mode
  - Automatic leap year recognition
  - Supports Day of the Week counter
  - Supports periodic time tick interrupt with 8 period options 1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2 and 1 second
  - Supports 1 Hz, clock output
  - Supports wake-up from idle mode, Power-down mode and Standby Power-down mode
  - Supports 32 kHz Oscillator gain control
  - Supports RTC Time Tick and Alarm Match interrupt
  - Support Time stamp
- UART
  - Supports low power UART (LPUART): baud rate clock from LXT(32.768 kHz) with 9600bps in Power-down mode even system clock is stopped
  - Support baud rate up to 12.5 MHz
  - Supports 16-byte FIFOs with programmable level trigger
  - Supports auto flow control (CTS and RTS)
  - Supports RS-485 9-bit mode and direction control
  - Programmable baud-rate generator up to 1/16 system clock
  - Programmable receiver FIFO trigger level
  - Supports wake-up function
  - Supports 8-bit receiver FIFO time-out detection function
  - Supports Auto-Baud Rate measurement and baud rate compensation function
  - Supports break error, frame error, parity error and receive/transmit FIFO overflow detection function
  - Supports nCTS, incoming data, RX FIFO reached threshold and RS-485 Address Match (AAD mode) wake-up function in idle mode.
  - Supports hardware or software enables to program nRTS pin to control RS-485

transmission direction
- Supports PDMA mode

- I^2C
  - Supports up to two sets of I^2C devices
  - Supports Master/Slave mode
  - Bidirectional data transfer between masters and slaves
  - Multi-master bus (no central master)
  - Supports 10 bits mode
  - Support High speed mode 3.4Mbps
  - Supports Standard mode (100 kbps), Fast mode (400 kbps) and Fast mode plus (1 Mbps)
  - Arbitration between simultaneously transmitting masters without corruption of serial data on the bus
  - Serial clock synchronization allows devices with different bit rates to communicate via one serial bus
  - Serial clock synchronization can be used as a handshake mechanism to suspend and resume serial transfer
  - Programmable clocks allow versatile rate control
  - Supports multiple address recognition (four slave address with mask option)
  - Supports SMBus and PMBus
  - Supports multi-address Power-down wake-up function
- I^2S
  - Supports one I^2S interface
  - Interface with external audio CODEC
  - Supports Master and Slave mode
  - Capable of handling 8-, 16-, 24- and 32-bit word sizes
  - Mono and stereo audio data
  - I^2S protocols: Philips standard, MSB-justified, and LSB-justified data format
  - PCM protocols: PCM standard, MSB-justified, and LSB-justified data format
  - PCM protocol supports TDM multi-channel transmission in one audio sample, the number of data channels can be set as 2, 4, 6, or 8
  - Two 16-level FIFO data buffers, one for transmitting and the other for receiving
  - Generates interrupt requests when buffer levels cross a programmable boundary
  - Supports two DMA requests, one for transmitting and the other for receiving
- SPI0
  - SPI Quad controller – SPI0
  - Supports Master or Slave mode operation
  - Supports 2-bit Transfer mode
  - Supports Dual and Quad I/O Transfer mode
  - Supports one/two data channel half-duplex transfer
  - Support receive-only mode
  - Configurable bit length of a transfer word from 8 to 32-bit
  - Provides separate 8-level depth transmit and receive FIFO buffers
  - Supports MSB first or LSB first transfer sequence
  - Supports the byte reorder function
  - Supports Byte or Word Suspend mode
  - Supports 3-wired, no slave select signal, bi-direction interface
  - Master up to 25 MHz, and Slave up to 25 MHz (when chip operating at V_{DD} = 2.7~3.6V)
  - Supports PDMA mode
- SPI / I^2S
  - Supports two sets of SPI/ I2S controllers – SPI1/ SPI2
  - Supports Master or Slave mode operation

- Supports two PDMA requests, one for transmitting and the other for receiving
- SPI supports configurable bit length of a transfer word from 8 to 32-bit
- SPI Provides separate 4-level of 32-bit (or 8-level of 16-bit) transmit and receive FIFO buffers which depended on SPI setting of data width
- SPI supports MSB first or LSB first transfer sequence
- SPI supports the byte reorder function
- SPI supports Byte or Word Suspend mode
- SPI supports one data channel half-duplex transfer
- SPI supports receive-only mode
- I2S interface with external audio CODEC
- I2S supports Master and Slave mode
- I2S supports 8-, 16-, 24- and 32-bit audio data sizes
- I2S supports mono and stereo audio data
- I2S supports PCM mode A, PCM mode B, I2S and MSB justified data format
- I2S Interface with external audio CODEC
- I2S provides two 4-level FIFO data buffers, one for transmitting and the other for receiving
- Generates interrupt requests when buffer levels cross a programmable boundary

- EADC
  - Analog input voltage range: 0~ AV_{DD}
  - Supports single 12-bit SAR EADC conversion
  - 12-bit resolution and 10-bit accuracy is guaranteed
  - Up to 13 external single-ended analog input channels
  - Up to 2 MSPS conversion rate
  - Supports three power saving modes:
    - Deep Power-down mode.
    - Power-down mode.
    - Standby mode.
  - Supports single EADC interrupt
  - Supports calibration and load calibration words capability.
  - An A/D conversion can be triggered by Software enable, External pin, Timer 0~3 overflow pulse trigger and PWM trigger.
  - 12-bit, 10-bit, 8-bit, 6-bit configurable resolution.
  - Maximum EADC clock frequency is 60 MHz.
  - Configurable EADC internal sampling time.
  - Up to 13 sample modules
    - Each of sample module 0~12 which is configurable for EADC converter channel EADC_CH0~12 and trigger source.
    - Double buffer for sample module 0~3
    - Configurable sampling time for each sample module.
    - Conversion results are held in 13 data registers with valid and overrun indicators.
  - Supports PDMA transfer
- USB 1.1 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 12 endpoints for configurable Control/Bulk/Interrupt/Isochronous transfer types and maximum 1k bytes buffer size
  - Provides remote wake-up capability Programmable initial value
- Digital Microphone Inputs
  - Provides one 32-level FIFO data buffers for receiving.

- Generates interrupt requests when buffer levels cross a programmable boundary.
- Supports PDMA transfer.
- Supports up to four channel digital microphones.
- Both digital PDM microphone inputs can be used simultaneously.

- Voice Active Detection
  - Configuration detect levels.
  - Supports idle mode wake-up function.
  - Supports auto switch DMIC path when CPU wake-up by VAD.
  - Generates interrupt requests when voice detected.
- Audio DPWM Modulator
  - Differential Audio PWM Output (DPWM)
  - Supports left channel, right channels and sub-woofer channel.
  - Supports sample rate from 16~96 kHz
  - Programmable biquad filter with 10 band.
  - PDMA data channel for streaming of PCM audio data.
  - Supports the single precision floating point for input data and BIQ coefficient.
  - Provides one 32-level FIFO data buffers for transmitting.
- Cyclic Redundancy Calculation Unit
  - Supports four common polynomials CRC-CCITT, CRC-8, CRC-16, and CRC-32
  - Programmable initial value
  - Supports programmable order reverse setting for input data and CRC checksum
  - Supports programmable 1's complement setting for input data and CRC checksum.
  - Supports 8-/16-/32-bit of data width
  - Programmable seed value
  - 8-bit write mode: 1-AHB clock cycle operation
  - 16-bit write mode: 2-AHB clock cycle operation
  - 32-bit write mode: 4-AHB clock cycle operation
  - Supports using DMA to write data to perform CRC operation
- Brown-out Detector
  - With 8 levels: 3.0V/2.8V/2.6V/2.4V/2.2V/2.0V/1.8V/1.6V
  - Supports Brown-out Interrupt and Reset option
- Low Voltage Reset
  - Threshold voltage levels: 1.5V
- Operating Temperature: -40℃~85℃
- Packages
  - All Green package (RoHS)
  - QFN 48-pin (6x6 mm) – in developing
  - LQFP 64-pin (7x7 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
CAN	Controller Area Network
DAP	Debug Access Port
DES	Data Encryption Standard
DMIC	Digital Microphone Inputs
DPWM	Audio DPWM Modulator
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	High Speed RC Oscillator
HXT	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
VAD	Voice Active Detection
WDT	Watchdog Timer
WWDT	Window Watchdog Timer

Table 3.1-1 List of Abbreviations

4 PARTS INFORMATION LIST AND PIN CONFIGURATION

4.1 Parts Information

PART NUMBER		NPCA1	
		21DD	21DY
Max. CPU frequency (MHz)		200	
Flash (KB)		512	
SRAM (KB)		192	
ISP Loader ROM (KB)		4	
I/O		57	41
32-bit Timer		4	
RTC		√	
Connectivity	UART	1	
	SPI	1	
	SPI/I^{2}S	2	
	I^{2}S	1	
	I^{2}C	2	
PWM		6	5
USB 1.1 FS Device		√	
12-bit ADC		13	12
Audio Function	Audio DPWM	2.1	-
	VAD	√	-
	DMIC	4	-
Package		LQFP 64 (7x7 mm)	QFN 48 (6x6 mm)
Status		Released	In Developing

Table 4.1-1 Devices Features and Peripheral Counts

4.2 Ordering Information

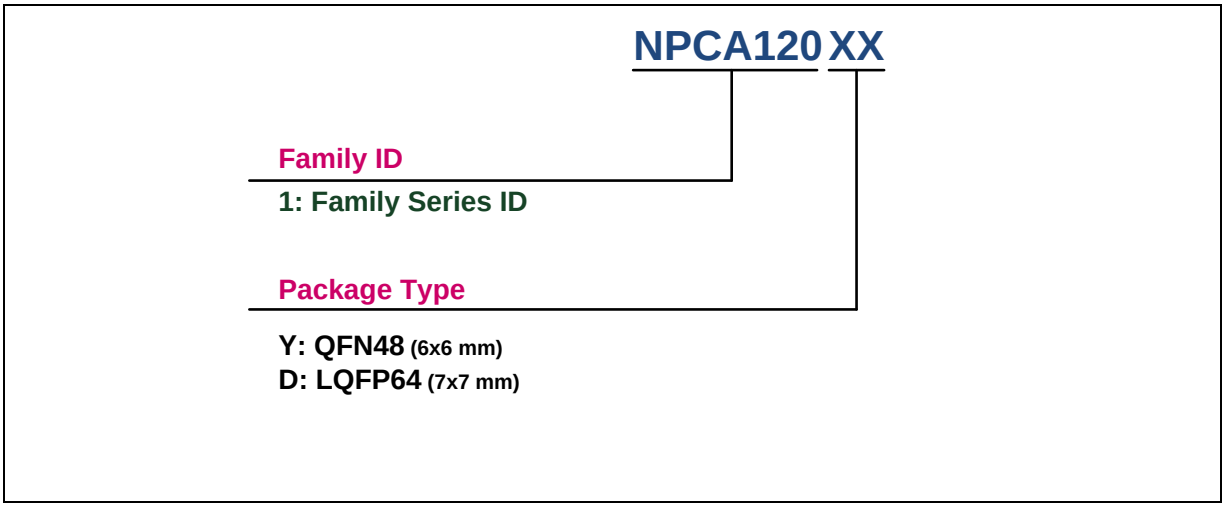


Figure 4.2-1 Ordering Information Scheme

4.3 Pin Configuration

4.3.1 QFN48 (6x6 mm) Pin Diagram

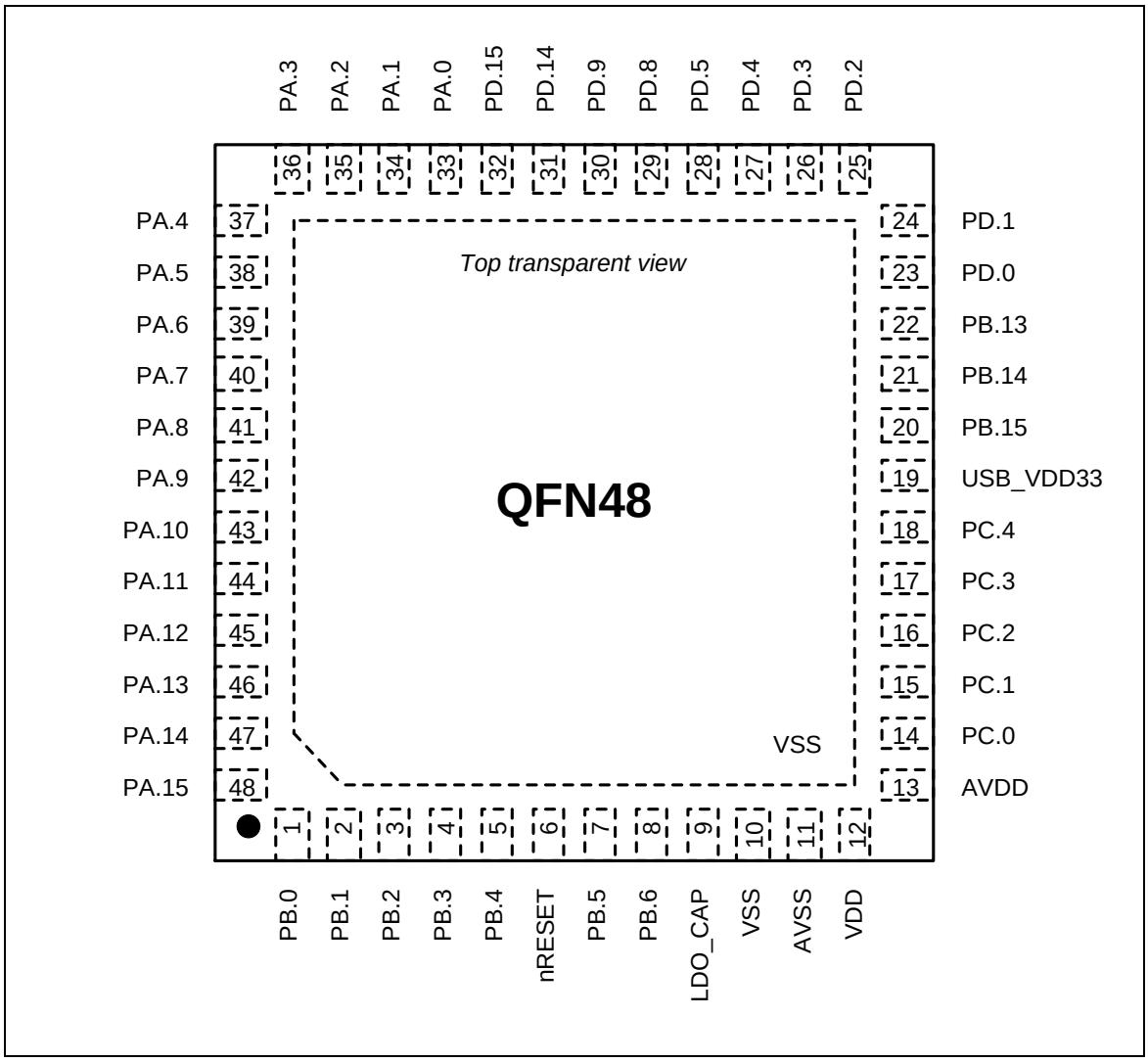


Figure 4.3-1 QFN48 (6x6 mm) Pin Diagram (In Developing)

4.3.2 LQFP64 (7x7 mm) Pin Diagram

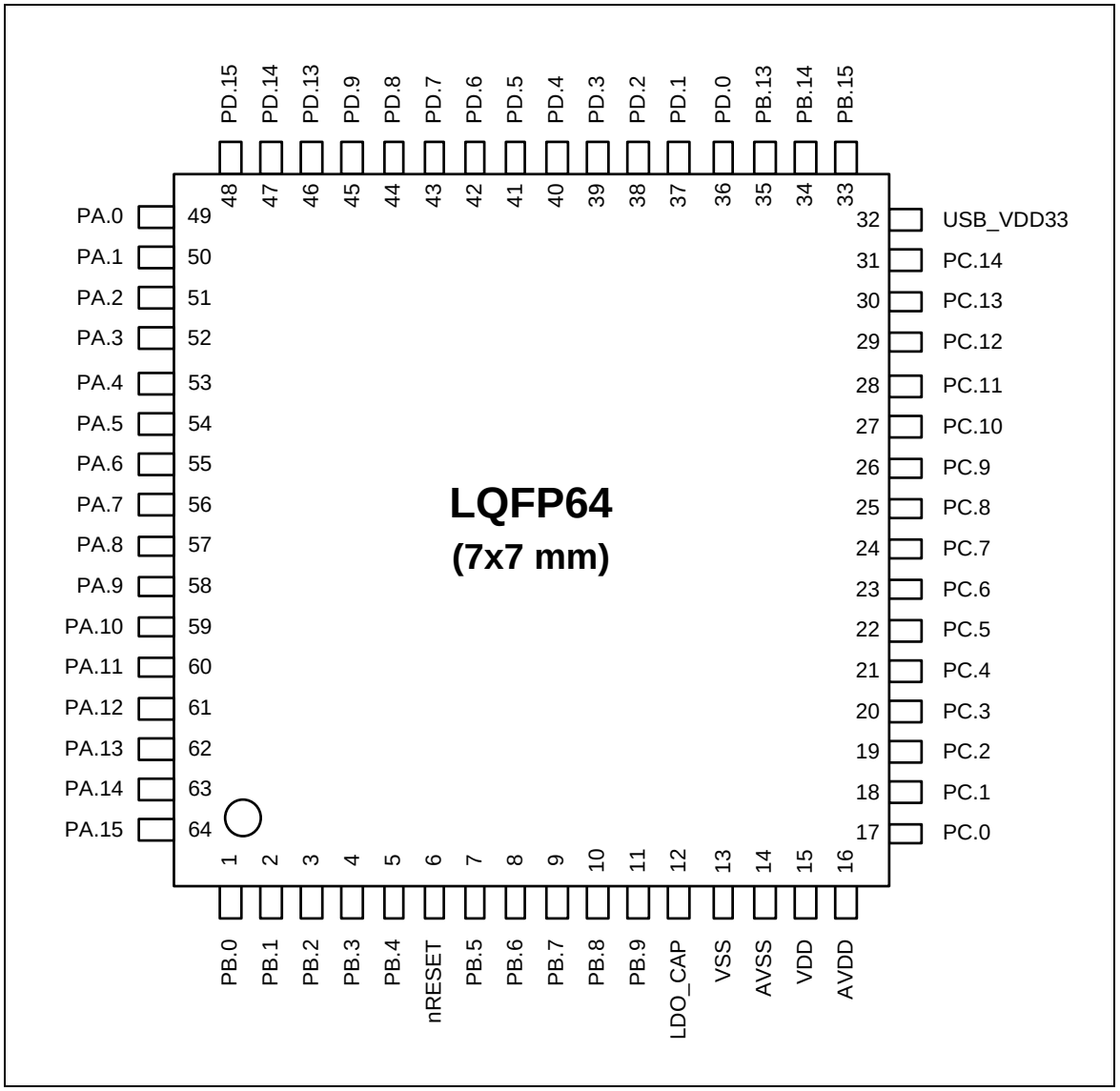


Figure 4.3-2 LQFP64 (7x7 mm) Pin Diagram

4.4 Pin Description

MFP = Multi-function pin.

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

Pins		Pin Name	Type	MFP	Description
QFN48 (6x6)	LQFP64 (7x7)				
1	1	PB.0	I/O	MFP0	General purpose digital I/O pin.
		PWM0_SYNC_IN	I/O	MFP1	PWM0 counter synchronous trigger input pin.
		I2C0_SCL	I/O	MFP2	I2C0 clock pin.
		PWM0_CH0	I/O	MFP3	PWM0 channel0 output/capture input.
2	2	PB.1	I/O	MFP0	General purpose digital I/O pin.
		PWM0_SYNC_OUT	I/O	MFP1	PWM0 counter synchronous trigger output pin.
		I2C0_SDA	I/O	MFP2	I2C0 data input/output pin.
		PWM0_CH1	I/O	MFP3	PWM0 channel1 output/capture input.
3	3	PB.2	I/O	MFP0	General purpose digital I/O pin.
		PWM0_CH0	I/O	MFP1	PWM0 channel0 output/capture input.
		TM2	I/O	MFP2	Timer2 event counter input / toggle output.
		PWM0_CH2	I/O	MFP3	PWM0 channel2 output/capture input.
4	4	PB.3	I/O	MFP0	General purpose digital I/O pin.
		PWM0_CH1	I/O	MFP1	PWM0 channel1 output/capture input.
		TM2_EXT	I/O	MFP2	Timer2 external capture input.
		DMIC_DAT1	I	MFP3	Digital microphone channel 1 data input pin.
		UART0_RXD	I	MFP4	UART0 Data receiver input pin.
		PWM0_CH3	I/O	MFP5	PWM0 channel3 output/capture input.
5	5	PB.4	I/O	MFP0	General purpose digital I/O pin.
		UART0_nCTS	I	MFP1	Clear to Send input pin for UART0.
		PWM0_CH0	I/O	MFP2	PWM0 channel0 output/capture input.
		DMIC_CLK1	O	MFP3	Digital microphone channel 1 clock output pin.
		UART0_TXD	O	MFP4	UART0 data transmitter output pin.
		PWM0_CH4	I/O	MFP5	PWM0 channel4 output/capture input.
6	6	RESETN	I	MFP0	External reset input: active LOW, with an internal pull-up. Set this pin low reset to initial state.
7	7	PB.5	I/O	MFP0	General purpose digital I/O pin.
		XT1_OUT	I	MFP1	External 4~24.576 MHz (high speed) crystal output pin.
		PWM0_CH1	I/O	MFP2	PWM0 channel1 output/capture input.
		I2C0_SDA	I/O	MFP3	I2C0 data input/output pin.
		I2C1_SDA	I/O	MFP4	I2C1 data input/output pin.
		DMIC_DAT0	I	MFP5	Digital microphone channel 0 data input pin.

Pins		Pin Name	Type	MFP	Description
QFN48 (6x6)	LQFP64 (7x7)				
8	8	PB.6	I/O	MFP0	General purpose digital I/O pin.
		XT1_IN	I	MFP1	External 4~24.576 MHz (high speed) crystal input pin.
		PWM0_CH2	I/O	MFP2	PWM0 channel2 output/capture input.
		I2C0_SCL	I/O	MFP4	I2C0 serial clock pin.
		I2C1_SCL	I/O	MFP5	I2C1 serial clock pin.
		DMIC_CLK0	O	MFP6	Digital microphone channel 0 clock output pin.
	9	PB.7	I/O	MFP0	General purpose digital I/O pin.
		UART0_nRTS	O	MFP1	Request to Send output pin for UART0.
		PWM0_CH3	I/O	MFP2	PWM0 channel3 output/capture input.
	10	PB.8	I/O	MFP0	General purpose digital I/O pin.
		UART0_TXD	O	MFP1	UART0 Data transmitter output pin.
		PWM0_CH4	I/O	MFP2	PWM0 channel4 output/capture input.
	11	PB.9	I/O	MFP0	General purpose digital I/O pin.
		UART0_RXD	I	MFP1	UART0 Data receiver input pin.
		PWM0_CH5	I/O	MFP2	PWM0 channel5 output/capture input.
9	12	LDO_CAP	P	MFP0	LDO output pin. Note: This pin needs to be connected with a 1uF capacitor.
10	13	VSS	P	MFP0	Ground pin for digital circuit.
11	14	AVSS	P	MFP0	Ground pin for analog circuit.
12	15	VDD	P	MFP0	Power supply for I/O ports and LDO source for internal PLL and digital circuit.
13	16	AVDD	P	MFP0	Power supply for internal analog circuit.
14	17	PC.0	I/O	MFP0	General purpose digital I/O pin.
		I2C1_SCL	I/O	MPF1	I2C1 clock pin.
		X32_OUT	O	MFP2	External 32.768 kHz (low-speed) crystal output pin.
		SPI1_MOSI	I/O	MFP3	SPI1 MOSI (Master Out, Slave In) pin; or I2S1 data output pin.
15	18	PC.1	I/O	MFP0	General purpose digital I/O pin.
		I2C1_SDA	I/O	MFP1	I2C1 data input/output pin.
		X32_IN	I	MFP2	External 32.768 kHz (low-speed) crystal input pin.
		SPI1_MISO	I/O	MFP3	SPI1 MISO (Master In, Slave Out) pin; or I2S1 data input pin.
16	19	PC.2	I/O	MFP0	General purpose digital I/O pin.
		I2C1_SMBSUS	O	MFP1	I2C1 SMBus SMBSUS# pin (PMBus CONTROL pin)
		TM3	I/O	MFP2	Timer3 event counter input / toggle output.
		SPI1_CLK	I/O	MFP3	SPI1 Serial Clock pin; or I2S1 bit clock pin.

Pins		Pin Name	Type	MFP	Description
QFN48 (6x6)	LQFP64 (7x7)				
17	20	PC.3	I/O	MFP0	General purpose digital I/O pin.
		I2C1_SMBAL	O	MFP1	I2C1 SMBus SMBALERT# pin
		TM3_EXT	I/O	MFP2	Timer3 external capture input.
		SPI1_SS	I/O	MFP3	SPI1 slave select pin; or I2S1 left right channel clock pin.
18	21	PC.4	I/O	MFP0	General purpose digital I/O pin.
		PWM0_CH2	I/O	MFP1	PWM0 channel2 output/capture input.
		CLKO	O	MFP2	Clock Output pin.
		SPI1_I2SMCLK	O	MFP3	SPI1 I2S master clock output pin.
	22	PC.5	I/O	MFP0	General purpose digital I/O pin.
		INT1	I	MFP1	External interrupt1 input pin.
		SPI2_MOSI	I/O	MFP2	SPI2 MOSI (Master Out, Slave In) pin.
	23	PC.6	I/O	MFP0	General purpose digital I/O pin.
		INT2	I	MFP1	External interrupt2 input pin.
		SPI2_MISO	I/O	MFP2	SPI2 MISO (Master In, Slave Out) pin.
	24	PC.7	I/O	MFP0	General purpose digital I/O pin.
		SPI0_SS0	I/O	MFP1	1st SPI0 Slave Select pin
		SPI2_CLK	I/O	MFP2	SPI2 serial clock pin.
	25	PC.8	I/O	MFP0	General purpose digital I/O pin.
		SPI0_MOSI1	I/O	MFP1	2nd SPI0 MOSI (Master Out, Slave In) pin.
		SPI2_SS	I/O	MFP2	SPI2 Slave Select pin.
	26	PC.9	I/O	MFP0	General purpose digital I/O pin.
		SPI0_MISO1	I/O	MFP1	2nd SPI0 MISO (Master In, Slave Out) pin.
		SPI2_I2SMCLK	O	MFP2	SPI2 I2S master clock output pin
	27	PC.10	I/O	MFP0	General purpose digital I/O pin.
		SPI0_MOSI0	I/O	MFP1	1st SPI0 MOSI (Master Out, Slave In) pin.
		PWM0_BRAKE0	I	MFP2	Brake input pin 0 of PWM0.
		DPWM_RN	O	MFP3	Audio DPWM right channel negative output pin.
	28	PC.11	I/O	MFP0	General purpose digital I/O pin.
		SPI0_MISO0	I/O	MFP1	1st SPI0 MISO (Master In, Slave Out) pin.
		PWM0_BRAKE1	I	MFP2	Brake input pin 1 of PWM0.
		DPWM_RP	O	MFP3	Audio DPWM right channel positive output pin.
	29	PC.12	I/O	MFP0	General purpose digital I/O pin.
		SPI0_CLK	I/O	MFP1	SPI0 serial clock pin.
		DPWM_LN	O	MFP3	Audio DPWM left channel negative output pin.

Pins		Pin Name	Type	MFP	Description
QFN48 (6x6)	LQFP64 (7x7)				
	30	PC.13	I/O	MFP0	General purpose digital I/O pin.
		PWM0_CH3	I/O	MFP1	PWM0 channel3 output/capture input.
		I2C0_SCL	I/O	MFP2	I2C0 clock pin.
		DPWM_LP	O	MFP3	Audio DPWM left channel positive output pin.
	31	PC.14	I/O	MFP0	General purpose digital I/O pin.
		PWM0_CH4	I/O	MFP1	PWM0 channel4 output/capture input.
		I2C0_SDA	I/O	MFP2	I2C0 data input/output pin.
		DPWM_SN	O	MFP3	Audio DPWM sub-woofer channel negative output pin.
19	32	USB_VDD33	P	MFP0	Power supply for USB, DC 3.3V.
20	33	PB.15	I/O	MFP0	General purpose digital I/O pin.
		USB_VBUS	P	MFP1	Power supply from USB or HUB.
		I2S0_MCLK	O	MFP2	I2S0 master clock output pin.
21	34	PB.14	I/O	MFP0	General purpose digital I/O pin.
		USB_D-	A	MFP1	USB differential signal D-.
		I2S0_DO	O	MFP2	I2S0 data output pin.
22	35	PB.13	I/O	MFP0	General purpose digital I/O pin.
		USB_D+	A	MFP1	USB differential signal D+.
		I2S0_DI	I	MFP2	I2S0 data input pin.
23	36	PD.0	I/O	MFP0	General purpose digital I/O pin.
		INT3	I	MFP1	External interrupt3 input pin.
		I2C1_SCL	I/O	MFP2	I2C1 clock pin.
		I2C0_SCL	I/O	MFP3	I2C0 clock pin.
		I2S0_BCLK	I/O	MFP4	I2S0 bit clock pin.
		DPWM_LN	O	MFP5	Audio DPWM left channel negative output pin.
24	37	PD.1	I/O	MFP0	General purpose digital I/O pin.
		INT4	I	MFP1	External interrupt4 input pin.
		I2C1_SDA	I/O	MFP2	I2C1 data p input/output in.
		I2C0_SDA	I/O	MFP3	I2C0 data input/output pin.
		I2S0_LRCK	I/O	MFP4	I2S0 left right channel clock pin.
		DPWM_LP	O	MFP5	Audio DPWM left channel positive output pin.

Pins		Pin Name	Type	MFP	Description
QFN48 (6x6)	LQFP64 (7x7)				
25	38	PD.2	I/O	MFP0	General purpose digital I/O pin.
		TRACE_CLK	O	MFP1	TPIU for ETM Tx trace clock output pin.
		SPI1_MOSI	I/O	MFP2	SPI1 MOSI (Master Out, Slave In) pin.
		I2S0_MCLK	O	MFP3	I2S0 master clock output pin.
		I2C1_SCL	I/O	MFP4	I2C1 clock pin.
		TM0	I/O	MFP5	Timer0 event counter input / toggle output.
26	39	PD.3	I/O	MFP0	General purpose digital I/O pin.
		TRACE_DATA0	O	MFP1	TPIU for ETM Tx trace data output bit0.
		SPI1_MISO	I/O	MFP2	SPI1 MISO (Master In, Slave Out) pin.
		I2S0_LRCK	I/O	MFP3	I2S0 left right channel clock pin.
		DMIC_CLK1	O	MFP4	Digital microphone channel 1 clock output pin.
		TM2	I/O	MFP5	Timer2 event counter input / toggle output.
27	40	PD.4	I/O	MFP0	General purpose digital I/O pin.
		TRACE_DATA1	O	MFP1	TPIU for ETM Tx trace data output bit1.
		SPI1_CLK	I/O	MFP2	SPI1 serial clock pin.
		I2S0_DI	I	MFP3	I2S0 data input pin.
		DMIC_DAT1	I	MFP4	Digital microphone channel 1 data input pin.
		TM1	I/O	MFP5	Timer1 event counter input / toggle output.
28	41	PD.5	I/O	MFP0	General purpose digital I/O pin.
		TRACE_DATA2	O	MFP1	TPIU for ETM Tx trace data output bit2.
		SPI1_SS	I/O	MFP2	SPI1 Slave Select pin.
		I2S0_DO	O	MFP3	I2S0 data output pin.
		DMIC_CLK0	O	MFP4	Digital microphone channel 0 clock output pin.
		DPWM_RN	O	MFP5	Audio DPWM right channel negative output pin.
	42	PD.6	I/O	MFP0	General purpose digital I/O pin.
		TRACE_DATA3	O	MFP1	TPIU for ETM Tx trace data output bit3.
		SPI1_I2SMCLK	O	MFP2	SPI1 I2S master clock output pin
		I2S0_BCLK	I/O	MFP3	I2S0 Bit Clock pin.
		DMIC_DAT0	I	MFP4	Digital microphone channel 0 data input pin.
		DPWM_RP	O	MFP5	Audio DPWM right channel positive output pin.
	43	PD.7	I/O	MFP0	General purpose digital I/O pin.
		PWM0_CH5	I/O	MFP1	PWM0 channel5 output/capture input.
		INT1	I	MFP2	External interrupt1 input pin.

Pins		Pin Name	Type	MFP	Description
QFN48 (6x6)	LQFP64 (7x7)				
29	44	PD.8	I/O	MFP0	General purpose digital I/O pin.
		ICE_CLK	I	MFP1	Serial wired debugger clock pin
		TM0	I/O	MFP2	Timer0 event counter input / toggle output.
		I2C1_SCL	I/O	MFP3	I2C1 clock pin.
		I2C0_SCL	I/O	MFP4	I2C0 clock pin.
		DPWM_SN	O	MFP5	Audio DPWM sub-woofer channel negative output pin.
30	45	PD.9	I/O	MFP0	General purpose digital I/O pin.
		ICE_DAT	I/O	MFP1	Serial wired debugger data pin
		TM0_EXT	I/O	MFP2	Timer0 external capture input.
		I2C1_SDA	I/O	MFP3	I2C1 data input/output pin.
		I2C0_SDA	I/O	MFP4	I2C0 data input/output pin.
		DPWM_SP	O	MFP5	Audio DPWM sub-woofer channel positive output pin.
	46	PD.13	I/O	MFP0	General purpose digital I/O pin.
		SPI0_SS1	O	MFP1	2nd SPI0 Slave Select pin
		EADC0_CH10	A	MFP2	EADC0 channel10 analog input.
31	47	PD.14	I/O	MFP0	General purpose digital I/O pin.
		UART0_nCTS	I	MFP1	Clear to Send input pin for UART0.
		EADC0_CH11	A	MFP2	EADC0 channel11 analog input.
		I2C0_SCL	I/O	MFP3	I2C0 clock pin.
		UART0_TXD	O	MFP4	UART0 data transmitter output pin.
		I2C1_SCL	I/O	MFP5	I2C1 clock pin.
32	48	PD.15	I/O	MFP0	General purpose digital I/O pin.
		UART0_nRTS	O	MFP1	Request to Send output pin for UART0.
		EADC0_CH12	A	MFP2	EADC0 channel12 analog input.
		I2C0_SDA	I/O	MFP3	I2C0 data input/output pin.
		UART0_RXD	I	MFP4	UART0 data receiver input pin.
		I2C1_SDA	I/O	MFP5	I2C1 data input/output pin.
33	49	PA.0	I/O	MFP0	General purpose digital I/O pin.
		SPI0_SS1	O	MFP1	2nd SPI0 Slave Select pin
		EADC0_CH0	A	MFP2	EADC0 channel0 analog input.
		DMIC_DAT0	I	MFP3	Digital microphone channel 0 data input pin.
34	50	PA.1	I/O	MFP0	General purpose digital I/O pin.
		SPI0_MOSI1	I/O	MFP1	2nd SPI0 MOSI (Master Out, Slave In) pin.
		EADC0_CH1	A	MFP2	EADC0 channel1 analog input.

Pins		Pin Name	Type	MFP	Description
QFN48 (6x6)	LQFP64 (7x7)				
		DMIC_CLK0	O	MFP3	Digital microphone channel 0 clock output pin.
35	51	PA2	I/O	MFP0	General purpose digital I/O pin.
		SPI0_MISO1	I/O	MFP1	2nd SPI0 MISO (Master In, Slave Out) pin.
		EADC0_CH2	A	MFP2	EADC0 channel2 analog input.
		DMIC_DAT1	I	MFP3	Digital microphone channel 1 data input pin.
36	52	PA.3	I/O	MFP0	General purpose digital I/O pin.
		SPI0_MOSI0	I/O	MFP1	1st SPI0 MOSI (Master Out, Slave In) pin.
		EADC0_CH3	A	MFP2	EADC0 channel3 analog input.
		DMIC_CLK1	O	MFP3	Digital microphone channel 1 clock output pin.
37	53	PA.4	I/O	MFP0	General purpose digital I/O pin.
		SPI0_MISO0	I/O	MFP1	1st SPI0 MISO (Master In, Slave Out) pin.
		EADC0_CH4	A	MFP2	EADC0 channel4 analog input.
		DPWM_LN	O	MFP3	Audio DPWM left channel negative output pin.
38	54	PA.5	I/O	MFP0	General purpose digital I/O pin.
		SPI0_CLK	I/O	MFP1	SPI0 serial clock pin.
		EADC0_CH5	A	MFP2	EADC0 channel5 analog input.
		DPWM_LP	O	MFP3	Audio DPWM left channel positive output pin.
39	55	PA.6	I/O	MFP0	General purpose digital I/O pin.
		SPI0_SS0	I/O	MFP1	1st SPI0 Slave Select pin
		EADC0_CH6	A	MFP2	EADC0 channel6 analog input.
40	56	PA.7	I/O	MFP0	General purpose digital I/O pin.
		UART0_TXD	O	MFP1	UART0 data transmitter output pin.
		EADC0_CH7	A	MFP2	EADC0 channel7 analog input.
		SPI2_MISO	I/O	MFP4	SPI2 MISO (Master In, Slave Out) pin; or I2S2 data input pin.
41	57	PA.8	I/O	MFP0	General purpose digital I/O pin.
		UART0_RXD	I	MFP1	UART0 data receiver input pin..
		EADC0_CH8	A	MFP2	EADC0 channel8 analog input.
		SPI2_MOSI	I/O	MFP4	SPI2 MOSI (Master Out, Slave In) pin; or I2S2 data output pin.
42	58	PA.9	I/O	MFP0	General purpose digital I/O pin.
		I2C0_SCL	I/O	MFP1	I2C0 Serial Clock pin
		EADC0_CH9	A	MFP2	EADC0 channel9 analog input.
		SPI2_SS	I/O	MFP4	SPI2 slave select pin; or I2S2 left right channel clock pin.
43	59	PA.10	I/O	MFP0	General purpose digital I/O pin.
		I2C0_SDA	I/O	MFP1	I2C0 data input/output pin.

Pins		Pin Name	Type	MFP	Description
QFN48 (6x6)	LQFP64 (7x7)				
		EADC0_ST	I	MFP2	EADC0 external trigger input.
		DPWM_RN	O	MFP3	Audio DPWM right channel negative output pin.
		SPI2_CLK	I/O	MFP4	SPI2 clock pin; or I2S2 bit clock pin.
44	60	PA.11	I/O	MFP0	General purpose digital I/O pin.
		I2C0_SMBSUS	O	MFP1	I2C0 SMBus SMBSUS# pin (PMBus CONTROL pin)
		TM0	I/O	MFP2	Timer0 event counter input / toggle output.
		DPWM_RP	O	MFP3	Audio DPWM right channel positive output pin.
45	61	PA.12	I/O	MFP0	General purpose digital I/O pin.
		I2C0_SMBAL	O	MFP1	I2C0 SMBus SMBALERT# pin
		TM0_EXT	I/O	MFP2	Timer0 external capture input.
		SPI2_I2SMCLK	O	MFP4	SPI2 I2S master clock output pin.
46	62	PA.13	I/O	MFP0	General purpose digital I/O pin.
		CLKO	O	MFP1	Clock Output pin.
		INT0	I	MFP2	External interrupt0 input pin.
		DPWM_SN	O	MFP3	Audio DPWM sub-woofer channel negative output pin.
		I2C1_SCL	I/O	MFP4	I2C1 clock pin.
47	63	PA.14	I/O	MFP0	General purpose digital I/O pin.
		SPI0_SS0	I/O	MFP1	1st SPI0 Slave Select pin
		TM1	I/O	MFP2	Timer1 event counter input / toggle output.
		DPWM_SP	O	MFP3	Audio DPWM sub-woofer channel positive output pin.
		I2C1_SDA	I/O	MFP4	I2C1 data input/output pin.
48	64	PA.15	I/O	MFP0	General purpose digital I/O pin.
		INT0	I	MFP1	External interrupt0 input pin.
		TM1_EXT	I/O	MFP2	Timer1 external capture input.

Note:

1. Part number NPCA121DY do not provide DPWM and DMIC functionality.

Table 4.4-1 Pin Description

4.5 GPIO Alternate Function Summary

MFP* = Multi-function pin. (Reference section)

Pin function is defined in SYS_GPx_MFPx registers. For example PA0~7 pin functions are defined in SYS_GPA_MFPL register, and PA8~15 pin functions are defined in SYS_GPA_MFPH register.

MFP0	MFP1	MFP2	MFP3	MFP4	MFP5
PA.0	SPI0_SS1	EADC0_CH0	DMIC_DAT0		
PA.1	SPI0_MOSI1	EADC0_CH1	DMIC_CLK0		
PA.2	SPI0_MISO1	EADC0_CH2	DMIC_DAT1		
PA.3	SPI0_MOSI0	EADC0_CH3	DMIC_CLK1		
PA.4	SPI0_MISO0	EADC0_CH4	DPWM_LN		
PA.5	SPI0_CLK	EADC0_CH5	DPWM_LP		
PA.6	SPI0_SS0	EADC0_CH6			
PA.7	UART0_TXD	EADC0_CH7		SPI2_MISO	
PA.8	UART0_RXD	EADC0_CH8		SPI2_MOSI	
PA.9	I2C0_SCL	EADC0_CH9		SPI2_SS	
PA.10	I2C0_SDA	EADC0_ST	DPWM_RN	SPI2_CLK	
PA.11	I2C0_SMBSUS	TM0	DPWM_RP		
PA.12	I2C0_SMBAL	TM0_EXT		SPI2_I2SMCLK	
PA.13	CLKO	INT0	DPWM_SN	I2C1_SCL	
PA.14	SPI0_SS0	TM1	DPWM_SP	I2C1_SDA	
PA.15	INT0	TM1_EXT			
PB.0	PWM0_SYNC_IN	I2C0_SCL	PWM0_CH0		
PB.1	PWM0_SYNC_OUT	I2C0_SDA	PWM0_CH1		
PB.2	PWM0_CH0	TM2	PWM0_CH2		
PB.3	PWM0_CH1	TM2_EXT	DMIC_DAT1	UART0_RXD	PWM0_CH3
PB.4	UART0_nCTS	PWM0_CH0	DMIC_CLK1	UART0_TXD	PWM0_CH4
PB.5	XT1_OUT	PWM0_CH1	I2C0_SDA	I2C1_SDA	DMIC_DAT0
PB.6	XT1_IN	PWM0_CH2	I2C0_SCL	I2C1_SCL	DMIC_CLK0
PB.7	UART0_nRTS	PWM0_CH3			
PB.8	UART0_TXD	PWM0_CH4			
PB.9	UART0_RXD	PWM0_CH5			
PB.13	USB_D+	I2S0_DI			
PB.14	USB_D-	I2S0_DO			
PB.15	USB_VBUS	I2S0_MCLK			
PC.0	I2C1_SCL	X32_OUT	SPI1_MOSI		
PC.1	I2C1_SDA	X32_IN	SPI1_MISO		

MFP0	MFP1	MFP2	MFP3	MFP4	MFP5
PC.2	I2C1_SMBSUS	TM3	SPI1_CLK		
PC.3	I2C1_SMBAL	TM3_EXT	SPI1_SS		
PC.4	PWM0_CH2	CLKO	SPI1_I2SMCLK		
PC.5	INT1	SPI2_MOSI			
PC.6	INT2	SPI2_MISO			
PC.7	SPI0_SS0	SPI2_CLK			
PC.8	SPI0_MOSI1	SPI2_SS			
PC.9	SPI0_MISO1	SPI2_I2SMCLK			
PC.10	SPI0_MOSI0	PWM0_BRAKE0	DPWM_RN		
PC.11	SPI0_MISO0	PWM0_BRAKE1	DPWM_RP		
PC.12	SPI0_CLK		DPWM_LN		
PC.13	PWM0_CH3	I2C0_SCL	DPWM_LP		
PC.14	PWM0_CH4	I2C0_SDA	DPWM_SN		
PC.15	SPI0_SS1		DPWM_SP		
PD.0	INT3	I2C1_SCL	I2C0_SCL	I2S0_BCLK	DPWM_LN
PD.1	INT4	I2C1_SDA	I2C0_SDA	I2S0_LRCK	DPWM_LP
PD.2	TRACE_CLK	SPI1_MOSI	I2S0_MCLK	I2C1_SCL	TM0
PD.3	TRACE_DATA0	SPI1_MISO	I2S0_LRCK	DMIC_CLK1	TM2
PD.4	TRACE_DATA1	SPI1_CLK	I2S0_DI	DMIC_DAT1	TM1
PD.5	TRACE_DATA2	SPI1_SS	I2S0_DO	DMIC_CLK0	DPWM_RN
PD.6	TRACE_DATA3	SPI1_I2SMCLK	I2S0_BCLK	DMIC_DAT0	DPWM_RP
PD.7	PWM0_CH5	INT1			
PD.8	ICE_CLK	TM0	I2C1_SCL	I2C0_SCL	DPWM_SN
PD.9	ICE_DAT	TM0_EXT	I2C1_SDA	I2C0_SDA	DPWM_SP
PD.10	INT5	EADC0_ST			
PD.11	UART0_TXD	INT2			
PD.12	UART0_RXD	INT3	PWM0_CH3	INT0	
PD.13	SPI0_SS1	EADC0_CH10			
PD.14	UART0_nCTS	EADC0_CH11	I2C0_SCL	UART0_TXD	I2C1_SCL
PD.15	UART0_nRTS	EADC0_CH12	I2C0_SDA	UART0_RXD	I2C1_SDA

Table 4.5-1 GPIO Alternate Function Summary

5 BLOCK DIAGRAM

5.1 NPCA121 Series Block Diagram

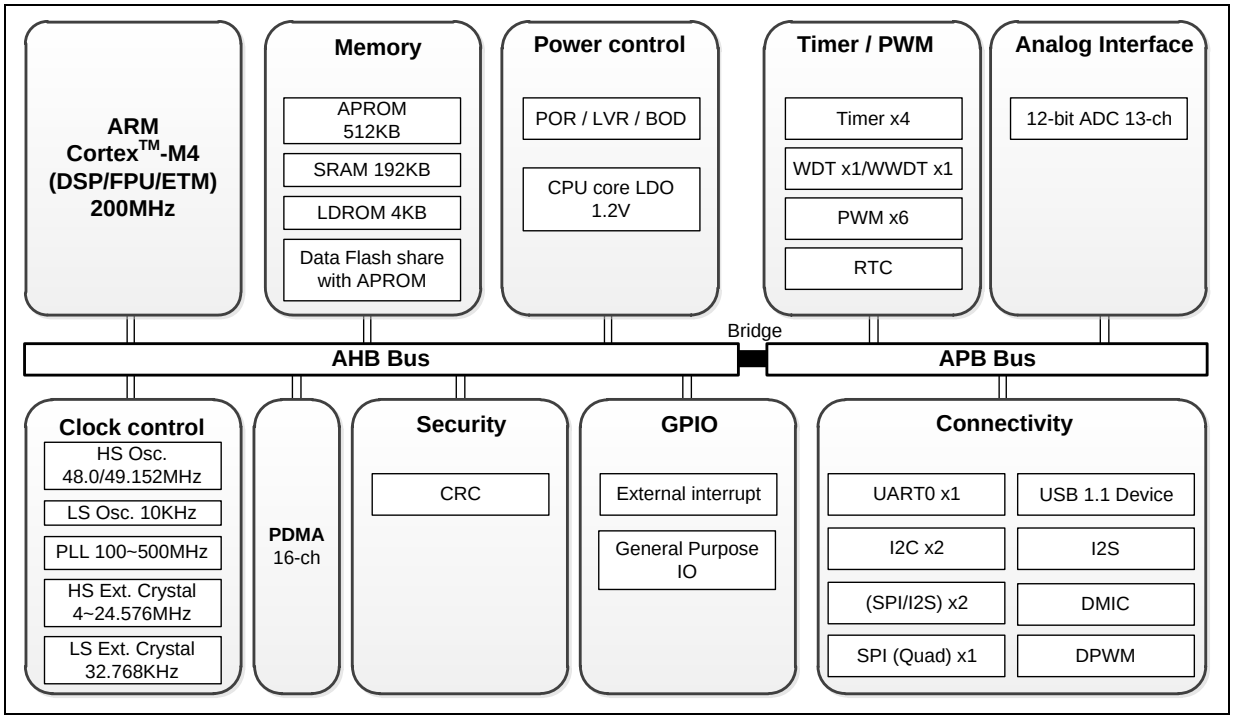


Figure 5.1-1 NPCA121 Series Block Diagram

6 FUNCTIONAL DESCRIPTION

6.1 ARM® Cortex®-M4 Core

The Cortex®-M4 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®-M4F is a processor with the same capability as the Cortex®-M4 processor and includes floating point arithmetic functionality. The NPCA121 series contains an embedded Cortex®-M4F processor. Throughout this document, the name Cortex®-M4 refers to both Cortex®-M4 and Cortex®-M4F processors. The following figure shows the functional controller of the processor.

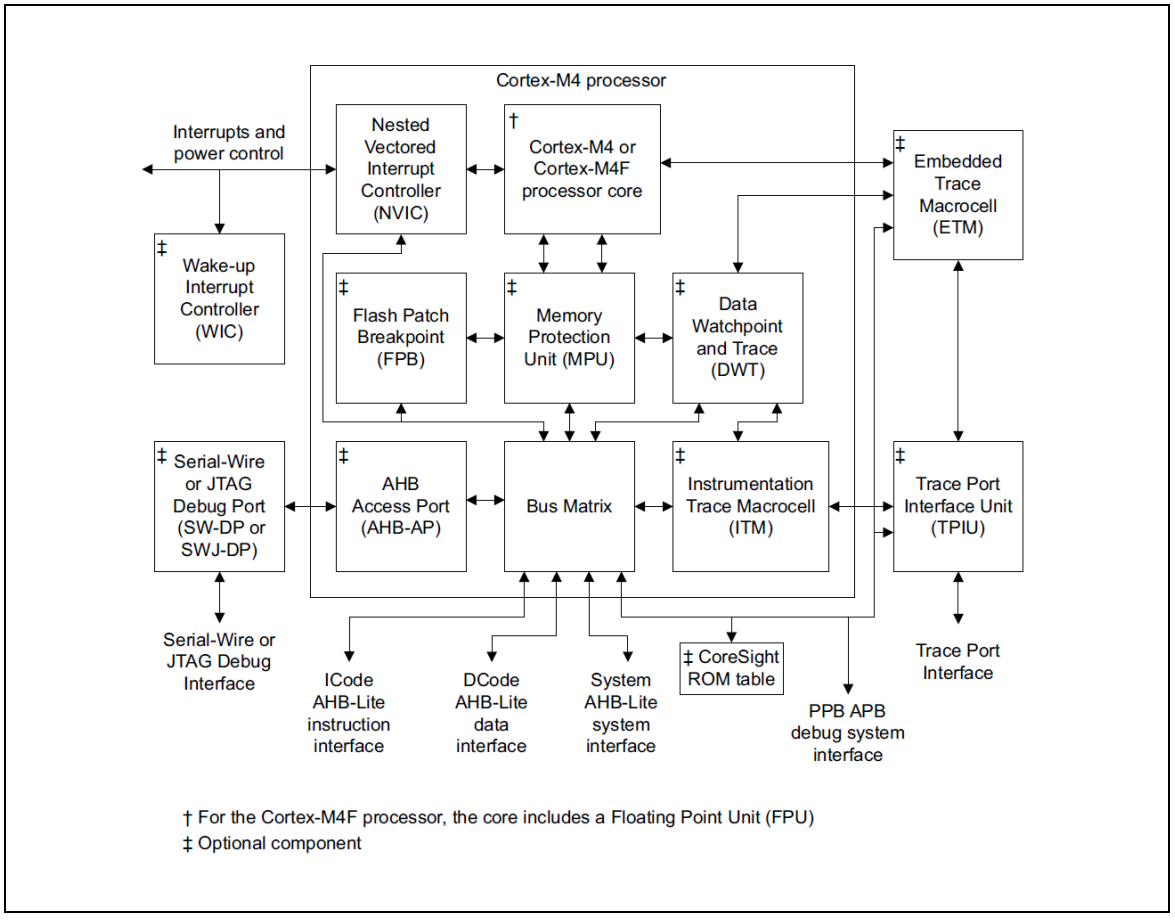


Figure 6.1-1 Cortex®-M4 Block Diagram

Cortex®-M4 processor features:

- A low gate count processor core, with low latency interrupt processing that has:
  - A subset of the Thumb instruction set, defined in the *ARMv7-M Architecture Reference Manual*
  - Banked Stack Pointer (SP)

- Hardware integer divide instructions, SDIV and UDIV
- Handler and Thread modes
- Thumb and Debug states
- Support for interruptible-continued instructions LDM, STM, PUSH, and POP for low interrupt latency
- Automatic processor state saving and restoration for low latency *Interrupt Service Routine (ISR)* entry and exit
- Support for ARMv6 big-endian byte-invariant or little-endian accesses
- Support for ARMv6 unaligned accesses

- Floating Point Unit (FPU) in the Cortex®-M4F processor providing:
  - 32-bit instructions for single-precision (C float) data-processing operations
  - Combined Multiply and Accumulate instructions for increased precision (Fused MAC)
  - Hardware support for conversion, addition, subtraction, multiplication with optional accumulate, division, and square-root
  - Hardware support for denormals and all IEEE rounding modes
  - 32 dedicated 32-bit single precision registers, also addressable as 16 double-word registers
  - Decoupled three stage pipeline
- Nested Vectored Interrupt Controller (NVIC) closely integrated with the processor core to achieve low latency interrupt processing. Features include:
  - External interrupts. Configurable from 1 to 240 (the NPCA121 series configured with 97 interrupts)
  - Bits of priority, configurable from 3 to 8
  - Dynamic reprioritization of interrupts
  - Priority grouping which enables selection of preempting interrupt levels and non-preempting interrupt levels
  - Support for tail-chaining and late arrival of interrupts, which enables back-to-back interrupt processing without the overhead of state saving and restoration between interrupts.
  - Processor state automatically saved on interrupt entry, and restored on interrupt exit with on instruction overhead
  - Support for Wake-up Interrupt Controller (WIC) with Ultra-low Power Sleep mode
- Memory Protection Unit (MPU). An optional MPU for memory protection, including:
  - Eight memory regions
  - Sub Region Disable (SRD), enabling efficient use of memory regions
  - The ability to enable a background region that implements the default memory map attributes
- Low-cost debug solution that features:
  - Debug access to all memory and registers in the system, including access to memory mapped devices, access to internal core registers when the core is halted, and access to debug control registers even while SYSRESETn is

asserted.

- Serial Wire Debug Port(SW-DP) debug access
- Optional Flash Patch and Breakpoint (FPB) unit for implementing breakpoints and code patches
- Optional Data Watchpoint and Trace (DWT) unit for implementing watchpoints, data tracing, and system profiling
- Optional Instrumentation Trace Macrocell (ITM) for support of printf() style debugging
- Optional Trace Port Interface Unit (TPIU) for bridging to a Trace Port Analyzer (TPA), including Single Wire Output (SWO) mode

- Bus interfaces:
  - Three Advanced High-performance Bus-Lite (AHB-Lite) interfaces: ICode, Dcode, and System bus interfaces
  - Private Peripheral Bus (PPB) based on Advanced Peripheral Bus (APB) interface
  - Bit-band support that includes atomic bit-band write and read operations.
  - Memory access alignment
  - Write buffer for buffering of write data
  - Exclusive access transfers for multiprocessor systems

6.2 System Manager

6.2.1 Overview

System management includes the following sections:

- System Reset
- System Power Distribution
- SRAM Memory Organization
- System Timer (SysTick)
- Nested Vectored Interrupt Controller (NVIC)
- System Control register

6.2.2 System Reset

A system reset sets all registers to their reset values except some of the registers listed in Table 6.2.2-1, which will maintain their original values after reset.
A system reset can be triggered by one of the nine sources listed below. The reset source can be identified by checking the reset flag bits in the System Reset Status Register (SYS_RSTSTS).

- Hardware Reset Sources
  - Power-on Reset
  - 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: writing 1 to CHIPRST (SYS_IPRST0[0]) will reset whole chip.
  - MCU Reset: writing 1 to SYSRESETREQ (AIRCR[2]) will reboot the device, according to the boot selection defined in configuration byte CONFIG0.
  - CPU Reset: writing 1 to CPURST (SYS_IPRST0[1]) will reset Cortex®-M4 core Only.

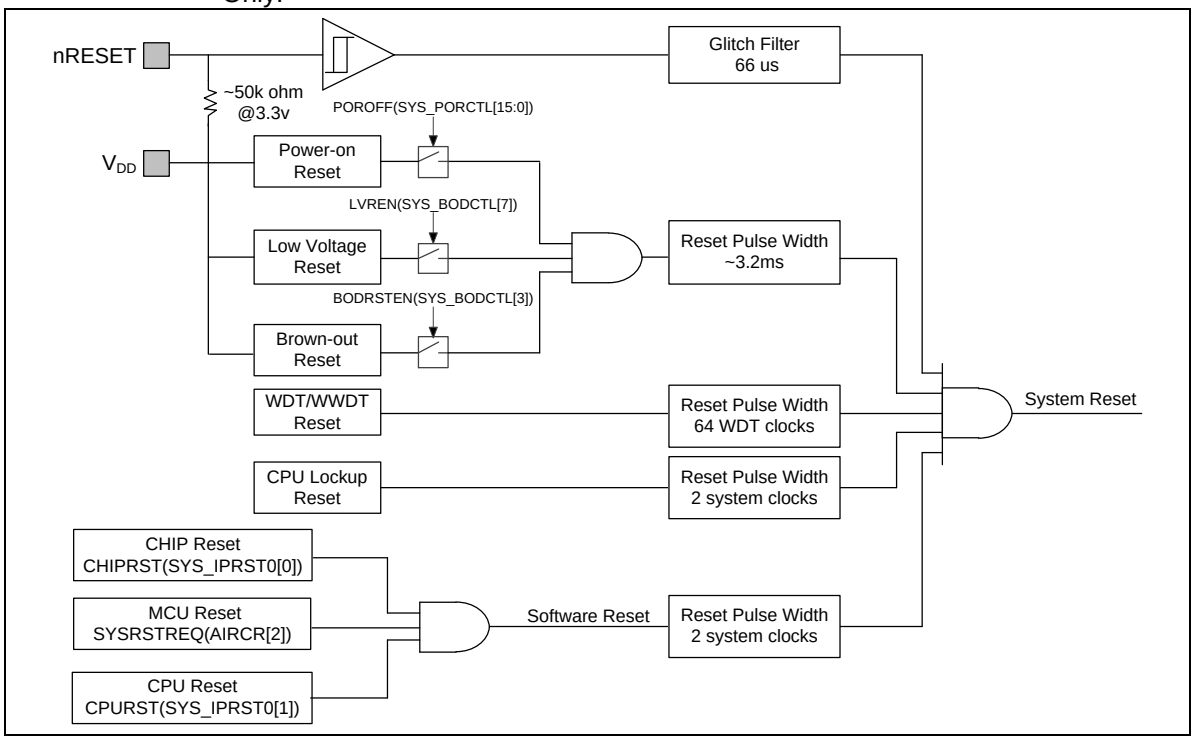


Figure 6.2-1 System Reset Sources

There are a total of 9 reset sources in the NPCA121 series. In general, CPU reset is used to reset Cortex®-M4 only; the other reset sources will reset Cortex®-M4 and all peripherals. However, there are small differences between each reset source and they are listed in Table 6.2.2-1.

Reset Sources Register	POR	NRESET	WDT	LVR	BOD	Lockup	CHIP	MCU	CPU
SYS_RSTSTS	Bit 0 = 1	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[18:16])**									
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])**	0x1	-	-	-	-	-	-	-	-
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	-	-
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 based on CONFIG0	Reload based on CONFIG0	Reload based on CONFIG0	Reload based on CONFIG0	Reload based on CONFIG0	-	Reload based on CONFIG0	-	-
Other Peripheral Registers	Reset Value								-
FMC Registers	Reset Value								
Note: '-' means that the value of register keeps original setting.									

Table 6.2.2-1 Reset Value of Registers

6.2.2.1 nRESET Reset

nRESET is triggered by pulling nRESET pin low. nRESET pin is an asynchronous reset input pin so that it can be used to reset the system any time.

Pull nRESET pin to < 0.3* V_{DD} for > 66us will reset the chip. Once entered reset state, only high voltage (> 0.7* V_{DD}) presented on nRESET pin for >66us can exit the reset state. See nRESET reset waveform in Figure 6.2-2.

After an nRESET reset, the PINRF bit (SYS_RSTSTS[1]) will be set, indicating the previous reset source is an nRESET reset. Writing 1 to PINRF clears the bit.

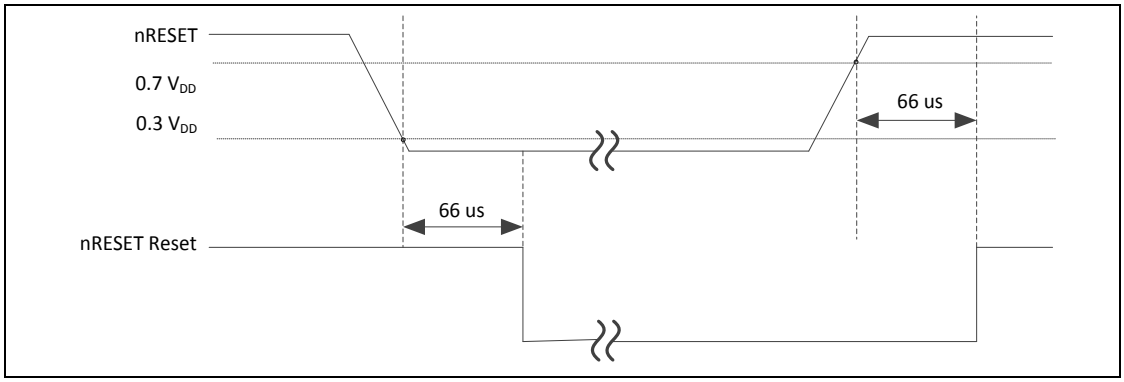


Figure 6.2-2 nRESET Reset Waveform

6.2.2.2 Power-on Reset (POR)

System power-on generates the Power-on reset (POR). When power is applied, the POR module

detects the rising voltage and generates reset signal. The reset signal stays active until the voltage is ready for MCU operation. PORF bit (SYS_RSTSTS[0]) will be set to 1 to indicate a POR reset event. The PORF bit (SYS_RSTSTS[0]) can be cleared by writing 1 to it.

6.2.2.3 Low Voltage Reset (LVR)

Writing 1 to LVREN bit (SYS_BODCTL[7]) enables the Low Voltage Reset (LVR) function. If enabled, the LVR function module keeps monitoring V_{DD} during system operation.
If V_{DD} voltage has been lower than V_{LVR}, a LVR reset will be triggered and the chip will be reset. Once triggered, the LVR reset keeps the chip in reset state until the V_{DD} voltage rises back and has been above V_{LVR}.
By default, Low Voltage Reset is enabled without De-glitch function.

6.2.2.4 Brown-out Detector Reset (BOD Reset)

Writing 1 to Brown-out Detector Enable Bit BODEN (SYS_BODCTL[0]) enables the Brown-out Detector (BOD) function. In addition, writing 1 to the BODRSTEN (SYS_BODCTL[3]) enables the BOD reset function. When enabled, the BOD module monitors V_{DD} during system operation, if V_{DD} voltage is lower than V_{BOD} (defined by BODVL bits in SYS_BODCTL[18:16]) for more than De-glitch time (defined by BODDGSEL bits in SYS_BODCTL[10:8]), a BOD reset will be triggered. The BOD reset keeps the chip in reset state until a condition is met that V_{DD} voltage has been higher than V_{BOD} for more than De-glitch time (defined by BODDGSEL).
Initial values of BODEN, BODVL and BODRSTEN (SYS_BODCTL[3]) are defined in flash configuration byte CONFIG0. Figure 6.2-3 shows the Brown-out Detector waveform.

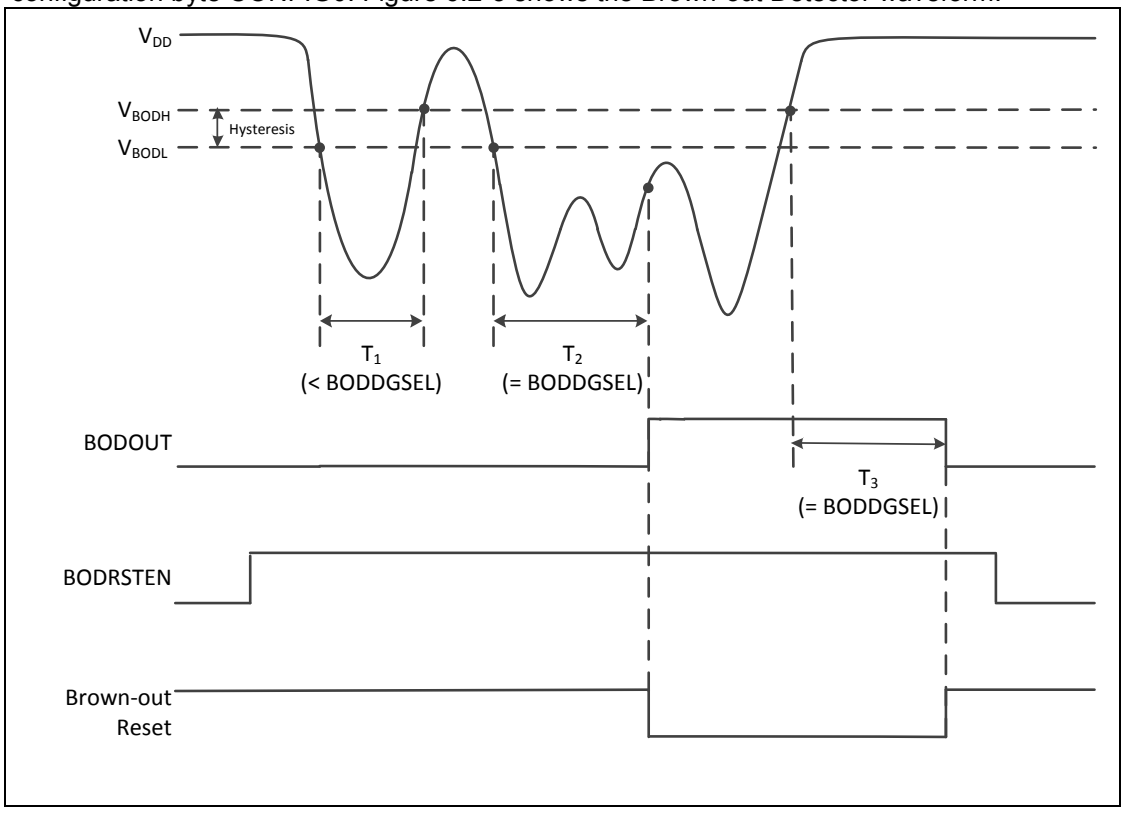


Figure 6.2-3 Brown-out Detector (BOD) Waveform

6.2.2.5 Watchdog Timer Reset (WDT)

A Watchdog reset can be identified by checking WDTRF bit (SYS_RSTSTS[2]).

6.2.2.6 CPU Lockup Reset

CPU enters lockup state after CPU produces hardfault at hardfault handler. 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 is in debug mode, the CPU lockup reset will be ignored.

6.2.2.7 CPU Reset, CHIP Reset and MCU Reset

A CPU Reset only resets the Cortex®-M4 core, while all other peripherals keep the state as is. Writing 1 to CPURST bit(SYS_IPRST0[1]) triggers a CPU reset.

CHIP Reset acts the same as Power-on Reset. The CPU and all peripherals are reset and BS(FMC_ISPCTL[1]) bit is automatically reloaded from CONFIG0. Writing 1 to CHIPRST bit (SYS_IPRST0[1]) triggers a CHIP Reset.

MCU Reset is similar to CHIP Reset as the only difference is that MCU Reset does not reload BS(FMC_ISPCTL[1]) from CONFIG0; so where device will boot from (APROM or LDROM) still depends on the old value in BS(FMC_ISPCTL[1]). Writing 1 to SYSRESETREQ bit (AIRCR[2]) triggers a MCU Reset.

6.2.2.8 Reset Flag

After wake-up or reset from each power modes, reset flag in SYS_RSTSTS register will take effect when system return Normal mode. Table 6.2.2-2 shows the LVRF (SYS_RSTSTS[3]), PINRF (SYS_RSTSTS[1]) and PORF (SYS_RSTSTS[0]) effect when wake-up or reset from different power modes. Note that LVR function cannot be used in DPD mode.

	Conditions				**Flag**		
Power Mode	Power-on Reset	Low Voltage Reset	nRESET Reset	Wakeup	LVRF SYS_RSTSTS[3]	PINRF SYS_RSTSTS[1]	PORF SYS_RSTSTS[0]
Normal / Idle	V	-	-	-	V	V	V
	-	V	-	-	V	-	-
	-	-	V	-	-	V	-
SPD0/1	V	-	-	-	V	V	V
	-	V	-	-	V	V	V
	-	-	V	-	-	V	V
	-	-	-	V	-	V	V
DPD	V	-	-	-	V	V	V
	-	-	V	-	-	V	V
	-	-	-	V	V	V	V

Table 6.2.2-2 Reset Flag Table

6.2.3 System Power Distribution

NPCA121 series device power distribution is divided into:

- 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 regulated 1.2 V power for digital operation.
- USB transceiver power from USB_V_{DD33} offers the power for operating the USB transceiver.

Analog power (AV_{DD}) should be at the same voltage level as digital power (V_{DD}).
Both power supplies should have decoupling capacitors placed as close as possible to pins preferably with no via.
The outputs of internal voltage regulator, LDO_CAP, requires an external capacitor which should be located close to LDO_CAP pin and returned directly to V_{SS}. The Figure 6.2-4 shows the power distribution of the NPCA121 series.

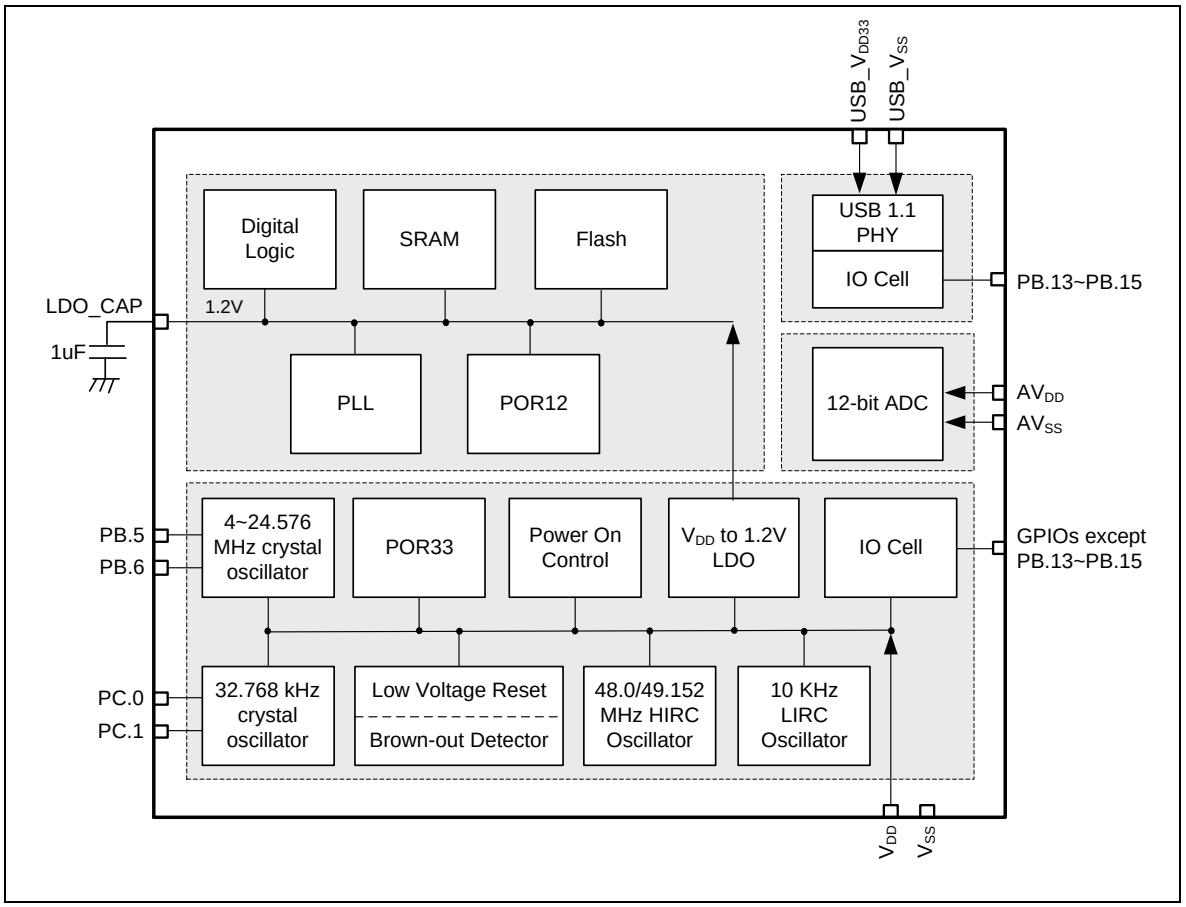


Figure 6.2-4 NPCA121 Series Power Distribution Diagram

6.2.4 Power Modes

The NPCA121 series power mode include Normal mode, Idle mode and Power-down mode. The system starts up in Normal mode. In Idle mode, only CPU clock is disabled while other peripherals work normally. If system is waiting for an interrupt only, the user can set system in Idle mode to save power and wake-up quickly. The user can set system into Power-down mode when system does not need to work for a long time. NPCA121 series provides several power-down modes with different power consumption level.

The NPCA121 series includes the following power modes:

Normal mode
The system starts up in Normal mode. All clock sources and peripheral can be enabled or disabled by user in register CLK_PWRCTL (System Power-down Control Register), CLK_AHBCLK (AHB Devices Clock Enable Control Register), CLK_APBCLK0 (APB Devices Clock Enable Control Register 0) and CLK_APBCLK1 (APB Devices Clock Enable Control Register 1). The user can disable unused clock or peripheral to save power.

Idle mode
If system is waiting for an interrupt only. The user can set system in idle mode. In idle mode, only CPU clock is disabled, other peripherals work normally. System waits interrupt to wake-up, returns to Normal mode and program execution continues. All interrupts can wake-up system from Idle mode.

Power-down mode (PD)
In Power-down mode (PD), CPU clock is disabled and LDO enters low power mode. All clock source will be disabled except LXT and LIRC, LXT and LIRC can be controlled by CLK_PWRCTL register. If they are enabled and peripheral clock source are selected as LXT or LIRC, the peripheral can keep working in Power-down mode. System waits wake-up source occurred to wake-up, returns to Normal mode and program execution continues. In system wake-up phase, system waits for LDO recovery, and clock sources are enabled again and stable.

Low Leakage Power-down mode (LLPD)
In Low Leakage Power-down mode (LLPD), CPU clock is disabled, LDO enters low power mode and LDO voltage drops down from current working voltage to 0.9 V to save power. All clock source will be disabled except LXT and LIRC, LXT and LIRC can be controlled by CLK_PWRCTL register. If they are enabled and peripheral clock source are selected as LXT or LIRC, the peripheral can keep working in Low Leakage Power-down mode. System waits wake-up source occurred to wake-up, returns to Normal mode and program execution continues. In system wake-up phase, besides waiting for LDO recovery and clock sources are enabled and stable, system also needs to wait for LDO voltage rising to the original working voltage. The Low Leakage Power-down mode wake-up sources are the same as Power-down mode.

Standby Power-down mode 0 (SPD0)
In Standby Power-down mode 0 (SPD0), all power supply is disabled except SPD0 control logic, LXT and LIRC. The SPD0 control logic controls Standby Power-down mode wake-up functions and system SRAM bank0 to retain data. After wake-up from Standby Power-down mode 0 (SPD0), system resets and executes code from the beginning again. All peripheral configurations return to default value and all SRAM data will be lost except system SRAM bank0 data can be retained.
After system wake-up from Standby Power-down mode 0 (SPD0), GPIO will keep their states before entering Standby Power-down mode 0. They cannot be controlled by GPIO or peripherals register after system wake-up, for example: UART cannot print message and ICE cannot download code. To control GPIO, the user have to write 1 to register CLK_IOPDCTL (GPIO Standby Power-down Control Register) after system wake-up to release GPIO hold state function.

Standby Power-down mode 1 (SPD1)
In Standby Power-down mode 1 (SPD1), all power supply is disabled except SPD1 control logic, LXT and LIRC. The SPD1 control logic controls Standby Power-down mode wake-up functions. After wake-up from Standby Power-down mode 1 (SPD1), system resets and executes code from the beginning again. All peripheral configurations return to default value and all SRAM data will be lost.
After system wake-up from Standby Power-down mode 1 (SPD1), GPIO will keep their states before entering Standby Power-down mode 1. They cannot be controlled by GPIO or peripherals register after system wake-up, for example: UART cannot print message and ICE cannot download code. To control GPIO, the user have to write 1 to register CLK_IOPDCTL (GPIO Standby Power-down Control Register) after system wake-up to release GPIO hold state function.

Deep Power-down mode (DPD)
In Deep Power-down mode (DPD), all power supply is disabled except DPD control logic. The DPD control logic controls Deep Power-down mode wake-up functions. After wake-up from Deep Power-down mode (DPD), system resets and executes code from the beginning again. All peripheral configurations return to default value and all SRAM data will be lost. The Deep Power-down mode (DPD) wake-up sources include Wake-up Timer and Wake-up Pin (PA.15).

6.2.5 Power Modes Settings and Wake-up Sources

NPCA121 series is equipped with power management unit to support different power modes for saving power. Table 6.2.5-1 lists all power mode at NPCA121 series.

Power Mode	CPU operating maximum speed (MHz)	LDO_CAP (V)	Clock Disable
Normal mode	160	1.20	All clocks are disabled by control register.
Turbo mode	200	1.26	All clocks are disabled by control register.
Idle mode	CPU enter Sleep mode	1.20	Only CPU clock is disabled.
Power-down mode (PD)	CPU enters Deep Sleep mode	1.20	Most clocks are disabled except LIRC/LXT, and only RTC/WDT/Timer/UART peripheral clocks still enable if their clock sources are selected as LIRC/LXT.
Low leakage Power-down mode (LLPD)	CPU enters Deep Sleep mode	0.9	Most clocks are disabled except LIRC/LXT, and only RTC/WDT/Timer/UART peripheral clocks still enable if their clock sources are selected as LIRC/LXT.
Standby Power-down mode 0 (SPD0)	Power off	Floating	Only LIRC still enable for wake-up timer usage
Standby Power-down mode 1 (SPD1)	Power off	Floating	Only LIRC still enable for wake-up timer usage
Deep Power-down mode (DPD)	Power off	Floating	Only LIRC still enable for wake-up timer usage

Table 6.2.5-1 Power Mode Table

There are different power mode entry settings. Each power mode has different entry setting and leaving condition. Table 6.2.5-2 shows the enry setting for each power mode. When chip power-on, chip is running at normal mode. User can enter each mode by configuring SLEEPDEEP (SCR[2]), PDEN (CLK_PWRCT[7]) and PDMSEL (CLK_PMUCTL[2:0]) bits and then execute WFI instruction.

Register/Instruction Mode	SLEEPDEEP (SCR[2])	PDEN (CLK_PWRCTL[7])	PDMSEL (CLK_PMUCTL[2:0])	CPU Run WFI Instruction
Normal mode	0	0	0	NO
Idle mode	0	0	0	YES
Power-down mode	1	1	0	YES
Low leakage Power-down mode	1	1	1	YES
Standby Power-down mode 0	1	1	4	YES
Standby Power-down mode 1	1	1	5	YES
Deep Power-down mode	1	1	6	YES

Table 6.2.5-2 Power Mode Difference Table

Note:

1. User must turn on LIRC before entering PD, LLPD and SPD0/1 mode.

There are several wake-up sources in Idle mode and Power-down mode. Table 6.2.5-3 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 retained.
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	RTC, WDT, I²C, Timer, UART, BOD, GPIO, EINT, 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 6.2.5-3 Power Mode Difference Table

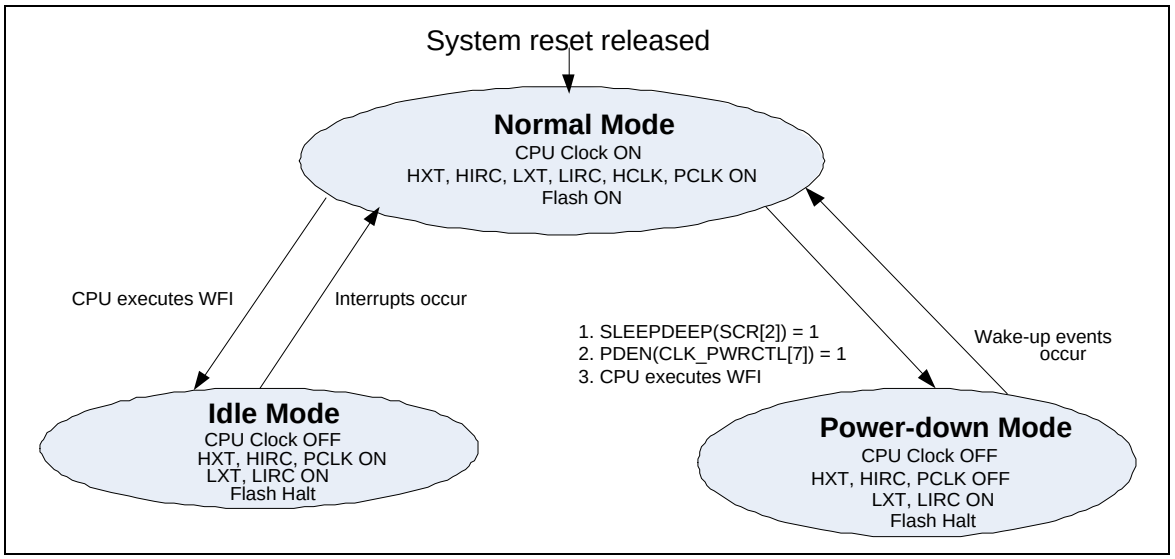


Figure 6.2-5 NPCA121 Series Power Mode State Machine

	Normal Mode	Idle Mode	PD, LLPD	SPD0, SPD1	DPD
HXT	ON	ON	Halt	Halt	Halt
HIRC	ON	ON	Halt	Halt	Halt
LXT	ON	ON	ON/OFF[1]	ON/OFF[1]	Halt
LIRC	ON	ON	ON	ON	ON/OFF[2]
PLL	ON	ON	Halt	Halt	Halt
HCLK/PCLK	ON	ON	Halt	Halt	Halt
CPU	ON	Halt	Halt	Halt	Halt
SRAM retention	ON	ON	ON	Halt	Halt
FLASH	ON	ON	Halt	Halt	Halt
TIMER	ON	ON	ON/OFF[3]	Halt	Halt
WDT	ON	ON	ON/OFF[4]	Halt	Halt
RTC	ON	ON	ON/OFF[5]	ON/OFF[5]	Halt
UART	ON	ON	ON/OFF[6]	Halt	Halt
Others	ON	ON	Halt	Halt	Halt

Table 6.2.5-4 Clocks in Power Modes

Note:

1. LXT ON or OFF depends on S/W setting in normal mode.
2. LIRC ON or OFF depends on S/W setting in normal 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 RTC clock source is selected as LXT and LXT is on.
6. If UART clock source is selected as LXT and LXT is on.

Wake-up sources in Power-down mode:

Table 6.2.5-5 lists all wake-up sources that can wake chip up from power down mode to normal mode, wake-up condition and conditions on how to re-enter the power down mode again.

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

Note that before entering each power down mode, user needs to enable wake-up source NVIC interrupt and make sure that other peripherals' interrupt are disabled. In addition, if BOD function is enabled, BOD NVIC interrupt must be enabled before entering power down mode.

Wake-Up Source	Wake-Up Condition	Power down mode			Condition that System Re-enter Power-Down Mode *
		PD/ LLPD	SPD0/1	DPD	
BOD	Brown-Out Detector Interrupt	Y	N	N	After software writes 1 to clear BODIF (SYS_BODCTL[4]).
BOD	Brown-Out Detector Interrupt	N	Y	N	BODWK(CLK_PMUSTS[4]) is cleared when SPD mode is entered.
LVR	LVR Reset	Y	Y	N	After software writes 1 to clear LVRF(SYS_RSTSTS[3])
POR	POR Reset	N	N	Y	PORWK(CLK_PMUSTS[0]) is cleared when DPD mode is entered.
POR	POR Reset	Y	Y	N	After software writes 1 to clear PORF(SYS_RSTSTS[0])
INT	External Interrupt	Y	N	N	After software write 1 to clear the Px_INTSRC[n] bit.
GPIO	GPIO Interrupt	Y	N	N	After software write 1 to clear the Px_INTSRC[n] bit.
GPIO(PA~PD) Wake-up pin	rising or falling edge event, 58-pin	N	Y	N	GPxWK(CLK_PMUSTS[11:8]) is cleared when SPD mode is entered.
GPIO(PA.15) Wake-up pin	rising or falling edge event , 1-pin	N	N	Y	PINWK(CLK_PMUSTS[1]) is cleared when DPD mode is entered.
TIMER	Timer Interrupt	Y	N	N	After software writes 1 to clear TWKF (TIMERx_INTSTS[1]) and TIF (TIMERx_INTSTS[0]).
Wakeup timer	Wakeup by wake-up timer time-out	N	Y	Y	DPD_TMRWK (CLK_PMUSTS[2]) or SPD_TMRWK (CLK_PMUSTS[6]) is cleared when SPD or DPD mode is entered.
WDT	WDT Interrupt	Y	N	N	After software writes 1 to clear WKF (WDT_CTL[5]) (Write Protected).
RTC	Alarm Interrupt	Y	N	N	After software writes 1 to clear ALMIF (RTC_INTSTS[0]).
	Time Tick Interrupt	Y	N	N	After software writes 1 to clear TICKIF (RTC_INTSTS[1]).
RTC	Wakeup by RTC alarm	N	Y	N	RTCWK(CLK_PMUSTS[5]) is cleared when SPD mode is entered.
	Wakeup by RTC tick time	N	Y	N	RTCWK(CLK_PMUSTS[5]) is cleared when SPD mode is entered.
UART	nCTS wake-up	Y	N	N	After software writes 1 to clear CTSWKF (UARTx_WKSTS[0]).
	RX Data wake-up	Y	N	N	After software writes 1 to clear DATWKF (UARTx_WKSTS[1]).
	Received FIFO Threshold Wake-up	Y	N	N	After software writes 1 to clear RFRTWKF (UARTx_WKSTS[2]).

	RS-485 AAD Mode Wake-up	Y	N	N	After software writes 1 to clear RS485WKF (UARTx_WKSTS[3]).
	Received FIFO Threshold Time-out Wake-up	Y	N	N	After software writes 1 to clear TOUTWKF (UARTx_WKSTS[4]).
I^2C	Address match wake-up	Y	N	N	After software writes 1 to clear WKAKDONE (I2C_WKSTS[1]). Then software writes 1 to clear WKIF(I2C_WKSTS[0]).
USBD	Remote Wake-up	Y	N	N	After software writes 1 to clear BUSIF (USBD_INTSTS[0]).

Table 6.2.5-5 Re-Entering Power-down Mode Condition

6.2.6 Brown-out Detector and Low Voltage Reset Controller Configuration

NPCA121 series is equipped with brown-out detector and low voltage reset controller function. Low voltage reset controller is enabled by setting LVREN(SYS_BODCTL[7]) to 1. Brown-out detector is enabled by setting both LVREN(SYS_BODCTL[7]) and BODEN(SYS_BODCTL[0]) to 1. Both brown-out detector and low voltage reset controller also integrates low power mode function, the low power mode is enabled by setting BODLPM(SYS_BODCTL[5]). When enable low power mode, brown-out detector and low voltage reset controller will consume less power, but the detection speed will become slow (The response time is about 13ms). When using brown-out detector and low voltage reset controller, the LIRC must be turned on. In addition, the brown-out detector cannot operate independently without enabled low voltage reset controller.

Power Mode	LVREN (SYS_BODCTL[7])	BODEN (SYS_BODCTL[0])	BODLPM (SYS_BODCTL[5])	LVR / BOD Operation Mode	
				LVR	BOD
Normal / Idle / PD	0	X	X	Disabled	Disabled
	1	0	X	Normal	Disabled
	1	1	0	Normal	Normal
	1	1	1	Low power	Low power
LLPD / SPD0 / SPD1	0	X	X	Disabled	Disabled
	1	0	X	Low power	Disabled
	1	1	0	Normal	Normal
	1	1	1	Low power	Low power
DPD	X	X	X	Disabled	Disabled

Table 6.2.6-1 Brown-out Detector and Low Voltage Reset Controller Effect Table

Note: X means don't care, write 1 or 0 to correspond register bit will not affect functionality.

6.2.7 System Memory Map

The NPCA121 series provides 4G-byte addressing space. The memory addresses assigned to each on-chip controllers are shown in Table 6.2.7-1. The detailed register definition, memory space, and programming will be described in the following sections for each on-chip peripheral. The NPCA121 series only supports little-endian data format.

Address Space	Token	Controllers
Flash and SRAM Memory Space		
0x0000_0000 – 0x0007_FFFF	FLASH_BA	FLASH Memory Space (512 Kbytes)
0x2000_0000 – 0x2000_7FFF	SRAM0_BA	SRAM Memory Space (32 Kbytes)
0x2000_8000 – 0x2002_FFFF	SRAM1_BA	SRAM Memory Space (160 Kbytes)
Peripheral Controllers Space (0x4000_0000 – 0x400F_FFFF)		
0x4000_0000 – 0x4000_01FF	SYS_BA	System Control Registers
0x4000_0200 – 0x4000_02FF	CLK_BA	Clock Control Registers
0x4000_0300 – 0x4000_03FF	NMI_BA	NMI Control Registers
0x4000_4000 – 0x4000_4FFF	GPIO_BA	GPIO Control Registers
0x4000_8000 – 0x4000_8FFF	PDMA_BA	Peripheral DMA Control Registers
0x4000_C000 – 0x4000_CFFF	FMC_BA	Flash Memory Control Registers
0x4003_1000 – 0x4003_1FFF	CRC_BA	CRC Generator Registers
APB Controllers Space (0x4000_0000 ~ 0x400F_FFFF)		
0x4004_0000 – 0x4004_0FFF	WDT_BA	Watchdog Timer Control Registers
0x4004_1000 – 0x4004_1FFF	RTC_BA	Real Time Clock (RTC) Control Register
0x4004_3000 – 0x4004_3FFF	EADC_BA	Enhanced Analog-Digital-Converter (EADC) Control Registers
0x4004_8000 – 0x4004_8FFF	I2S0_BA	I^2S0 Interface Control Registers
0x4005_0000 – 0x4005_0FFF	TMR01_BA	Timer0/Timer1 Control Registers
0x4005_1000 – 0x4005_1FFF	TMR23_BA	Timer2/Timer3 Control Registers
0x4005_8000 – 0x4005_8FFF	PWM0_BA	PWM0 Control Registers
0x4006_0000 – 0x4006_0FFF	SPI0_BA	SPI0 Control Registers
0x4006_1000 – 0x4006_1FFF	SPI1_BA	SPI1 Control Registers
0x4006_2000 – 0x4006_2FFF	SPI2_BA	SPI2 Control Registers
0x4006_3000 – 0x4006_30FF	DMIC_BA	DMIC Control Registers
0x4006_3100 – 0x4006_3FFF	VAD_BA	VAD Control Registers
0x4006_4000 – 0x4006_4FFF	DPWM_BA	DPWM Control Registers
0x4007_0000 – 0x4007_0FFF	UART0_BA	UART0 Control Registers
0x4008_0000 – 0x4008_0FFF	I2C0_BA	I^2C0 Control Registers
0x4008_1000 – 0x4008_1FFF	I2C1_BA	I^2C1 Control Registers
0x400C_0000 – 0x400C_0FFF	USBD_BA	USB Device Control Register

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 6.2.7-1 Address Space Assignments for On-Chip Controllers

6.2.8 SRAM Memory Organization

The NPCA121 series supports up to 192 KB of embedded SRAM and the SRAM organization is separated to two banks: SRAM bank0 and SRAM bank1. The SRAM bank0 supports parity error check to make sure chip operating more stable.

- Supports up to 192 KB of SRAM
- Supports byte / half word / word write
- Supports parity error check function for SRAM bank0
- Supports oversize response error

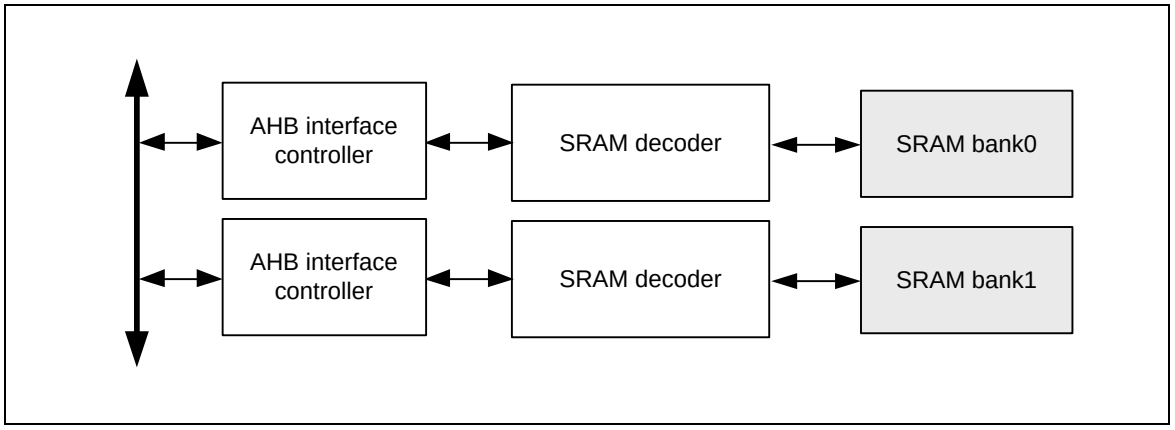


Figure 6.2-6 SRAM Block Diagram

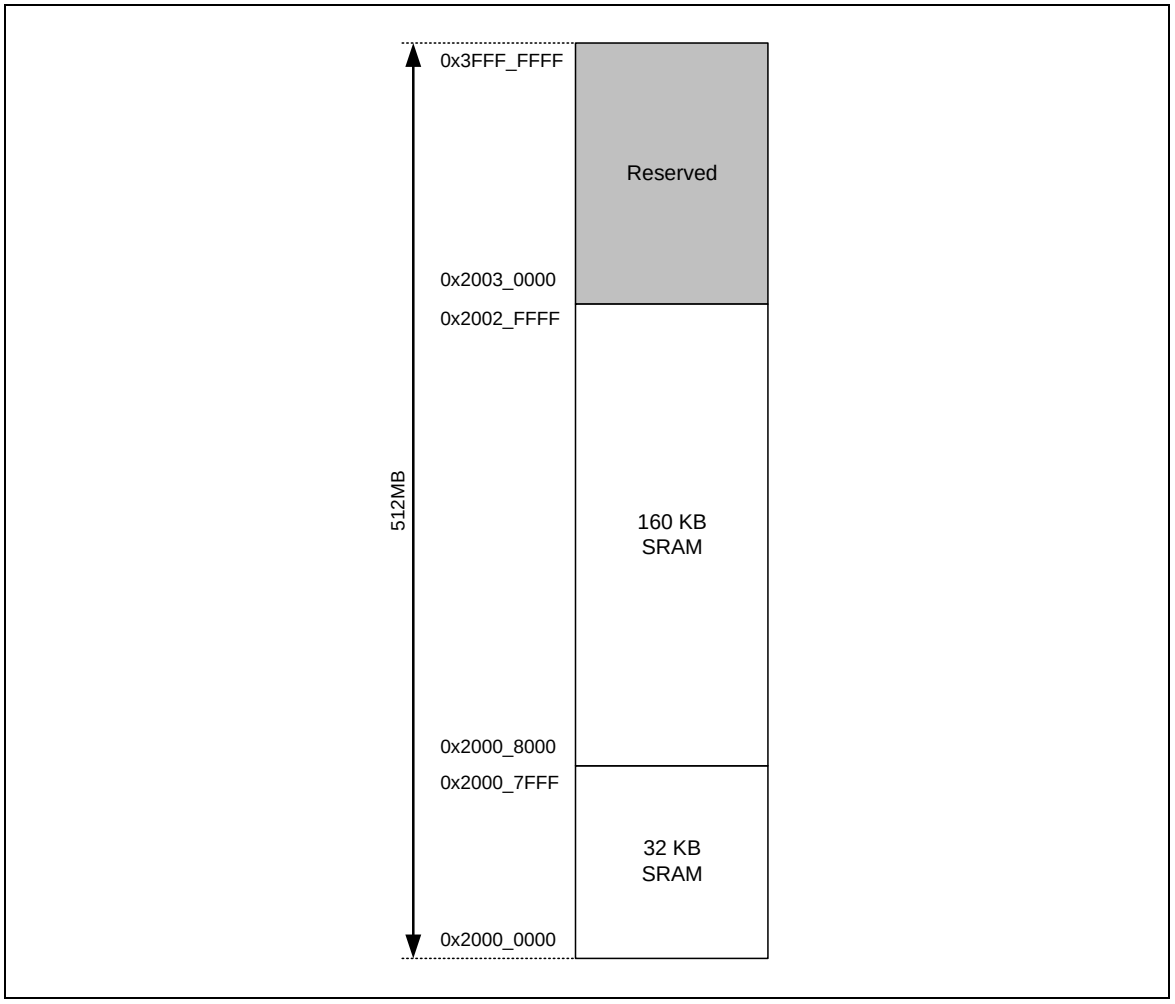


Figure 6.2-7 SRAM Memory Organization

SRAM address from 0x2000_0000 to 0x2000_7FFF has byte parity error check function. When CPU is accessing SRAM address from 0x2000_0000 to 0x2000_7FFF the parity error checking mechanism is operating dynamically. If parity error occurrs, the PERRIF (SYS_SRAM_STATUS[0]) will be asserted to 1 and the SYS_SRAM_ERRADDR register will record the address with parity error. Chip will enter interrupt when SRAM parity error occurs if PERRIEN (SYS_SRAM_INTCTL[0]) is set to 1. When SRAM parity error occurs, chip will stop detecting SRAM parity errors until user writes 1 to clear the PERRIF(SYS_SRAM_STATUS[0]) bit.

6.2.9 HIRC Auto Trim

This chip supports auto-trim function: the HIRC trim, according to the accurate LXT (32.768 kHz crystal oscillator) or USB SOF (Start-Of-Frame), automatically gets accurate HIRC output frequency, 0.25 % deviation within all temperature ranges.
For instance, the system needs an accurate 49.152 MHz or 48 MHz clock. 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” or “11”, and the auto-trim function will be enabled. Interrupt status bit FREQLOCK (SYS_IRCTISTS[8] HIRC frequency lock status) “1” indicates the HIRC output frequency is accurate within 0.25% deviation.

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 = 0x4000_0000**				
SYS_PDID	SYS_BA+0x00	R	Part Device Identification Number Register	0x1DXX_05XX [1]
SYS_RSTSTS	SYS_BA+0x04	R/W	System Reset Status Register	0x0000_0043
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	0x000X_038X
SYS_PORCTL	SYS_BA+0x24	R/W	Power-On-Reset Controller Register	0x0000_0000
SYS_USBPHY	SYS_BA+0x2C	R/W	USB PHY Control Register	0x0000_0000
SYS_GPA_MFPL	SYS_BA+0x30	R/W	GPIOA Low Byte Multiple Function Control Register	0x0000_0000
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	0x0110_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_0011
SYS_SRAM_INTCTL	SYS_BA+0xC0	R/W	System SRAM Interrupt Enable Control Register	0x0000_0000
SYS_SRAM_STATUS	SYS_BA+0xC4	R/W	System SRAM Parity Error Status Register	0x0000_0000
SYS_SRAM_ERRADDR	SYS_BA+0xC8	R	System SRAM Parity Check Error Address Register	0x0000_0000
SYS_IRCTCTL	SYS_BA+0xF0	R/W	HIRC Trim Control Register	0x0000_0000
SYS_IRCTIEN	SYS_BA+0xF4	R/W	HIRC Trim Interrupt Enable Register	0x0000_0000
SYS_IRCTISTS	SYS_BA+0xF8	R/W	HIRC Trim Interrupt Status Register	0x0000_0000
SYS_REGLCTL	SYS_BA+0x100	R/W	Register Lock Control Register	0x0000_0000
SYS_RCADJ	SYS_BA+0x110	R/W	HIRC Trim Value Register	0x0000_0XXX

Note:

1. Any register not listed here is reserved and must not be written. The result of a read operation on these bits is undefined.
2. The reserved register fields that listed in register description must be written to their reset value. Writing reserved fields with other than reset values may produce undefined results.

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	0x1DXX_05XX [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.

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_0043

31	30	29	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	PMURF	SYSRF	BODRF	LVRF	WDTRF	PINRF	PORF

Bits	Description	
[31:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8]	**CPULKRF**	**CPU Lockup Reset Flag** 0 = No reset from CPU lockup occurred. 1 = The Cortex-M4 lockup occurred and chip is reset. **Note:** Write 1 to clear this bit to 0. **Note:** when ICE is connected, CPU lockup event sets this flag to 1 but will not reset chip.
[7]	**CPURF**	**CPU Reset Flag** The CPU reset flag is set by hardware if software writes CPURST (SYS_IPRST0[1]) 1 to reset Cortex®-M4 Core and Flash Memory Controller (FMC). 0 = No reset from CPU. 1 = The Cortex®-M4 Core and FMC are reset by software setting CPURST to 1. **Note:** Write 1 to clear this bit to 0.
[6]	**PMURF**	**PMU Reset Flag** 0 = No reset from POR, PINR, WDTR, LVR, BODR, SYSR and CPULKR. 1 = When POR, PINR, WDTR, LVR, BODR, SYSR and CPULKR occurred. **Note:** Write 1 to clear this bit to 0.
[5]	**SYSRF**	**System Reset Flag** The system reset flag is set by the "Reset Signal" from the Cortex®-M4 Core to indicate the previous reset source. 0 = No reset from Cortex®-M4. 1 = The Cortex®-M4 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®-M4 core. **Note:** Write 1 to clear this bit to 0.

Bits	Description	
[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:** Write 1 to clear this bit to 0.
[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 issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.
[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 issued the reset signal to reset the system. **Note1:** Write 1 to clear this bit to 0. **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 issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.
[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 issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.

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
CRCRST	Reserved				PDMARST	CPURST	CHIPRST

Bits	Description	
[31: 8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7]	**CRCRST**	**CRC Calculation Controller Reset (Write Protected)** Set this bit to 1 will generate a reset signal to the CRC calculation controller. User needs to set this bit to 0 to release from the reset state. 0 = CRC calculation controller normal operation. 1 = CRC calculation controller reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[6:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**PDMARST**	**PDMA Controller Reset (Write Protected)** 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 Protected)** Setting this bit will only reset the processor core and Flash Memory Controller(FMC); 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 Protected)** Setting this bit will reset the whole chip, including Processor core and all peripherals; this bit will automatically return to 0 after the 2 clock cycles. The CHIPRST is same as the POR reset, all the chip controllers are reset and the chip settings from flash configuration are also reloaded. 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	HIRCCKF	I2S0RST	EADCRST	USBDRST	Reserved		
23	22	21	20	19	18	17	16
Reserved							UART0RST
15	14	13	12	11	10	9	8
DMICRST	SPI2RST	SPI1RST	SPI0RST	Reserved		I2C1RST	I2C0RST
7	6	5	4	3	2	1	0
Reserved		TMR3RST	TMR2RST	TMR1RST	TMR0RST	GPIORST	Reserved

Bits	Description	
[31]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[30]	**HIRCCKF**	**HIRC Clock Filter Enable Bit** 0 = HIRC clock filter Enabled. 1 = HIRC clock filter Disabled.
[29]	**I2S0RST**	**I^2S0 Controller Reset** 0 = I^2S0 controller normal operation. 1 = I^2S0 controller reset.
[28]	**EADCRST**	**EADC Controller Reset** 0 = EADC controller normal operation. 1 = EADC controller reset.
[27]	**USBDRST**	**USBD Controller Reset** 0 = USBD controller normal operation. 1 = USBD controller reset.
[26:17]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[16]	**UART0RST**	**UART0 Controller Reset** 0 = UART0 controller normal operation. 1 = UART0 controller reset.
[15]	**DMICRST**	**DMIC Controller Reset** 0 = DMIC controller normal operation. 1 = DMIC controller reset.
[14]	**SPI2RST**	**SPI2 Controller Reset**

		0 = SPI2 controller normal operation. 1 = SPI2 controller reset.
[13]	**SPI1RST**	**SPI1 Controller Reset** 0 = SPI1 controller normal operation. 1 = SPI1 controller reset.
[12]	**SPI0RST**	**SPI0 Controller Reset** 0 = SPI0 controller normal operation. 1 = SPI0 controller reset.
[11:10]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.

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							PWM0RST
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	DPWMRST	Reserved					

Bits	Description	
[31:17]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[16]	**PWM0RST**	**PWM0 Controller Reset** 0 = PWM0 controller normal operation. 1 = PWM0 controller reset.
[15:7]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[6]	**DPWMRST**	**DPWM Controller Reset** 0 = Audio DPWM controller normal operation. 1 = Audio DPWM controller reset.
[5:0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

Brown-out Detector Control Register (SYS_BODCTL)

Part of the SYS_BODCTL control registers values are initialized by flash configuration and writeable bits are write-protected.

Register	Offset	R/W	Description	Reset Value
SYS_BODCTL	SYS_BA+0x18	R/W	Brown-Out Detector Control Register	0x000X_038X

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

Bits	Description	
[31:19]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[18:16]	**BODVL**	**Brown-out Detector Threshold Voltage Selection (Write Protected)** The default value is set by flash controller user configuration register CBOV (CONFIG0 [23:21]). 000 = Brown-Out Detector threshold voltage is 1.6V. 001 = Brown-Out Detector threshold voltage is 1.8V. 010 = Brown-Out Detector threshold voltage is 2.0V. 011 = Brown-Out Detector threshold voltage is 2.2V. 100 = Brown-Out Detector threshold voltage is 2.4V. 101 = Brown-Out Detector threshold voltage is 2.6V. 110 = Brown-Out Detector threshold voltage is 2.8V. 111 = Brown-Out Detector threshold voltage is 3.0V. **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.
[15:11]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[10:8]	**BODDGSEL**	**Brown-out Detector Output De-glitch Time Select (Write Protected)** 000 = Without de-glitch function. 001 = 3 system clock (HCLK). 010 = 7 system clock (HCLK). 011 = 15 system clock (HCLK). 100 = 31 system clock (HCLK). 101 = 63 system clock (HCLK). 110 = 127 system clock (HCLK). 111 = 255 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 Protected)** 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. **Note:** 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. The detected voltage is higher than BODVL setting or BODEN is 0. 1 = Brown-out Detector output status is 1. The detected voltage is lower than BODVL setting. If the BODEN is 0, BOD function is disabled and this bit will be 0.
[5]	**BODLPM**	**Brown-out Detector Low Power Mode (Write Protected)** 0 = BOD operate in normal mode (default). 1 = BOD Low Power mode Enabled. **Note 1:** The low power mode can reduce the current to about 1/10 but slow the BOD response. **Note 2:** This bit is write protected. Refer to the SYS_REGLCTL register.
[4]	**BODIF**	**Brown-out Detector Interrupt Flag** 0 = Brown-out Detector has not detected a BOD event on V_{DD} down through or up through the voltage of BODVL setting. 1 = When Brown-out Detector detects that V_{DD} crosses BODLVL setting from either direction, this bit is set to 1 and the brown-out interrupt is requested if brown-out interrupt is enabled. **Note:** Write 1 to clear this bit to 0.
[3]	**BODRSTEN**	**Brown-out Reset Enable Bit (Write Protected)** 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. **Note 1:** 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 latch until BODEN is set to 0. BOD interrupt can be blocked by disabling the NVIC BOD interrupt or disabling BOD function (set BODEN low). **Note 2:** The reset value of SYS_BODCTL[3] is determined by user flash configuration. **Note 3:** This bit is write protected. Refer to the SYS_REGLCTL register.
[2:1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

Bits	Description	
[0]	**BODEN**	**Brown-out Detector Enable Bit (Write Protected)** The default value is set by flash controller user configuration register CBODEN (CONFIG0 [19]). 0 = Brown-out Detector function Disabled. 1 = Brown-out Detector function Enabled. **Note 1:** The reset value of SYS_BODCTL[0] is determined by user flash configuration. **Note 2:** Brown-out detector can only work when both BODEN(SYS_BODCTL[0]) and LVREN(SYS_BODCTL[7]) are set to 1. **Note 3:** When both BODEN(SYS_BODCTL[0]) and LVREN(SYS_BODCTL[7]) are set to 1, NVIC BOD interrupt must be enabled before entering power down mode. **Note 4:** This bit is write protected. Refer to the SYS_REGLCTL register.

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_0000

31	30	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. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	POROFF	**Power-on Reset Enable Bit (Write Protected)** When power is applied to device, the POR circuit generates a reset signal to reset the entire chip function. Noise on the power may cause the POR to become active again. User can disable internal POR circuit to avoid unpredictable noise to cause chip reset by writing 0x5AA5 to this field. The POR function will be active again when this field is set to another value or chip is reset by other reset source, including: nRESET, Watchdog, LVR reset, BOD reset, ICE reset command and the software-chip reset function. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

USB PHY Control Register (SYS_USBPHY)

Register	Offset	R/W	Description	Reset Value
SYS_USBPHY	SYS_BA+0x2C	R/W	USB PHY 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							USB_PHY_EN
7	6	5	4	3	2	1	0
Reserved							

Bits	Description	
[31:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8]	**USB_PHY_EN**	**USB PHY Enable (Write Protect)** This bit is used to enable/disable USB PHY function. 0 = USB PHY function Disabled (default). 1 = USB PHY function Enabled.
[7:0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

GPIOA Low Byte Multiple Function Control Register (SYS_GPA_MFPL)

Register	Offset	R/W	Description	Reset Value
SYS_GPA_MFPL	SYS_BA+0x30	R/W	GPIOA Low Byte Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PA7MFP				PA6MFP			
23	22	21	20	19	18	17	16
PA5MFP				PA4MFP			
15	14	13	12	11	10	9	8
PA3MFP				PA2MFP			
7	6	5	4	3	2	1	0
PA1MFP				PA0MFP			

Bits	Description	
[31:28]	PA7MFP	PA.7 Multi-function Pin Selection
[27:24]	PA6MFP	PA.6 Multi-function Pin Selection
[23:20]	PA5MFP	PA.5 Multi-function Pin Selection
[19:16]	PA4MFP	PA.4 Multi-function Pin Selection
[15:12]	PA3MFP	PA.3 Multi-function Pin Selection
[11:8]	PA2MFP	PA.2 Multi-function Pin Selection
[7:4]	PA1MFP	PA.1 Multi-function Pin Selection
[3:0]	PA0MFP	PA.0 Multi-function Pin Selection

GPIOA High Byte Multiple Function Control Register (SYS_GPA_MFPH)

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
PA9MFP				PA8MFP			

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]	PA9MFP	PA.9 Multi-function Pin Selection
[3:0]	PA8MFP	PA.8 Multi-function Pin Selection

GPIOB Low Byte Multiple Function Control Register (SYS_GPB_MFPL)

Register	Offset	R/W	Description	Reset Value
SYS_GPB_MFPL	SYS_BA+0x38	R/W	GPIOB Low Byte Multiple Function Control Register	0x0110_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)

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				Reserved			
15	14	13	12	11	10	9	8
Reserved							
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**
[19:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7:4]	**PB9MFP**	**PB.9 Multi-function Pin Selection**
[3:0]	**PB8MFP**	**PB.8 Multi-function Pin Selection**

GPIOC Low Byte Multiple Function Control Register (SYS_GPC_MFPL)

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
PC7MFP				PC6MFP			
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]	PC7MFP	PC.7 Multi-function Pin Selection
[27:24]	PC6MFP	PC.6 Multi-function Pin Selection
[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)

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
PC15MFP				PC14MFP			
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]	PC15MFP	PC.15 Multi-function Pin Selection
[27:24]	PC14MFP	PC.14 Multi-function Pin Selection
[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)

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
PD7MFP				PD6MFP			
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]	PD7MFP	PD.7 Multi-function Pin Selection
[27:24]	PD6MFP	PD.6 Multi-function Pin Selection
[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

GPIOD High Byte Multiple Function Control Register (SYS_GPD_MFPH)

Register	Offset	R/W	Description	Reset Value
SYS_GPD_MFPH	SYS_BA+0x4C	R/W	GPIOD High Byte Multiple Function Control Register	0x0000_0011

31	30	29	28	27	26	25	24
PD15MFP				PD14MFP			
23	22	21	20	19	18	17	16
PD13MFP				PD12MFP			
15	14	13	12	11	10	9	8
PD11MFP				PD10MFP			
7	6	5	4	3	2	1	0
PD9MFP				PD8MFP			

Bits	Description	
[31:28]	PD15MFP	PD.15 Multi-function Pin Selection
[27:24]	PD14MFP	PD.14 Multi-function Pin Selection
[23:20]	PD13MFP	PD.13 Multi-function Pin Selection
[19:16]	PD12MFP	PD.12 Multi-function Pin Selection
[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

System SRAM Parity Error Interrupt Enable Control Register (SYS_SRAM_INTCTL)

Register	Offset	R/W	Description	Reset Value
SYS_SRAM_INTCTL	SYS_BA+0xC0	R/W	System SRAM Interrupt Enable Control Register	0x0000_0000

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

Bits	Description	
[31:1]	Reserved	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	PERRIEN	**SRAM Parity Check Error Interrupt Enable Bit** 0 = SRAM parity check error interrupt Disabled. 1 = SRAM parity check error interrupt Enabled.

System SRAM Parity Check Status Register (SYS_SRAM_STATUS)

Register	Offset	R/W	Description	Reset Value
SYS_SRAM_STATUS	SYS_BA+0xC4	R/W	System SRAM Parity Error 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							PERRIF

Bits	Description	
[31:1]	Reserved	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	PERRIF	**SRAM Parity Check Error Flag** This bit indicates the System SRAM parity error occurred. Write 1 to clear this bit to 0. 0 = No System SRAM parity error. 1 = System SRAM parity error occur.

System SRAM Parity Error Address Register (SYS_SRAM_ERRADDR)

Register	Offset	R/W	Description	Reset Value
SYS_SRAM_ERRADDR	SYS_BA+0xC8	R	System SRAM Parity Check Error Address Register	0x0000_0000

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

Bits	Description	
[31:0]	ERRADDR	**System SRAM Parity Error Address** This register shows system SRAM parity error byte address.

HIRC Trim Control Register (SYS_IRCTCTL)

Register	Offset	R/W	Description	Reset Value
SYS_IRCTCTL	SYS_BA+0xF0	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. Any values read should be ignored. When writing to this field always write with reset value.
[10]	**REFCKSEL**	**Reference Clock Selection** 0 = HIRC trim reference clock is from LXT (32.768 kHz). 1 = HIRC trim reference clock is from USB SOF (Start-Of-Frame) packet.
[9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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.
[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 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 reference clocks. 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. Any values read should be ignored. When writing to this field always write with reset value.
[1:0]	**FREQSEL**	**Trim Frequency Selection** This field indicates the target frequency of 48 MHz and 49.152 MHz internal high speed RC oscillator (HIRC) auto trim. During auto trim operation, if clock error detected with CESTOPEN is set to 1 or trim retry limitation count reached, this field will be cleared to 00 automatically. 01 = Enable HIRC auto trim function and trim HIRC to 48 MHz. 11 = Enable HIRC auto trim function and trim HIRC to 49.152 MHz. Others = Disable HIRC auto trim function.

HIRC Trim Interrupt Enable Register (SYS_IRCTIEN)

Register	Offset	R/W	Description	Reset Value
SYS_IRCTIEN	SYS_BA+0xF4	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. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**CLKEIEN**	**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_IRCTISTS[2]) is set during auto trim operation, an interrupt will be triggered to notify the clock frequency is inaccuracy. 0 = Disable CLKERRIF(SYS_IRCTISTS[2]) status to trigger an interrupt to CPU. 1 = Enable CLKERRIF(SYS_IRCTISTS[2]) status to trigger an interrupt to CPU.
[1]	**TFAILIEN**	**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_IRCTCTL[1:0]). If this bit is high and TFAILIF(SYS_IRCTISTS[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_IRCTISTS[1]) status to trigger an interrupt to CPU. 1 = Enable TFAILIF(SYS_IRCTISTS[1]) status to trigger an interrupt to CPU.
[0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

HIRC Trim Interrupt Status Register (SYS_IRCTISTS)

Register	Offset	R/W	Description	Reset Value
SYS_IRCTISTS	SYS_BA+0xF8	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. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**CLKERRIF**	**Clock Error Interrupt Status** When the frequency of 32.768 kHz external low speed crystal oscillator (LXT) or 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_IRCTIEN[2]) is high, an interrupt will be triggered to notify the clock frequency is inaccuracy. Write 1 to clear this to 0. 0 = Clock frequency is accuracy. 1 = Clock frequency is inaccuracy.
[1]	**TFAILIF**	**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_IRCTIEN[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 = Trim value update limitation count does not reach. 1 = Trim value update limitation count reached and HIRC 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. 0 = The internal high-speed oscillator frequency doesn't lock at 48 MHz or 49.152 MHz yet. 1 = The internal high-speed oscillator frequency locked at 48 MHz or 49.152 MHz.

Register Lock Control Register (SYS_REGLCTL)

This register is written to disable/enable register protection and read for the REGLCTL status.
Some of the system control registers are protected to avoid inadvertent write that may disturb the chip operation. These system control registers are protected after power-on reset until the user disables this register protection. For user to program these protected registers, a register protection disable sequence needs to be followed. The register protection disable sequence is writing the data “59h”, “16h” “88h” to the register SYS_REGLCTL address at 0x4000_0100. Any different data value, different sequence or any other write to other address during these three data writes will abort the whole sequence.
After the protection is disabled, user can check the protection disable bit at address 0x4000_0100 bit0, 1 is protection disable, and 0 is protection enable. Then user can update the target protected register value. The register protection can be re-enabled by writing any data to the address 0x4000_0100.

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. Any values read should be ignored. When writing to this field always write with reset value.
[7:0]	**REGLCTL**	**Register Lock Control Code (Write Only)** Some registers have a write-protection function. To write to these registers, this write protection must be by writing the sequence value “59h”, “16h”, “88h” to this address. After this sequence is completed, the REGLCTL bit will be set to 1 and write-protected registers can be write accessed. **Register Lock Control Disable Index (Read Only)** 0 = Write-protection Enabled for write protected registers. Any write to the protected registers is ignored. 1 = Write-protection Disabled for write protected registers. The Protected registers are: **SYS_IPRST0**: address 0x4000_0008 **SYS_BODCTL**: address 0x4000_0018 **SYS_PORCTL**: address 0x4000_0024 **SYS_USBPHY:** address 0x4000_002C **CLK_PWRCTL**: address 0x4000_0200 (bit[6] is not protected for power-down wake-up interrupt clear) **SYS_RCADJ:** address 0x4000_0110

		CLK_APBCLK0 [0]: address 0x4000_0208 (bit[0] is watchdog clock enable) **CLK_CLKSEL0**: address 0x4000_0210 (for HCLK and CPU STCLK clock source select) **CLK_CLKSEL1 [1:0]**: address 0x4000_0214 (for watchdog clock source select) **CLK_CLKSEL1 [31:30]**: address 0x4000_0214 (for window watchdog clock source select) **CLK_CLKDSTS:** address 0x4000_0274 **NMIEN**: address 0x4000_0300 **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 0x4004_0000 **AHBMCTL:** address 0x4000_0400 **CLK_PLLCTL:** address 0x4000_0240 **PWM_CTL0:** address 0x4005_8000 **PWM_DTCTL0_1:** address 0x4005_8070 **PWM_DTCTL2_3:** address 0x4005_8074 **PWM_DTCTL4_5:** address 0x4005_8078 **PWM_BRKCTL0_1:** address 0x4005_80C8 **PWM_BRKCTL2_3:** address 0x4005_80CC **PWM_BRKCTL4_5:** address 0x4005_80D0 **PWM_INTEN1:** address 0x4005_80E4 **PWM_INTSTS1:** address 0x4005_80EC

HIRC Trim Value Register (SYS_RCADJ)

Register	Offset	R/W	Description	Reset Value
SYS_RCADJ	SYS_BA+0x110	R/W	HIRC Trim Value 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						RCADJ	
7	6	5	4	3	2	1	0
RCADJ							

Bits	Description	
[31:10]	Reserved	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[9:0]	RCADJ	**HIRC Trim Value (Write Protect)** This bit is the protected bit, which means programming it needs to write “59h”, “16h”, “88h” to address 0x4000_0100 to disable register protection. Refer to the register SYS_REGLCTL at address SYS_BA+0x100. This field reflects the HIRC trim value. Software can update HIRC trim value by writing this field. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

6.2.12 System Timer (SysTick)

The Cortex®-M4 integrates a 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_VAL) to zero, and reload (wrap) to the value in the SysTick Reload Value Register (SYST_LOAD) on the next clock cycle, and then decrement on subsequent clocks. When the counter decrements to zero, the COUNTFLAG status bit is set. A read or write on Current Value Register clears the COUNTFLAG bit to 0.

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

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

For more detailed information, please refer to the "*ARM® Cortex™-M4 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	0x0000_0000
SYST_VAL	SCS_BA+0x18	R/W	SysTick Current Value Register	0x0000_0000

Note:

1. Any register not listed here is reserved and must not be written. The result of a read operation on these bits is undefined.
2. The reserved register fields that listed in register description must be written to their reset value. Writing reserved fields with other than reset values may produce undefined results.

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. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.
[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	0x0000_0000

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. Any values read should be ignored. When writing to this field always write with reset value.
[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	0x0000_0000

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. Any values read should be ignored. When writing to this field always write with reset value.
[23:0]	**CURRENT**	**System Tick Current Value** Current counter value. This is the value of the counter at the time it is sampled. The counter does not provide read-modify-write protection. The register is write-clear. A software write of any value will clear the register to 0.

6.2.13 Nested Vectored Interrupt Controller (NVIC)

The NVIC and the processor core interface are closely coupled to enable low latency interrupt processing and efficient processing of late arriving interrupts. The NVIC maintains knowledge of the stacked, or nested, interrupts to enable tail-chaining of interrupts. You can only fully access the NVIC from privileged mode, but you can cause interrupts to enter a pending state in user mode if you enable the Configuration and Control Register. Any other user mode access causes a bus fault. You can access all NVIC registers using byte, halfword, and word accesses unless otherwise stated. NVIC registers are located within the SCS (System Control Space). All NVIC registers and system debug registers are little-endian regardless of the endianness state of the processor.

The NVIC supports:

- An implementation-defined number of interrupts, in the range 1-240 interrupts.
- A programmable priority level of 0-16 for each interrupt; a higher level corresponds to a lower priority, so level 0 is the highest interrupt priority.
- Level and pulse detection of interrupt signals.
- Dynamic reprioritization of interrupts.
- Grouping of priority values into group priority and subpriority fields.
- Interrupt tail-chaining.
- An external Non Maskable Interrupt (NMI)
- WIC with Ultra-low Power Sleep mode support

The processor automatically stacks its state on exception entry and unstacks this state on exception exit, with no instruction overhead. This provides low latency exception handling.

6.2.13.1 Exception Model and System Interrupt Map

The Table 6.2.13-1 lists the exception model supported by NPCA121 Series. Software can set 16 levels of priority on some of these exceptions as well as on all interrupts. The highest user-configurable priority is denoted as “0x00” and the lowest priority is denoted as “0xF0” (The 4-LSB always 0). The default priority of all the user-configurable interrupts is “0x00”. Note that priority “0” is treated as the fourth priority on the system, after three system exceptions “Reset”, “NMI” and “Hard Fault”.

When any interrupts is accepted, the processor will automatically fetch the starting address of the interrupt service routine (ISR) from a vector table in memory. On system reset, the vector table is fixed at address 0x00000000. Privileged software can write to the VTOR to relocate the vector table start address to a different memory location, in the range 0x00000080 to 0x3FFFFF80,

The vector table contains the initialization value for the stack pointer on reset, and the entry point addresses for all exception handlers. The vector number on previous page defines the order of entries in the vector table associated with exception handler entry as illustrated in previous section.

Exception Type	Vector Number	Vector Address	Priority
Reset	1	0x00000004	-3
NMI	2	0x00000008	-2
Hard Fault	3	0x0000000C	-1
Memory Manager Fault	4	0x00000010	Configurable
Bus Fault	5	0x00000014	Configurable
Usage Fault	6	0x00000018	Configurable
Reserved	7 ~ 10		Reserved
SVCall	11	0x0000002C	Configurable

Debug Monitor	12	0x00000030	Configurable
Reserved	13		Reserved
PendSV	14	0x00000038	Configurable
SysTick	15	0x0000003C	Configurable
Interrupt (IRQ0 ~ IRQ)	16 ~ 111	0x00000000 + (Vector Number)*4	Configurable

Table 6.2.13-1 Exception Model

Vector Number	Interrupt Number (Bit In Interrupt Registers)	Interrupt Name	Interrupt Description
0 ~ 15	-	-	System exceptions
16	0	BODOUT	Brown-Out low voltage detected interrupt
17	1	IRC_INT	IRC TRIM interrupt
18	2	PWRWU_INT	Clock controller interrupt for chip wake-up from power-down state
19	3	SRAM_PERR	SRAM parity check error interrupt
20	4	CLKFAIL	Clock fail detected interrupt
21	5	Reserved	Reserved
22	6	RTC_INT	Real time clock interrupt
23	7	Reserved	Reserved
24	8	WDT_INT	Watchdog Timer interrupt
25	9	WWDT_INT	Window Watchdog Timer interrupt
26	10	EINT0	External interrupt
27	11	EINT1	External interrupt
28	12	EINT2	External interrupt
29	13	EINT3	External interrupt
30	14	EINT4	External interrupt
31	15	EINT5	External interrupt
32	16	GPA_INT	External interrupt from PA[15:0] pin
33	17	GPB_INT	External interrupt from PB[14:12/9:0] pin
34	18	GPC_INT	External interrupt from PC[15:0] pin
35	19	GPD_INT	External interrupt from PD[15:0] pin
36 ~ 37	20 ~ 21	Reserved	Reserved
38	22	SPI0_INT	SPI0 interrupt
39	23	SPI1_INT	SPI1 interrupt
40	24	BRAKE0_INT	PWM0 brake interrupt

41	25	PWM0_P0_INT	PWM0 pair 0 interrupt
42	26	PWM0_P1_INT	PWM0 pair 1 interrupt
43	27	PWM0_P2_INT	PWM0 pair 2 interrupt
44 ~ 47	28 ~ 31	Reserved	Reserved
48	32	TMR0_INT	Timer 0 interrupt
49	33	TMR1_INT	Timer 1 interrupt
50	34	TMR2_INT	Timer 2 interrupt
51	35	TMR3_INT	Timer 3 interrupt
52	36	UART0_INT	UART0 interrupt
53	37	Reserved	Reserved
54	38	I2C0_INT	I2C0 interrupt
55	39	I2C1_INT	I2C1 interrupt
56	40	PDMA_INT	PDMA interrupt
57	41	Reserved	Reserved
58	42	EADC0_INT	EADC interrupt source 0
59	43	EADC1_INT	EADC interrupt source 1
60	44	Reserved	Reserved
61	45	Reserved	Reserved
62	46	EADC2_INT	EADC interrupt source 2
63	47	EADC3_INT	EADC interrupt source 3
64 ~ 66	48~ 50	Reserved	Reserved
67	51	SPI2_INT	SPI2 interrupt
68	52	DMIC_INT	DMIC interrupt
69	53	USBD_INT	USB device interrupt
70 ~ 71	54 ~ 55	Reserved	Reserved
72	56	VAD_INT	VAD interrupt
73 ~ 77	57 ~ 61	Reserved	Reserved
78	62	DPWM_INT	DPWM interrupt
79 ~ 83	63 ~ 67	Reserved	Reserved
84	68	I2S0_INT	I^2S0 interrupt
85 ~ 111	69 ~ 95	Reserved	Reserved

Table 6.2.13-2 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: **NVIC_BA = 0xE000_E100**				
NVIC_ISER0	0xE000E100	R/W	IRQ0 ~ IRQ31 Set-Enable Control Register	0x0000_0000
NVIC_ISER1	0xE000E104	R/W	IRQ32 ~ IRQ63 Set-Enable Control Register	0x0000_0000
NVIC_ISER2	0xE000E108	R/W	IRQ64 ~ IRQ95 Set-Enable Control Register	0x0000_0000
NVIC_ICER0	0xE000E180	R/W	IRQ0 ~ IRQ31 Clear-Enable Control Register	0x0000_0000
NVIC_ICER1	0xE000E184	R/W	IRQ32 ~ IRQ63 Clear-Enable Control Register	0x0000_0000
NVIC_ICER2	0xE000E188	R/W	IRQ64 ~ IRQ95 Clear-Enable Control Register	0x0000_0000
NVIC_ISPR0	0xE000E200	R/W	IRQ0 ~ IRQ31 Set-Pending Control Register	0x0000_0000
NVIC_ISPR1	0xE000E204	R/W	IRQ32 ~ IRQ63 Set-Pending Control Register	0x0000_0000
NVIC_ISPR2	0xE000E208	R/W	IRQ64 ~ IRQ95 Set-Pending Control Register	0x0000_0000
NVIC_ICPR0	0xE000E280	R/W	IRQ0 ~ IRQ31 Clear-Pending Control Register	0x0000_0000
NVIC_ICPR1	0xE000E284	R/W	IRQ32 ~ IRQ63 Clear-Pending Control Register	0x0000_0000
NVIC_ICPR2	0xE000E288	R/W	IRQ64 ~ IRQ95 Clear-Pending Control Register	0x0000_0000
NVIC_IABR0	0xE000E300	R/W	IRQ0 ~ IRQ31 Active Bit Register	0x0000_0000
NVIC_IABR1	0xE000E304	R/W	IRQ32 ~ IRQ63 Active Bit Register	0x0000_0000
NVIC_IABR2	0xE000E308	R/W	IRQ64 ~ IRQ95 Active Bit Register	0x0000_0000
NVIC_IPRn **n=0,1..23**	0xE000E400 +0x4*n	R/W	IRQ0 ~ IRQ95 Priority Control Register	0x0000_0000
STIR	0xE000F000	W	Software Trigger Interrupt Registers	0x0000_0000

Note:

1. Any register not listed here is reserved and must not be written. The result of a read operation on these bits is undefined.
2. The reserved register fields that listed in register description must be written to their reset value. Writing reserved fields with other than reset values may produce undefined results.

IRQ0 ~ IRQ31 Set-Enable Control Register (NVIC_ISER0)

Register	Offset	R/W	Description	Reset Value
NVIC_ISER0	0xE000E100	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 Set Enable Bit** The NVIC_ISER0-NVIC_ISER2 registers enable interrupts, and show which interrupts are enabled. Write Operation: 0 = No effect. 1 = Interrupt Enabled. Read Operation: 0 = Interrupt Disabled. 1 = Interrupt Enabled.

IRQ32 ~ IRQ63 Set-Enable Control Register (NVIC_ISER1)

Register	Offset	R/W	Description	Reset Value
NVIC_ISER1	0xE000E104	R/W	IRQ32 ~ IRQ63 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 Set Enable Bit** The NVIC_ISER0-NVIC_ISER2 registers enable interrupts, and show which interrupts are enabled Write Operation: 0 = No effect. 1 = Interrupt Enabled. Read Operation: 0 = Interrupt Disabled. 1 = Interrupt Enabled.

IRQ64 ~ IRQ95 Set-Enable Control Register (NVIC_ISER2)

Register	Offset	R/W	Description	Reset Value
NVIC_ISER2	0xE000E108	R/W	IRQ64 ~ IRQ95 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 Set Enable Bit** The NVIC_ISER0-NVIC_ISER2 registers enable interrupts, and show which interrupts are enabled Write Operation: 0 = No effect. 1 = Interrupt Enabled. Read Operation: 0 = Interrupt Disabled. 1 = Interrupt Enabled.

IRQ0 ~ IRQ31 Clear-Enable Control Register (NVIC_ICER0)

Register	Offset	R/W	Description	Reset Value
NVIC_ICER0	0xE000E180	R/W	IRQ0 ~ IRQ31 Clear-Enable Control Register	0x0000_0000

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

Bits	Description	
[31:0]	**CALENA**	**Interrupt Clear Enable Bit** The NVIC_ICER0-NVIC_ICER2 registers disable interrupts, and show which interrupts are enabled. Write Operation: 0 = No effect. 1 = Interrupt Disabled. Read Operation: 0 = Interrupt Disabled. 1 = Interrupt Enabled.

IRQ32 ~ IRQ63 Clear-Enable Control Register (NVIC_ICER1)

Register	Offset	R/W	Description	Reset Value
NVIC_ICER1	0xE000E184	R/W	IRQ32 ~ IRQ63 Clear-Enable Control Register	0x0000_0000

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

Bits	Description	
[31:0]	**CALENA**	**Interrupt Clear Enable Bit** The NVIC_ICER0-NVIC_ICER2 registers disable interrupts, and show which interrupts are enabled. Write Operation: 0 = No effect. 1 = Interrupt Disabled. Read Operation: 0 = Interrupt Disabled. 1 = Interrupt Enabled.

IRQ64 ~ IRQ95 Clear-Enable Control Register (NVIC_ICER2)

Register	Offset	R/W	Description	Reset Value
NVIC_ICER2	0xE000E188	R/W	IRQ64 ~ IRQ95 Clear-Enable Control Register	0x0000_0000

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

Bits	Description	
[31:0]	CALENA	**Interrupt Clear Enable Bit** The NVIC_ICER0-NVIC_ICER2 registers disable interrupts, and show which interrupts are enabled. Write Operation: 0 = No effect. 1 = Interrupt Disabled. Read Operation: 0 = Interrupt Disabled. 1 = Interrupt Enabled.

IRQ0 ~ IRQ31 Set-Pending Control Register (NVIC_ISPR0)

Register	Offset	R/W	Description	Reset Value
NVIC_ISPR0	0xE000E200	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	**Interrupt Set-pending** The NVIC_ISPR0-NVIC_ISPR2 registers force interrupts into the pending state, and show which interrupts are pending Write Operation: 0 = No effect. 1 = Changes interrupt state to pending. Read Operation: 0 = Interrupt is not pending. 1 = Interrupt is pending.

IRQ32 ~ IRQ63 Set-Pending Control Register (NVIC_ISPR1)

Register	Offset	R/W	Description	Reset Value
NVIC_ISPR1	0xE000E204	R/W	IRQ32 ~ IRQ63 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	**Interrupt Set-pending** The NVIC_ISPR0-NVIC_ISPR2 registers force interrupts into the pending state, and show which interrupts are pending Write Operation: 0 = No effect. 1 = Changes interrupt state to pending. Read Operation: 0 = Interrupt is not pending. 1 = Interrupt is pending.

IRQ64 ~ IRQ95 Set-Pending Control Register (NVIC_ISPR2)

Register	Offset	R/W	Description	Reset Value
NVIC_ISPR2	0xE000E208	R/W	IRQ64 ~ IRQ95 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	**Interrupt Set-pending** The NVIC_ISPR0-NVIC_ISPR2 registers force interrupts into the pending state, and show which interrupts are pending Write Operation: 0 = No effect. 1 = Changes interrupt state to pending. Read Operation: 0 = Interrupt is not pending. 1 = Interrupt is pending.

IRQ0 ~ IRQ31 Clear-Pending Control Register (NVIC_ICPR0)

Register	Offset	R/W	Description	Reset Value
NVIC_ICPR0	0xE000E280	R/W	IRQ0 ~ IRQ31 Clear-Pending Control Register	0x0000_0000

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

Bits	Description	
[31:0]	CALPEND	**Interrupt Clear-pending** The NVIC_ICPR0-NVIC_ICPR2 registers remove the pending state from interrupts, and show which interrupts are pending Write Operation: 0 = No effect. 1 = Removes pending state an interrupt. Read Operation: 0 = Interrupt is not pending. 1 = Interrupt is pending.

IRQ32 ~ IRQ63 Clear-Pending Control Register (NVIC_ICPR1)

Register	Offset	R/W	Description	Reset Value
NVIC_ICPR1	0xE000E284	R/W	IRQ32 ~ IRQ63 Clear-Pending Control Register	0x0000_0000

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

Bits	Description	
[31:0]	CALPEND	**Interrupt Clear-pending** The NVIC_ICPR0-NVIC_ICPR2 registers remove the pending state from interrupts, and show which interrupts are pending Write Operation: 0 = No effect. 1 = Removes pending state an interrupt. Read Operation: 0 = Interrupt is not pending. 1 = Interrupt is pending.

IRQ64 ~ IRQ95 Clear-Pending Control Register (NVIC_ICPR2)

Register	Offset	R/W	Description	Reset Value
NVIC_ICPR2	0xE000E288	R/W	IRQ64 ~ IRQ95 Clear-Pending Control Register	0x0000_0000

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

Bits	Description	
[31:0]	**CALPEND**	**Interrupt Clear-pending** The NVIC_ICPR0-NVIC_ICPR2 registers remove the pending state from interrupts, and show which interrupts are pending Write Operation: 0 = No effect. 1 = Removes pending state an interrupt. Read Operation: 0 = Interrupt is not pending. 1 = Interrupt is pending.

IRQ0 ~ IRQ31 Active Bit Register (NVIC_IABR0)

Register	Offset	R/W	Description	Reset Value
NVIC_IABR0	0xE000E300	R/W	IRQ0 ~ IRQ31 Active Bit Register	0x0000_0000

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

Bits	Description	
[31:0]	**ACTIVE**	**Interrupt Active Flags** The NVIC_IABR0-NVIC_IABR2 registers indicate which interrupts are active. 0 = interrupt not active. 1 = interrupt active.

IRQ32 ~ IRQ63 Active Bit Register (NVIC_IABR1)

Register	Offset	R/W	Description	Reset Value
NVIC_IABR1	0xE000E304	R/W	IRQ32 ~ IRQ63 Active Bit Register	0x0000_0000

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

Bits	Description	
[31:0]	ACTIVE	**Interrupt Active Flags** The NVIC_IABR0-NVIC_IABR2 registers indicate which interrupts are active. 0 = interrupt not active. 1 = interrupt active.

IRQ64 ~ IRQ95 Active Bit Register (NVIC_IABR2)

Register	Offset	R/W	Description	Reset Value
NVIC_IABR2	0xE000E308	R/W	IRQ64 ~ IRQ95 Active Bit Register	0x0000_0000

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

Bits	Description	
[31:0]	ACTIVE	**Interrupt Active Flags** The NVIC_IABR0-NVIC_IABR2 registers indicate which interrupts are active. 0 = interrupt not active. 1 = interrupt active.

IRQ0 ~ IRQ95 Interrupt Priority Register (NVIC_IPRn)

Register	Offset	R/W	Description	Reset Value
NVIC_IPRn n=0,1..23	0xE000E400 +0x4*n	R/W	IRQ0 ~ IRQ95 Priority Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PRI_4n_3				Reserved			
23	22	21	20	19	18	17	16
PRI_4n_2				Reserved			
15	14	13	12	11	10	9	8
PRI_4n_1				Reserved			
7	6	5	4	3	2	1	0
PRI_4n_0				Reserved			

Bits	Description	
[31:28]	**PRI_4n_3**	**Priority of IRQ_4n+3** "0" denotes the highest priority and "15" denotes the lowest priority
[27:24]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[23:20]	**PRI_4n_2**	**Priority of IRQ_4n+2** "0" denotes the highest priority and "15" denotes the lowest priority
[19:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:12]	**PRI_4n_1**	**Priority of IRQ_4n+1** "0" denotes the highest priority and "15" denotes the lowest priority
[11:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7:4]	**PRI_4n_0**	**Priority of IRQ_4n+0** "0" denotes the highest priority and "15" denotes the lowest priority
[3:0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

Software Trigger Interrupt Register (STIR)

Register	Offset	R/W	Description	Reset Value
STIR	0xE000F000	W	Software Trigger Interrupt Registers	0x0000_0000

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

Bits	Description	
[31:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8:0]	**INTID**	**Interrupt ID** Write to the STIR To Generate An Interrupt from Software When the USERSETMPEND bit in the SCR is set to 1, unprivileged software can access the STIR Interrupt ID of the interrupt to trigger, in the range 0-63. For example, a value of 0x03 specifies interrupt IRQ3.

6.2.13.4 NMI Control Registers

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

Register	Offset	R/W	Description	Reset Value
NMI Base Address: **NMI_BA = 0x4000_0300**				
NMIEN	NMI_BA+0x00	R/W	NMI Source Interrupt Enable Register	0x0000_0000
NMISTS	NMI_BA+0x04	R	NMI source interrupt Status Register	0x0000_0000

NMI Source Interrupt Enable Register (NMIEN)

Register	Offset	R/W	Description	Reset Value
NMIEN	NMI_BA+0x00	R/W	NMI Source 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	**UART0_INT**	**EINT5**	**EINT4**	**EINT3**	**EINT2**	**EINT1**	**EINT0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**RTC_INT**	**Reserved**	**CLKFAIL**	**SRAM_PERR**	**PWRWU_INT**	**IRC_INT**	**BODOUT**

Bits	Description	
[31:15]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[14]	**UART0_INT**	**UART0 NMI Source Enable (Write Protected)** 0 = UART0 NMI source Disabled. 1 = UART0 NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[13]	**EINT5**	**External Interrupt 5 NMI Source Enable (Write Protect)** 0 = External interrupt 5 NMI source Disabled. 1 = External interrupt 5 NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[12]	**EINT4**	**External Interrupt 4 NMI Source Enable (Write Protected)** 0 = External interrupt 4 NMI source Disabled. 1 = External interrupt 4 NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[11]	**EINT3**	**External Interrupt 3 NMI Source Enable (Write Protected)** 0 = External interrupt 3 NMI source Disabled. 1 = External interrupt 3 NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[10]	**EINT2**	**External Interrupt 2 NMI Source Enable (Write Protected)** 0 = External interrupt 2 NMI source Disabled. 1 = External interrupt 2 NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[9]	**EINT1**	**External Interrupt 1 NMI Source Enable (Write Protected)** 0 = External interrupt 1 NMI source Disabled. 1 = External interrupt 1 NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

[8]	**EINT0**	**External Interrupt 0 NMI Source Enable (Write Protected)** 0 = External interrupt 0 NMI source Disabled. 1 = External interrupt 0 NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[7]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[6]	**RTC_INT**	**RTC NMI Source Enable (Write Protected)** 0 = RTC NMI source Disabled. 1 = RTC NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[5]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[4]	**CLKFAIL**	**Clock Fail Detected NMI Source Enable (Write Protected)** 0 = Clock fail detected interrupt NMI source Disabled. 1 = Clock fail detected interrupt NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[3]	**SRAM_PERR**	**SRAM Parity Check Error NMI Source Enable (Write Protected)** 0 = SRAM parity check error NMI source Disabled. 1 = SRAM parity check error NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[2]	**PWRWU_INT**	**Power-down Mode Wake-up NMI Source Enable (Write Protected)** 0 = Power-down mode wake-up NMI source Disabled. 1 = Power-down mode wake-up NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[1]	**IRC_INT**	**IRC TRIM NMI Source Enable (Write Protected)** 0 = IRC TRIM NMI source Disabled. 1 = IRC TRIM NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[0]	**BODOUT**	**BOD NMI Source Enable (Write Protected)** 0 = BOD NMI source Disabled. 1 = BOD NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

NMI Source Interrupt Status Register (NMISTS)

Register	Offset	R/W	Description	Reset Value
NMISTS	NMI_BA+0x04	R	NMI source 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	UART0_INT	EINT5	EINT4	EINT3	EINT2	EINT1	EINT0
7	6	5	4	3	2	1	0
Reserved	RTC_INT	Reserved	CLKFAIL	SRAM_PERR	PWRWU_INT	IRC_INT	BODOUT

Bits	Description	
[31:15]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[14]	**UART0_INT**	**UART0 Interrupt Flag (Read Only)** 0 = UART0 interrupt is deasserted. 1 = UART0 interrupt is asserted.
[13]	**EINT5**	**External Interrupt 5 Interrupt Flag (Read Only)** 0 = External Interrupt 5 interrupt is deasserted. 1 = External Interrupt 5 interrupt is asserted.
[12]	**EINT4**	**External Interrupt 4 Interrupt Flag (Read Only)** 0 = External Interrupt 4 interrupt is deasserted. 1 = External Interrupt 4 interrupt is asserted.
[11]	**EINT3**	**External Interrupt 3 Interrupt Flag (Read Only)** 0 = External Interrupt 3 interrupt is deasserted. 1 = External Interrupt 3 interrupt is asserted.
[10]	**EINT2**	**External Interrupt 2 Interrupt Flag (Read Only)** 0 = External Interrupt 2 interrupt is deasserted. 1 = External Interrupt 2 interrupt is asserted.
[9]	**EINT1**	**External Interrupt 1 Interrupt Flag (Read Only)** 0 = External Interrupt 1 interrupt is deasserted. 1 = External Interrupt 1 interrupt is asserted.
[8]	**EINT0**	**External Interrupt 0 Interrupt Flag (Read Only)** 0 = External Interrupt 0 interrupt is deasserted. 1 = External Interrupt 0 interrupt is asserted.
[7]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[6]	**RTC_INT**	**RTC Interrupt Flag (Read Only)** 0 = RTC interrupt is deasserted. 1 = RTC interrupt is asserted.
[5]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[4]	**CLKFAIL**	**Clock Fail Detected Interrupt Flag (Read Only)** 0 = Clock fail detected interrupt is deasserted. 1 = Clock fail detected interrupt is asserted.
[3]	**SRAM_PERR**	**SRAM Parity Check Error Interrupt Flag (Read Only)** 0 = SRAM parity check error interrupt is deasserted. 1 = SRAM parity check error interrupt is asserted.
[2]	**PWRWU_INT**	**Power-down Mode Wake-up Interrupt Flag (Read Only)** 0 = Power-down mode wake-up interrupt is deasserted. 1 = Power-down mode wake-up interrupt is asserted.
[1]	**IRC_INT**	**IRC TRIM Interrupt Flag (Read Only)** 0 = HIRC TRIM interrupt is deasserted. 1 = HIRC TRIM interrupt is asserted.
[0]	**BODOUT**	**BOD Interrupt Flag (Read Only)** 0 = BOD interrupt is deasserted. 1 = BOD interrupt is asserted.

6.2.13.5 AHB Bus Matrix Priority Control Register

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

Register	Offset	R/W	Description	Reset Value
AHB Base Address: **AHB_BA = 0x4000_0400**				
AHBMCTL	0x40000400	R/W	AHB Bus Matrix Priority Control Register	0x0000_0001

AHB Bus Matrix Priority Control Register (AHBMCTL)

Register	Offset	R/W	Description	Reset Value
AHBMCTL	0x40000400	R/W	AHB Bus Matrix Priority 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							**INTACTEN**

Bits	Description	
[31:1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**INTACTEN**	**Highest AHB Bus Priority of Cortex M4 Core Enable Bit (Write Protected)** Enable Cortex®-M4 Core With Highest AHB Bus Priority In AHB Bus Matrix 0 = Round-robin mode. 1 = Cortex®-M4 CPU with highest bus priority when interrupt occur. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

6.2.14 System Control Register

The Cortex®-M4 status and operation mode control are managed by System Control Registers. These registers also manage CPUID, Cortex®-M4 interrupt priority and Cortex®-M4 power management.
For more detailed information, please refer to the "*ARM® Cortex™-M4 Technical Reference Manual*" and "*ARM® v6-M Architecture Reference Manual*".

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

Register	Offset	R/W	Description	Reset Value
SCR Base Address: **SCS_BA = 0xE000_E000**				
ICSR	SCS_BA+0xD04	R/W	Interrupt Control and State Register	0x0000_0000
AIRCR	SCS_BA+0xD0C	R/W	Application Interrupt and Reset Control Register	0xFA05_0000
SCR	SCS_BA+0xD10	R/W	System Control Register	0x0000_0000
SHPR1	SCS_BA+0xD18	R/W	System Handler Priority Register 1	0x0000_0000
SHPR2	SCS_BA+0xD1C	R/W	System Handler Priority Register 2	0x0000_0000
SHPR3	SCS_BA+0xD20	R/W	System Handler Priority Register 3	0x0000_0000

Note:

1. Any register not listed here is reserved and must not be written. The result of a read operation on these bits is undefined.
2. The reserved register fields that listed in register description must be written to their reset value. Writing reserved fields with other than reset values may produce undefined results.

Interrupt Control State Register (ICSR)

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

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

Bits	Description	
[31]	NMIPENDSET	**NMI Set-pending Bit** Write Operation: 0 = No effect. 1 = Changes NMI exception state to pending. Read Operation: 0 = NMI exception is not pending. 1 = NMI exception is pending. **Note:** Because NMI is the highest-priority exception, normally the processor enters the NMI exception handler as soon as it detects a write of 1 to this bit. Entering the handler then clears this bit to 0. This means a read of this bit by the NMI exception handler returns 1 only if the NMI signal is reasserted while the processor is executing that handler.
[30:29]	Reserved	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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]	PENDSVRTC_CAL	**PendSV Clear-pending Bit** Write Operation: 0 = No effect. 1 = Removes the pending state from the PendSV exception. **Note:** This is a write only bit. To clear the PENDSV bit, you must “write 0 to PENDSVSET and write 1 to PENDSVRTC_CAL” 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]	**PENDSTRTC_CAL**	**SysTick Exception Clear-pending Bit** Write Operation: 0 = No effect. 1 = Removes the pending state from the SysTick exception. **Note:** This is a write only bit. To clear the PENDST bit, you must "write 0 to PENDSTSET and write 1 to PENDSTRTC_CAL" at the same time.
[24]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[23]	**ISRPREEMPT**	**Interrupt Preempt Bit (Read Only)** If set, a pending exception will be serviced on exit from the debug halt state.
[22]	**ISRPENDING**	**Interrupt Pending Flag, Excluding NMI and Faults (Read Only)** 0 = Interrupt not pending. 1 = Interrupt pending.
[21:18]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[17:12]	**VECTPENDING**	**Number of the Highest Pended Exception** Indicate the Exception Number of the Highest Priority Pending Enabled Exception 0 = no pending exceptions. Nonzero = the exception number of the highest priority pending enabled exception. The value indicated by this field includes the effect of the BASEPRI and FAULTMASK registers, but not any effect of the PRIMASK register.
[11]	**RETTOBASE**	**Preempted Active Exceptions Indicator** Indicate whether There are Preempted Active Exceptions 0 = there are preempted active exceptions to execute. 1 = there are no active exceptions, or the currently-executing exception is the only active exception.
[10:7]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[6:0]	**VECTACTIVE**	**Number of the Current Active Exception** 0 = Thread mode. Non-zero = The exception number of the currently active exception.

Application Interrupt and Reset Control Register (AIRCR)

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

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

Bits	Description	
[31:16]	**VECTORKEY**	**Register Access Key** When writing this register, this field should be 0x05FA, otherwise the write action will be unpredictable. The VECTORKEY filed is used to prevent accidental write to this register from resetting the system or clearing of the exception status.
[15]	**ENDIANNESS**	**Data Endianness** 0 = Little-endian. 1 = Big-endian.
[14:11]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[10:8]	**PRIGROUP**	**Interrupt Priority Grouping** This field determines the Split Of Group priority from subpriority,
[7:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**SYSRESETREQ**	**System Reset Request** Writing This Bit to 1 Will Cause A Reset Signal To Be Asserted To The Chip And Indicate A Reset Is Requested This bit is write only and self-cleared as part of the reset sequence.
[1]	**VECTCLRACTIVE**	**Exception Active Status Clear Bit** Setting This Bit To 1 Will Clears All Active State Information For Fixed And Configurable Exceptions This bit is write only and can only be written when the core is halted. **Note:** It is the debugger's responsibility to re-initialize the stack.
[0]	**VECTRESET**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

PRIGROUP	Binary Point	Group Priority Bits	Subpriority Bits	Number Of Group Priorities	Subpriorities
0b000	bxxxxxxx.y	[7:1]	[0]	128	2
0b001	bxxxxxx.yy	[7:2]	[1:0]	64	4
0b010	bxxxxx.yyy	[7:3]	[2:0]	32	8
0b011	bxxxx.yyyy	[7:4]	[3;0]	16	16
0b100	bxxx.yyyyy	[7:5]	[4:0]	8	32
0b101	bxx.yyyyyy	[7:6]	[5:0]	4	64
0b110	bx.yyyyyyy	[7]	[6:0]	2	128
0b111	b.yyyyyyyy	None	[7:0]	1	256

Table 6.2.14-1 Priority Grouping

System Control Register (SCR)

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

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

Bits	Description	
[31:5]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[4]	**SEVONPEND**	**Send Event on Pending** 0 = Only enabled interrupts or events can wake up the processor, while 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. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**SLEEPDEEP**	**Processor Deep Sleep and Sleep Mode Selection** Control Whether the Processor Uses Sleep Or Deep Sleep as its Low Power Mode. 0 = Sleep. 1 = Deep sleep.
[1]	**SLEEPONEXIT**	**Sleep-on-exit Enable Control** This bit indicate Sleep-On-Exit when Returning from Handler Mode to Thread Mode. 0 = Do not sleep when returning to Thread mode. 1 = Enters sleep, or deep sleep, on return from an ISR to Thread mode. Setting this bit to 1 enables an interrupt driven application to avoid returning to an empty main application.
[0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

System Handler Priority Register 1 (SHPR1)

Register	Offset	R/W	Description	Reset Value
SHPR1	SCS_BA+0xD18	R/W	System Handler Priority Register 1	0x0000_0000

31	30	29	28	27	26	25	24
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	
[31:24]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[23:20]	**PRI_6**	Priority of system handler 6, UsageFault
[19:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:12]	**PRI_5**	Priority of system handler 5, BusFault
[11:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7:4]	**PRI_4**	Priority of system handler 4, MemManage
[3:0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

System Handler Priority Register 2 (SHPR2)

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

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

Bits	Description	
[31:28]	**PRI_11**	**Priority of System Handler 11 – SVCall** "0" denotes the highest priority and "0xF" denotes the lowest priority.
[27:0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

System Handler Priority Register 3 (SHPR3)

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

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

Bits	Description	
[31:28]	**PRI_15**	**Priority of System Handler 15 – SysTick** "0" denotes the highest priority and "0xF" denotes the lowest priority.
[27:24]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[23:20]	**PRI_14**	**Priority of System Handler 14 – PendSV** "0" denotes the highest priority and "0xF" denotes the lowest priority.
[19:0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

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®-M4 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 4~24.576 MHz external high speed crystal (HXT) and internal high speed RC oscillator (HIRC) to reduce the overall system power consumption. The Figure 6.3-1 shows the clock generator and the overview of the clock source control.

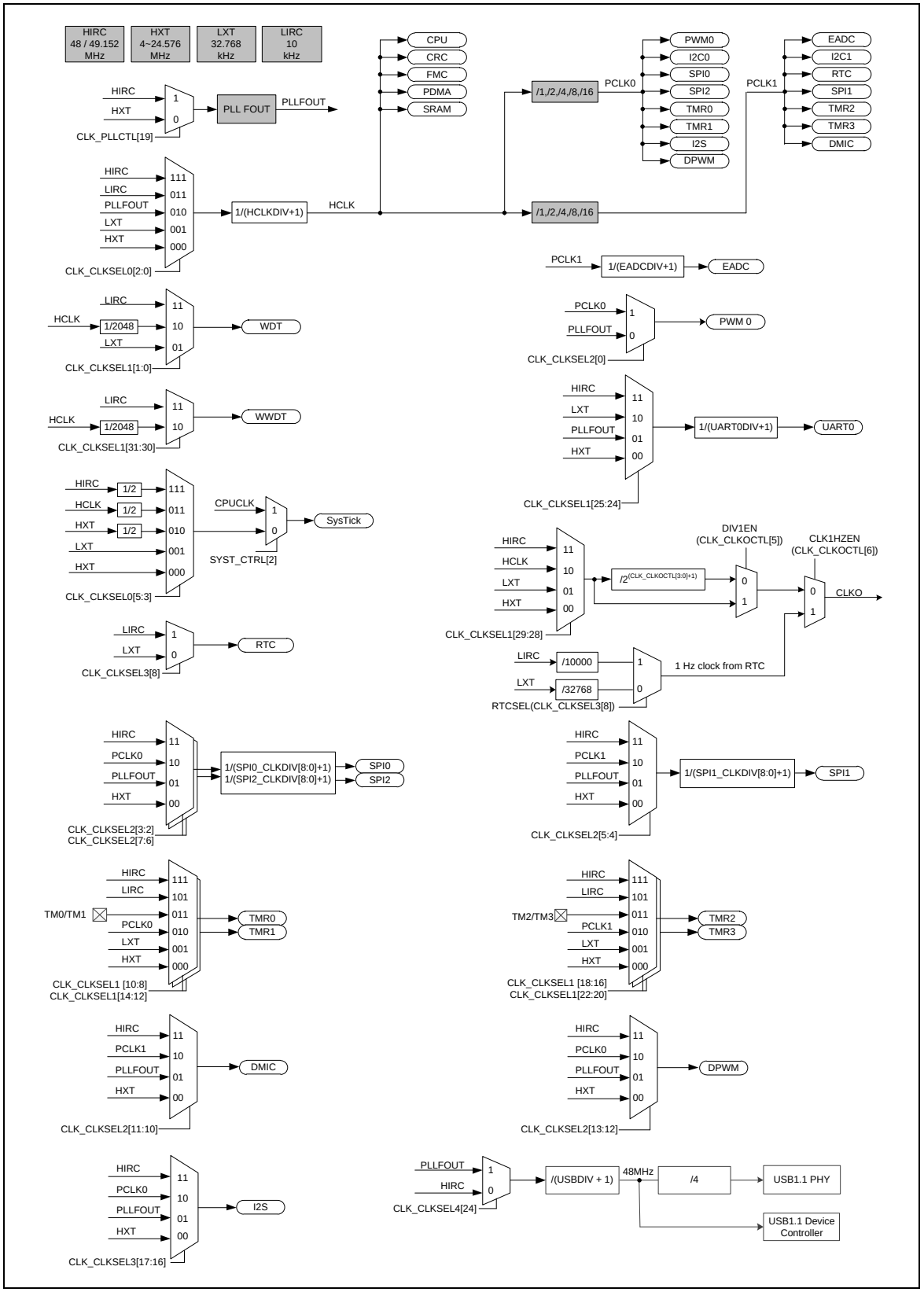


Figure 6.3-1 Clock Generator Global View Diagram

6.3.2 Clock Generator

Five clock sources can be used to drive all the internal clocks:

- 32.768 kHz external low speed crystal oscillator (LXT)
- 4~24.576 MHz external high speed crystal oscillator (HXT)
- Programmable PLL output clock frequency (PLLFOUT), PLL source can be selected from external 4~24.576 MHz external high speed crystal (HXT) or internal high speed oscillator (HIRC)
- Selectable 48.0 MHz or 49.152 MHz internal high speed RC oscillator (HIRC)
- 10 kHz internal low speed RC oscillator (LIRC)

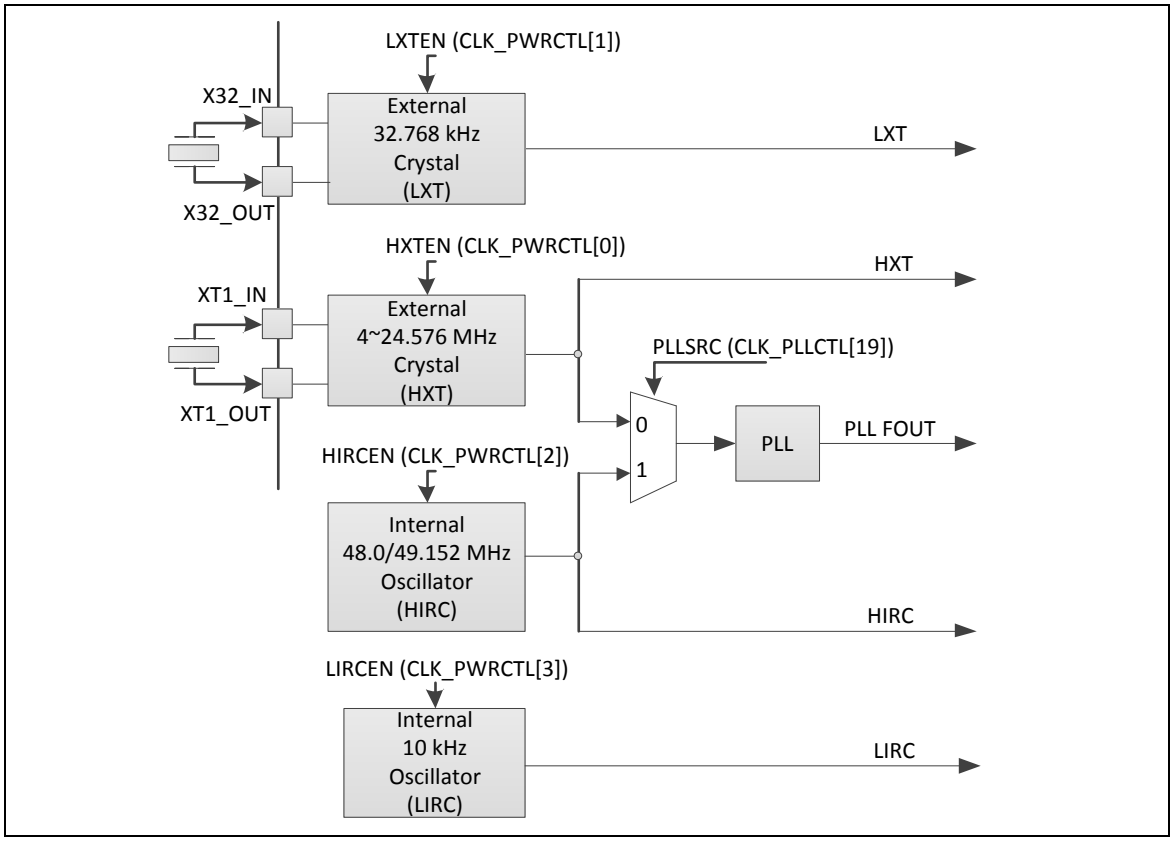


Figure 6.3-2 Clock Generator Block Diagram

6.3.3 System Clock and SysTick Clock

Five clock sources can be used to drive the system clock (HCLK), as shown in Figure 6.3-3. Clock source can be chosen by configuring HCLKSEL bits(CLK_CLKSEL0[2:0]).

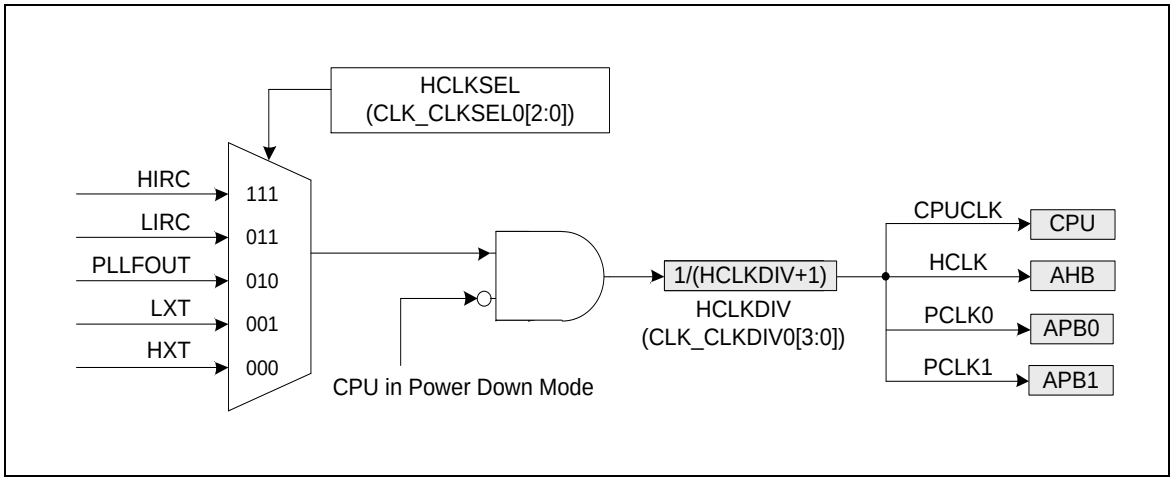


Figure 6.3-3 System Clock Block Diagram

There are two clock failure detectors monitoring HXT and LXT; each has its own enabling and interrupt control.

If HXT failure detector is enabled, the HIRC clock will be also enabled automatically. The clock controller will automatically switch the system clock (HCLK) source from HXT to HIRC if the following conditions are met:

- HCLK clock source was from HXT, or from PLLOUT and PLL source clock was from HXT,
- HXT clock failure has been detected.

An HXT clock failure condition will set HXTFIF bit (CLK_CLKDSTS[0]) 1, and raise an HXT failure interrupt if HXTFIEN (CLK_CLKDCTL[5]) is enabled.

To recover from HXT failure, user can first disable HXT, then enable HXT, and then check if the HXT clock stable bit HXTSTB (CLK_STATUS[0]) is 1. HXTSTB bit being 1 means HXT is recovered and enabled so that system clock source can be switched to HXT again.

The hardware procedure of HXT failure detection and system clock source auto switch to HIRC is shown in the Figure 6.3-4.

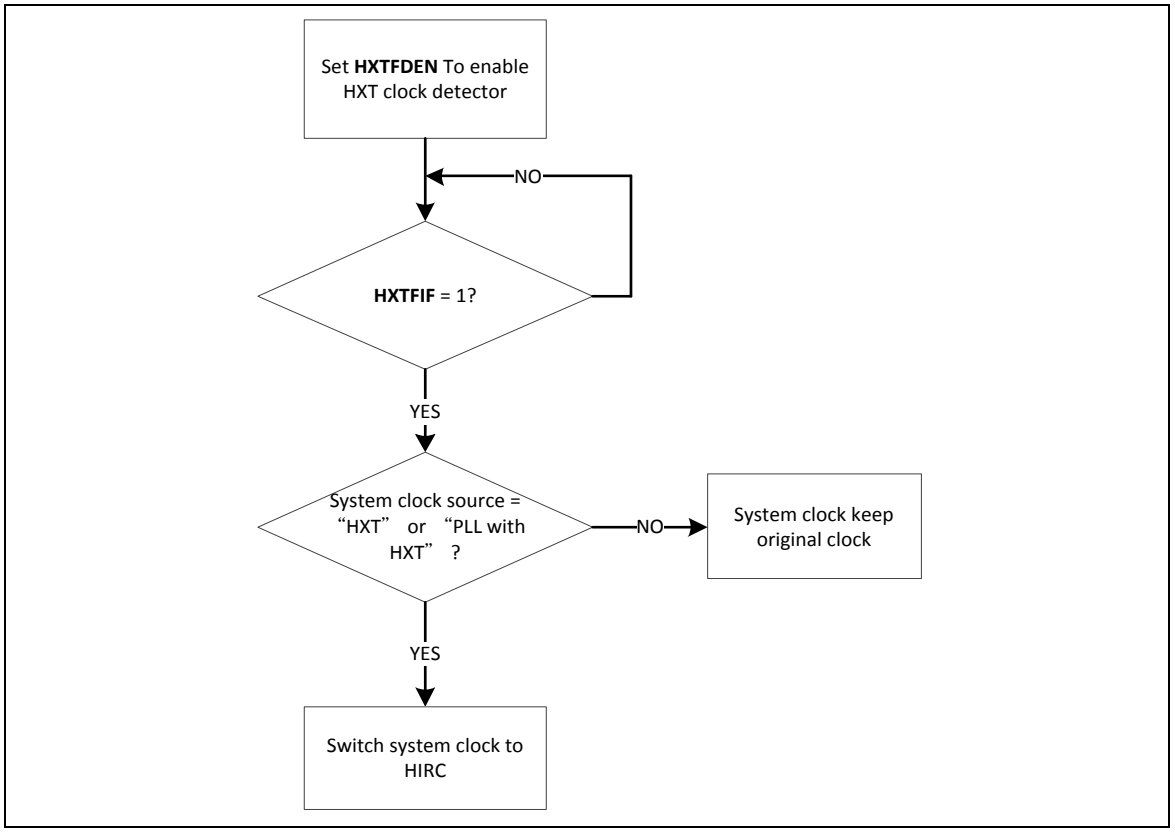


Figure 6.3-4 HXT Stop Protect Procedure

The SysTick clock source can be from CPU clock or external reference clock, determined by CLKSRC bit (SYST_CTRL[2]).

- If CLKSRC = 1, CPU core clock is used for SysTick,
- If CLKSRC = 0, SysTick clock source is from one of the 5 external reference clock, which is chosen by STCLKSEL bits (CLK_CLKSEL0[5:3]), shown in Figure 6.3-5.

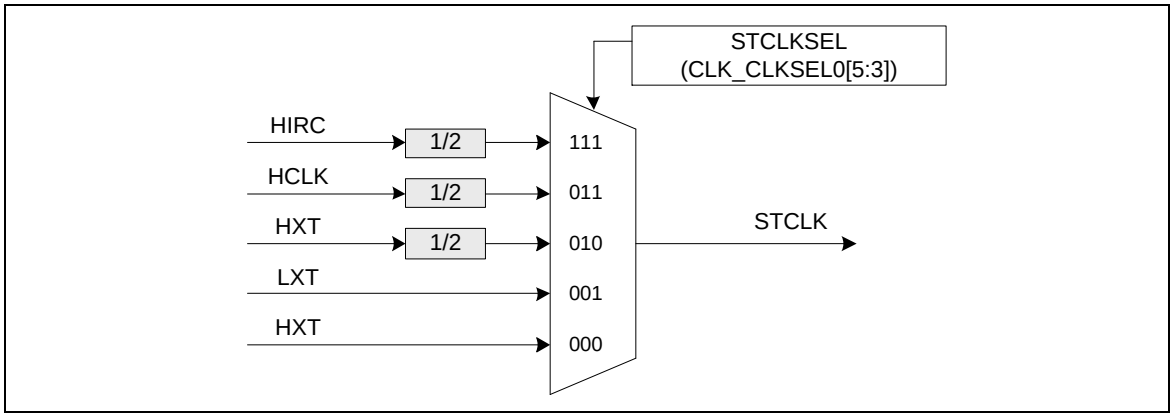


Figure 6.3-5 SysTick Clock Control Block Diagram

6.3.4 Peripheral Clock

Each peripheral module can have its own clock source selection and configuration, please refer to

CLK_CLKSEL1 and CLK_CLKSEL2 register description for more detailed information.

6.3.5 Power-down Mode Clock

Different power down modes have different impact on the system clocks. Under a certain power down mode, some clock sources (including system clocks and peripheral clocks) are disabled while some other clock sources are still available. However regardless the power down mode the following clocks are always available:

- Clock Generator
  - 10 kHz internal low speed RC oscillator (LIRC) clock
  - 32.768 kHz external low speed crystal oscillator (LXT) clock
- Peripheral Clock which uses LXT or LIRC as clock source

6.3.6 Clock Output

The NPCA121 series device is equipped with a power-of-2 frequency divider which is composed of 16 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 stays in low state.

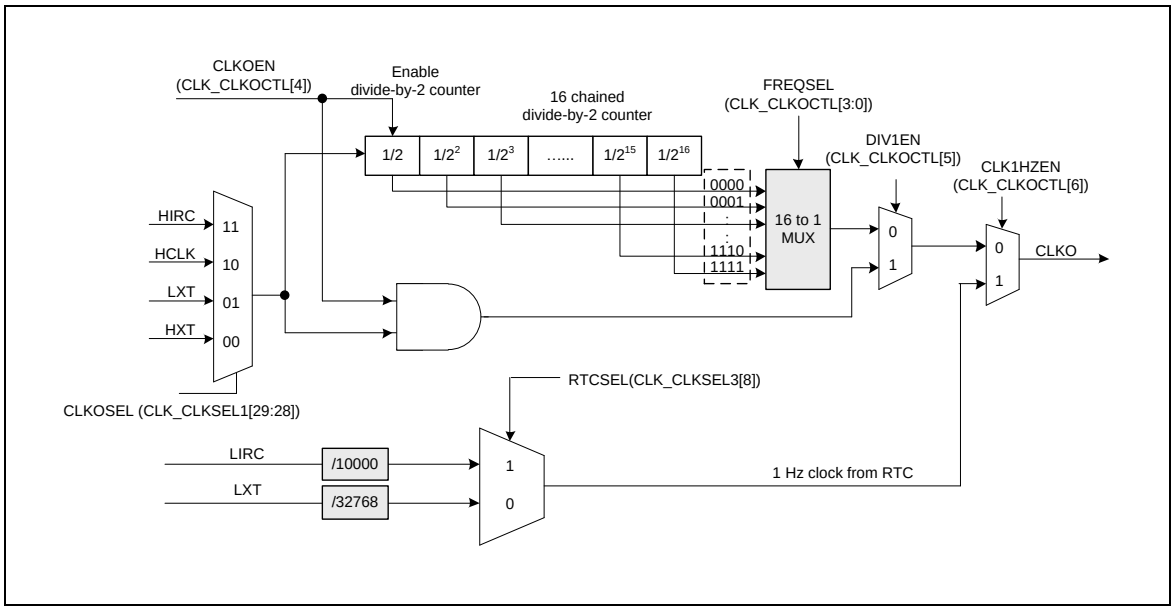


Figure 6.3-6 Clock Output Block Diagram

6.3.7 Clock Setting Limitation

The maximum frequency of PCLK0 and PCLK1 is 90 MHz. If the frequency of HCLK greater than 90 MHz, the APB1DIV (CLK_PCLKDIV[6:4]) and APB0DIV(CLK_PCLKDIV[2:0]) must be set to the appropriate value to keep the PCLK0 and PCLK1 less than or equal to 90MHz.

6.3.8 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 = 0x4000_0200**				
CLK_PWRCTL	CLK_BA+0x00	R/W	System Power-down Control Register	0x0000_1C1X
CLK_AHBCLK	CLK_BA+0x04	R/W	AHB Devices Clock Enable Control Register	0x0000_8004
CLK_APBCLK0	CLK_BA+0x08	R/W	APB Devices Clock Enable Control Register 0	0x0000_0001
CLK_APBCLK1	CLK_BA+0x0C	R/W	APB Devices Clock Enable Control Register 1	0x0000_0000
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	0xB377_7703
CLK_CLKSEL2	CLK_BA+0x18	R/W	Clock Source Select Control Register 2	0x0000_00A9
CLK_CLKSEL3	CLK_BA+0x1C	R/W	Clock Source Select Control Register 3	0x0000_0000
CLK_CLKDIV0	CLK_BA+0x20	R/W	Clock Divider Number Register 0	0x0006_0000
CLK_CLKSEL4	CLK_BA+0x24	R/W	Clock Source Select Control Register 4	0x0000_0000
CLK_PCLKDIV	CLK_BA+0x34	R/W	APB Clock Divider Register	0x0000_0000
CLK_PLLCTL	CLK_BA+0x40	R/W	PLL Control Register	0x0005_8430
CLK_STATUS	CLK_BA+0x50	R	Clock Status Monitor Register	0x0000_0018
CLK_CLKOCTL	CLK_BA+0x60	R/W	Clock Output Control Register	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 Range Detector Upper Boundary Register	0x0000_0000
CLK_CDLOWB	CLK_BA+0x7C	R/W	Clock Frequency Range Detector Lower Boundary Register	0x0000_0000
CLK_PMUCTL	CLK_BA+0x90	R/W	Power Manager Control Register	0x0000_0080
CLK_PMUSTS	CLK_BA+0x94	R/W	Power Manager Status Register	0x0000_0001
CLK_LDOCTL	CLK_BA+0x98	R/W	Chip LDO Control Register	0x0000_0000
CLK_SWKDBCTL	CLK_BA+0x9C	R/W	Standby Power-down Wake-up De-bounce Control Register	0x0000_0000
CLK_PASWKCTL	CLK_BA+0xA0	R/W	GPA Standby Power-down Wakeup Control Register	0x0000_0000
CLK_PBSWKCTL	CLK_BA+0xA4	R/W	GPB Standby Power-down Wakeup Control Register	0x0000_0000
CLK_PCSWKCTL	CLK_BA+0xA8	R/W	GPC Standby Power-down Wakeup Control Register	0x0000_0000
CLK_PDSWKCTL	CLK_BA+0xAC	R/W	GPD Standby Power-down Wakeup Control Register	0x0000_0000

CLK_IOPDCTL	CLK_BA+0xB0	R/W	GPIO Standby Power-down Control Register	0x0000_0000

Note:

1. Any register not listed here is reserved and must not be written. The result of a read operation on these bits is undefined.
2. The reserved register fields that listed in register description must be written to their reset value. Writing reserved fields with other than reset values may produce undefined results.

6.3.9 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_1C1X

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

Bits	Description	
[31:14]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[13]	**HXTTBEN**	**HXT Crystal TURBO Mode (Write Protected)** This is a protected register. Please refer to open lock sequence to program it. 0 = HXT Crystal TURBO mode disabled. 1 = HXT Crystal TURBO mode enabled.
[12]	**HXTSELTYP**	**HXT Crystal Type Select Bit (Write Protected)** This is a protected register. Please refer to open lock sequence to program it. 0 = Select INV type. 1 = Select GM type. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[11:10]	**HXTGAIN**	**HXT Gain Control Bit (Write Protected)** 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:** This bit is write protected. Refer to the SYS_REGLCTL register.
[9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8]	**PDWTCPU**	**this Bit Control the Power-down Entry Condition (Write Protected)** 1 = Chip enters Power-down mode when the both PDWTCPU and PDEN bits are set to 1 and CPU runs WFI instruction. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

[7]	**PDEN**	**System Power-down Enable (Write Protected)** 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 will not enter Power-down mode after CPU sleep command WFI. 1 = Chip enters Power-down mode after 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 any wake-up source is occurred. Refer Power Modes and Wake-up Sources chapter. **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 Protected)** 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 Protected)** When the chip wakes up from Power-down mode, the clock control will delay certain clock cycles to wait system clock stable. The delayed clock cycle is 4096 clock cycles when chip works at external high speed crystal oscillator (HXT), and 128 clock cycles when chip works at 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 Protected)** 0 = Internal low speed RC oscillator (LIRC) Disabled. 1 = Internal low speed RC oscillator (LIRC) Enabled. **Note 1:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note 2:** The reset value of this bit is 1. **Note 3:** The value of this bit must be kept 1.
[2]	**HIRCEN**	**HIRC Enable Bit (Write Protected)** 0 = Internal high speed RC oscillator (HIRC) Disabled. 1 = Internal high speed RC oscillator (HIRC) Enabled. **Note 1:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note 2:** The reset value of this bit is 1.
[1]	**LXTEN**	**LXT Enable Bit (Write Protected)** 0 = External low speed crystal (LXT) Disabled. 1 = External low speed crystal (LXT) Enabled. **Note 1:** This bit is write protected. Refer to the SYS_REGLCTL register.

		Note 2: The reset value of this bit is 0.
[0]	**HXTEN**	**HXT Enable Bit (Write Protected)** The bit default value is set by flash controller user configuration register CONFIG0 [26]. When the default clock source is from HXT, this bit is set to 1 automatically. 0 = External high speed crystal (HXT) Disabled. 1 = External high speed crystal (HXT) Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

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	0x0000_8004

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15]	**FMCIDLE**	**Flash Memory Controller Clock Enable Bit in IDLE Mode** 0 = FMC clock Disabled when chip is under IDLE mode. 1 = FMC clock Enabled when chip is under IDLE mode.
[14:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7]	**CRCCKEN**	**CRC Generator Controller Clock Enable Bit** 0 = CRC peripheral clock Disabled. 1 = CRC peripheral clock Enabled.
[6:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.

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		I2S0CKEN	EADCCKEN	USBDCKEN	Reserved		
23	22	21	20	19	18	17	16
Reserved							UART0CKEN
15	14	13	12	11	10	9	8
DMICCKEN	SPI2CKEN	SPI1CKEN	SPI0CKEN	Reserved		I2C1CKEN	I2C0CKEN
7	6	5	4	3	2	1	0
Reserved	CLKOCKEN	TMR3CKEN	TMR2CKEN	TMR1CKEN	TMR0CKEN	RTCCKEN	WDTCKEN

Bits	Description	
[31:30]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[29]	**I2S0CKEN**	**I²S0 Clock Enable Bit** 0 = I²S0 Clock Disabled. 1 = I²S0 Clock Enabled.
[28]	**EADCCKEN**	**Enhanced Analog-digital-converter (EADC) Clock Enable Bit** 0 = EADC clock Disabled. 1 = EADC clock Enabled.
[27]	**USBDCKEN**	**USB Device Clock Enable Bit** 0 = USB Device clock Disabled. 1 = USB Device clock Enabled.
[26:17]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[16]	**UART0CKEN**	**UART0 Clock Enable Bit** 0 = UART0 clock Disabled. 1 = UART0 clock Enabled.
[15]	**DMICCKEN**	**DMIC Clock Enable Bit** 0 = DMIC clock Disabled. 1 = DMIC clock Enabled.
[14]	**SPI2CKEN**	**SPI2 Clock Enable Bit** 0 = SPI2 clock Disabled. 1 = SPI2 clock Enabled.
[13]	**SPI1CKEN**	**SPI1 Clock Enable Bit** 0 = SPI1 clock Disabled.

		1 = SPI1 clock Enabled.
[12]	**SPI0CKEN**	**SPI0 Clock Enable Bit** 0 = SPI0 clock Disabled. 1 = SPI0 clock Enabled.
[11:10]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.
[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]	**RTCCKEN**	**Real-time-clock APB Interface Clock Enable Bit** This bit is used to control the RTC APB clock only. The RTC peripheral clock source is selected from RTCSEL(CLK_CLKSEL3[8]). It can be selected to 32.768 kHz external low speed crystal or 10 kHz internal low speed RC oscillator (LIRC). 0 = RTC clock Disabled. 1 = RTC clock Enabled.
[0]	**WDTCKEN**	**Watchdog Timer Clock Enable Bit (Write Protected)** 0 = Watchdog timer clock Disabled. 1 = Watchdog timer clock Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

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+0x0C	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							PWM0CKEN
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	DPWMCKEN	Reserved					

Bits	Description	
[31:17]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[16]	**PWM0CKEN**	**PWM0 Clock Enable Bit** 0 = PWM0 clock Disabled. 1 = PWM0 clock Enabled.
[15:7]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[6]	**DPWMCKEN**	**DPWM Clock Enable Bit** 0 = DPWM clock Disabled. 1 = DPWM clock Enabled.
[5:0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

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							HIRCFSEL
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		STCLKSEL			HCLKSEL		

Bits	Description	
[31:25]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[24]	**HIRCFSEL**	**Internal High Speed RC Oscillator Frequency Selection. (Write Protect)** Determines which trim setting to use for internal high speed RC oscillator. 0 = 49.152 MHz . 1 = 48.0 MHz. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[23:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5:3]	**STCLKSEL**	**Cortex®-M4 SysTick Clock Source Selection (Write Protected)** If SYST_CTRL[2]=0, SysTick uses listed clock source below. 000 = Clock source from HXT. 001 = Clock source from LXT. 010 = Clock source from HXT/2. 011 = Clock source from HCLK/2. 111 = Clock source from HIRC/2. **Note1:** 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. **Note2:** The reset value of this field is 111b. **Note3:** This bit is write protected. Refer to the SYS_REGLCTL register.
[2:0]	**HCLKSEL**	**HCLK Clock Source Selection (Write Protected)** Before clock switching, the related clock sources (both pre-select and new-select) must be turned on and stable flag must be 1. The default value is reloaded from the value of CFOSC (CONFIG0[26]) in user configuration register of Flash controller by any reset. Therefore the default value is either 000b or 111b. 000 = Clock source from HXT. 001 = Clock source from LXT.

		010 = Clock source from PLL. 011 = Clock source from LIRC. 111= Clock source from HIRC. Other = Reserved. Do not use. **Note:** This bit is 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	0xB377_7703

31	30	29	28	27	26	25	24
WWDTSEL		CLKOSEL		Reserved		UART0SEL	
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						WDTSEL	

Bits	Description	
[31:30]	**WWDTSEL**	**Window Watchdog Timer Clock Source Selection** 10 = Clock source from HCLK/2048. 11 = Clock source from internal low speed RC oscillator (LIRC). Others = Reserved. Do not use.
[29:28]	**CLKOSEL**	**Clock Divider Clock Source Selection** 00 = Clock source from external high speed crystal oscillator (HXT). 01 = Clock source from external low speed crystal oscillator (LXT). 10 = Clock source from HCLK. 11 = Clock source from internal high speed RC oscillator (HIRC).
[27:26]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[25:24]	**UART0SEL**	**UART0 Clock Source Selection** 00 = Clock source from external high speed crystal oscillator (HXT). 01 = Clock source from PLL. 10 = Clock source from external low speed crystal oscillator (LXT). 11 = Clock source from internal high speed RC oscillator (HIRC).
[23]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[22:20]	**TMR3SEL**	**TIMER3 Clock Source Selection** 000 = Clock source from external high speed crystal oscillator (HXT). 001 = Clock source from external low speed crystal oscillator (LXT). 010 = Clock source from PCLK1. 011 = Clock source from external clock TM3 pin. 101 = Clock source from internal low speed RC oscillator (LIRC).

		111 = Clock source from internal high speed RC oscillator (HIRC). Others = Reserved. Do not use.
[19]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[18:16]	**TMR2SEL**	**TIMER2 Clock Source Selection** 000 = Clock source from external high speed crystal oscillator (HXT). 001 = Clock source from external low speed crystal oscillator (LXT). 010 = Clock source from PCLK1. 011 = Clock source from external clock TM2 pin. 101 = Clock source from internal low speed RC oscillator (LIRC). 111 = Clock source from internal high speed RC oscillator (HIRC). Others = Reserved. Do not use.
[15]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[14:12]	**TMR1SEL**	**TIMER1 Clock Source Selection** 000 = Clock source from external high speed crystal oscillator (HXT). 001 = Clock source from external low speed crystal oscillator (LXT). 010 = Clock source from PCLK0. 011 = Clock source from external clock TM1 pin. 101 = Clock source from internal low speed RC oscillator (LIRC). 111 = Clock source from internal high speed RC oscillator (HIRC). Others = Reserved. Do not use.
[11]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[10:8]	**TMR0SEL**	**TIMER0 Clock Source Selection** 000 = Clock source from external high speed crystal oscillator (HXT). 001 = Clock source from external low speed crystal oscillator (LXT). 010 = Clock source from PCLK0. 011 = Clock source from external clock TM0 pin. 101 = Clock source from internal low speed RC oscillator (LIRC). 111 = Clock source from internal high speed RC oscillator (HIRC). Others = Reserved. Do not use.
[7:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1:0]	**WDTSEL**	**Watchdog Timer Clock Source Selection (Write Protected)** 00 = Reserved. Do not use. 01 = Clock source from external low speed crystal oscillator (LXT). 10 = Clock source from HCLK/2048. 11 = Clock source from internal low speed RC oscillator (LIRC). **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

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_CLKSEL2	CLK_BA+0x18	R/W	Clock Source Select Control Register 2	0x0000_00A9

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		DPWMSEL		DMICSEL		Reserved	
7	6	5	4	3	2	1	0
SPI2SEL		SPI1SEL		SPI0SEL		Reserved	PWM0SEL

Bits	Description	
[31:14]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[13:12]	**DPWMSEL**	**DPWM Clock Source Selection** 00 = Clock source from external high speed crystal oscillator (HXT). 01 = Clock source from PLL. 10 = Clock source from PCLK0. 11 = Clock source from internal high speed RC oscillator (HIRC).
[11:10]	**DMICSEL**	**DMIC Clock Source Selection** 00 = Clock source from external high speed crystal oscillator (HXT). 01 = Clock source from PLL. 10 = Clock source from PCLK1. 11 = Clock source from internal high speed RC oscillator (HIRC).
[9:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7:6]	**SPI2SEL**	**SPI2 Clock Source Selection** 00 = Clock source from external high speed crystal oscillator (HXT). 01 = Clock source from PLL. 10 = Clock source from PCLK0. 11 = Clock source from internal high speed RC oscillator (HIRC).
[5:4]	**SPI1SEL**	**SPI1 Clock Source Selection** 00 = Clock source from external high speed crystal oscillator (HXT). 01 = Clock source from PLL. 10 = Clock source from PCLK1. 11 = Clock source from internal high speed RC oscillator (HIRC).
[3:2]	**SPI0SEL**	**SPI0 Clock Source Selection**

		00 = Clock source from external high speed crystal oscillator (HXT). 01 = Clock source from PLL. 10 = Clock source from PCLK0. 11 = Clock source from internal high speed RC oscillator (HIRC).
[1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**PWM0SEL**	**PWM0 Clock Source Selection** The peripheral clock source of PWM0 is defined by PWM0SEL. 0 = Clock source from PLL. 1 = Clock source from PCLK0.

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+0x1C	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						I2S0SEL	
15	14	13	12	11	10	9	8
Reserved							RTCSEL
7	6	5	4	3	2	1	0
Reserved							

Bits	Description	
[31:18]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[17:16]	**I2S0SEL**	**I^2S0 Clock Source Selection** 00 = Clock source from external high speed crystal oscillator (HXT). 01 = Clock source from PLL clock. 10 = Clock source from PCLK0. 11 = Clock source from internal high speed RC oscillator (HIRC).
[15:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8]	**RTCSEL**	**RTC Clock Source Selection** 0 = Clock source from external low speed crystal oscillator (LXT). 1 = Clock source from internal low speed RC oscillator (LIRC).
[7:0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

Clock Divider Number Register 0 (CLK_CLKDIV0)

Register	Offset	R/W	Description	Reset Value
CLK_CLKDIV0	CLK_BA+0x20	R/W	Clock Divider Number Register 0	0x0006_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
EADCDIV							
15	14	13	12	11	10	9	8
Reserved				UART0DIV			
7	6	5	4	3	2	1	0
USBDIV				HCLKDIV			

Bits	Description	
[31:24]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[23:16]	**EADCDIV**	**EADC Clock Divide Number From EADC Clock Source** EADC clock frequency = (EADC clock source frequency) / (EADCDIV + 1).
[15:12]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[11:8]	**UART0DIV**	**UART0 Clock Divide Number From UART0 Clock Source** UART0 clock frequency = (UART0 clock source frequency) / (UART0DIV + 1).
[7:4]	**USBDIV**	**USB Clock Divide Number From PLL Clock** USB clock frequency = (USB clock source 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 4 (CLK_CLKSEL4)

Register	Offset	R/W	Description	Reset Value
CLK_CLKSEL4	CLK_BA+0x24	R/W	Clock Source Select Control Register 4	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							USBSEL
23	22	21	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	
[31:25]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[24]	**USBSEL**	**USB Clock Source Selection** 0 = Clock source from internal high speed RC oscillator (HIRC). 1 = Clock source from PLL.
[23:0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

APB Clock Divider Register (CLK_PCLKDIV)

Register	Offset	R/W	Description	Reset Value
CLK_PCLKDIV	CLK_BA+0x34	R/W	APB Clock Divider Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	APB1DIV			Reserved	APB0DIV		

Bits	Description	
[31:7]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[6:4]	**APB1DIV**	**APB1 Clock Divider** APB1 clock can be divided from HCLK 000: PCLK1 = HCLK. 001: PCLK1 = 1/2 HCLK. 010: PCLK1 = 1/4 HCLK. 011: PCLK1 = 1/8 HCLK. 100: PCLK1 = 1/16 HCLK. Others: Reserved. Do not use. **Note:** When the clock rate of HCLK greater than 100 MHz, the value of APB1DIV (CLK_PCLKDIV[6:4]) and APB0DIV(CLK_PCLKDIV[2:0]) must be greater than 0.
[3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**APB0DIV**	**APB0 Clock Divider** APB0 clock can be divided from HCLK 000: PCLK0 = HCLK. 001: PCLK0 = 1/2 HCLK. 010: PCLK0 = 1/4 HCLK. 011: PCLK0 = 1/8 HCLK. 100: PCLK0 = 1/16 HCLK. Others: Reserved. Do not use. **Note:** When the clock rate of HCLK greater than 100 MHz, the value of APB1DIV (CLK_PCLKDIV[6:4]) and APB0DIV(CLK_PCLKDIV[2:0]) must be greater than 0.

PLL Control Register (CLK_PLLCTL)

Register	Offset	R/W	Description	Reset Value
CLK_PLLCTL	CLK_BA+0x40	R/W	PLL Control Register	0x0005_8430

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. Any values read should be ignored. When writing to this field always write with reset value.
[23]	**STBSEL**	**PLL Stable Counter Selection (Write Protected)** 0 = PLL stable time is 6144 PLL source clock (suitable for source clock is equal to or less than 12 MHz). 1 = PLL stable time is 12288 PLL source clock (suitable for source clock is larger than 12 MHz). **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[22:20]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[19]	**PLLSRC**	**PLL Source Clock Selection (Write Protected)** 0 = PLL source clock from external high-speed crystal oscillator (HXT). 1 = PLL source clock from internal high-speed oscillator (HIRC). **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[18]	**OE**	**PLL OE (FOUT Enable) Pin Control (Write Protected)** 0 = PLL FOUT Enabled. 1 = PLL FOUT is fixed low. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[17]	**BP**	**PLL Bypass Control (Write Protected)** 0 = PLL is in normal mode (default). 1 = PLL clock output is same as PLL input clock FIN. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[16]	**PD**	**Power-down Mode (Write Protected)** 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).

		Note: This bit is write protected. Refer to the SYS_REGLCTL register.
[15:14]	**OUTDIV**	**PLL Output Divider Control (Write Protected)** Refer to the formulas below the table. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[13:9]	**INDIV**	**PLL Input Divider Control (Write Protected)** Refer to the formulas below the table. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[8:0]	**FBDIV**	**PLL Feedback Divider Control (Write Protected)** Refer to the formulas below the table. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

Output Clock Frequency formula:

$$FOUT = FIN \times \frac{2 * NF}{NR} \times \frac{1}{NO}$$

FREF = FIN / NR, where FREF is the comparison frequency for the PFD (phase frequency detector).

FVCO = FOUT * NO

For proper operation in normal mode, the following constraints must be satisfied:

4 MHz ≤ FREF ≤ 8 MHz

200 MHz ≤ FVCO ≤ 500 MHz

50 MHz ≤ FOUT ≤ 200 MHz

Symbol	Description
FOUT	Output Clock Frequency
FIN	Input (Reference) Clock Frequency
NR	Input Divider (INDIV + 1)
NF	Feedback Divider (FBDIV + 2)
NO	OUTDIV = “00” : NO = 1 OUTDIV = “01” : NO = 2 OUTDIV = “10” : NO = 2 OUTDIV = “11” : NO = 4

Table 6.3.9-1 The symbol definition of PLL Output Frequency formula

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+0x50	R	Clock Status Monitor Register	0x0000_0018

31	30	29	28	27	26	25	24
Reserved							
23	22	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	Reserved	PLLSTB	LXTSTB	HXTSTB

Bits	Description	
[31:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7]	**CLKSFAIL**	**Clock Switching Fail Flag (Read Only)** This bit is updated when software switches system clock source (CLK_CLKSEL0[2:0]). 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.
[6:5]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[4]	**HIRCSTB**	**HIRC Clock Source Stable Flag (Read Only)** 0 = Internal high speed RC oscillator (HIRC) clock is not stable or disabled. 1 = Internal high speed RC oscillator (HIRC) clock is stable and enabled.
[3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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 = External low speed crystal oscillator (LXT) clock is not stable or disabled. 1 = External low speed crystal oscillator (LXT) clock is stabled and enabled.
[0]	**HXTSTB**	**HXT Clock Source Stable Flag (Read Only)** 0 = External high speed crystal oscillator (HXT) clock is not stable or disabled. 1 = External high speed crystal oscillator (HXT) clock is stable and enabled.

Clock Output Control Register (CLK_CLKOCTL)

Register	Offset	R/W	Description	Reset Value
CLK_CLKOCTL	CLK_BA+0x60	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	CLK1HZEN	DIV1EN	CLKOEN	FREQSEL			

Bits	Description	
[31:7]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[6]	**CLK1HZEN**	**Clock Output 1Hz Enable Bit** 0 = 1 Hz clock output for RTC frequency compensation Disabled. 1 = 1 Hz clock output for RTC frequency compensation Enabled. **Note:** RTC IP need to be enabled.
[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].

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. Any values read should be ignored. When writing to this field always write with reset value.
[17]	**HXTFQIEN**	**HXT Clock Frequency Range Detector Interrupt Enable Bit** 0 = External high speed crystal oscillator (HXT) clock frequency Range Detector fail interrupt Disabled. 1 = External high speed crystal oscillator (HXT) clock frequency Range Detector fail interrupt Enabled.
[16]	**HXTFQDEN**	**HXT Clock Frequency Monitor Enable Bit** 0 = External high speed crystal oscillator (HXT) clock frequency Range Detector Disabled. 1 = External high speed crystal oscillator (HXT) clock frequency Range Detector Enabled.
[15:14]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[13]	**LXTFIEN**	**LXT Clock Fail Interrupt Enable Bit** 0 = External low speed crystal oscillator (LXT) clock fail interrupt Disabled. 1 = External low speed crystal oscillator (LXT) clock fail interrupt Enabled.
[12]	**LXTFDEN**	**LXT Clock Fail Detector Enable Bit** 0 = External low speed crystal oscillator (LXT) clock fail detector Disabled. 1 = External low speed crystal oscillator (LXT) clock fail detector Enabled.
[11:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5]	**HXTFIEN**	**HXT Clock Fail Interrupt Enable Bit** 0 = External high speed crystal oscillator (HXT) clock fail interrupt Disabled. 1 = External high speed crystal oscillator (HXT) clock fail interrupt Enabled.
[4]	**HXTFDEN**	**HXT Clock Fail Detector Enable Bit** 0 = External high speed crystal oscillator (HXT) clock fail detector Disabled. 1 = External high speed crystal oscillator (HXT) clock fail detector Enabled.

[3:0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

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. Any values read should be ignored. When writing to this field always write with reset value.
[8]	**HXTFQIF**	**HXT Clock Frequency Range Detector Interrupt Flag** 0 = External high speed crystal oscillator (HXT) clock frequency is normal. 1 = External high speed crystal oscillator (HXT) clock frequency is abnormal. **Note:** Write 1 to clear the bit to 0.
[7:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1]	**LXTFIF**	**LXT Clock Fail Interrupt Flag** 0 = External low speed crystal oscillator (LXT) clock is normal. 1 = External low speed crystal oscillator (LXT) stops. **Note:** Write 1 to clear the bit to 0.
[0]	**HXTFIF**	**HXT Clock Fail Interrupt Flag** 0 = External high speed crystal oscillator (HXT) clock is normal. 1 = External high speed crystal oscillator (HXT) clock stops. **Note:** Write 1 to clear the bit to 0.

Clock Frequency Detector Upper Boundary Register (CLK_CDUPB)

Register	Offset	R/W	Description	Reset Value
CLK_CDUPB	CLK_BA+0x78	R/W	Clock Frequency Range Detector Upper 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						UPERBD	
7	6	5	4	3	2	1	0
UPERBD							

Bits	Description	
[31:10]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[9:0]	**UPERBD**	**HXT Clock Frequency Range Detector Upper Boundary Value** The bits define the maximum value of frequency range detector window. The HXT detected frequency value is 512 * (the frequency of HXT / the frequency of HIRC) If the HXT detected frequency value higher than this maximum frequency value (UPERBD), the HXT Clock Frequency Range Detector Interrupt Flag (HXTFQIF(CLK_CLKDSTS[8])) will set to 1.

Clock Frequency Detector Lower Boundary Register (CLK_CDLOWB)

Register	Offset	R/W	Description	Reset Value
CLK_CDLOWB	CLK_BA+0x7C	R/W	Clock Frequency Range Detector Lower 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. Any values read should be ignored. When writing to this field always write with reset value.
[9:0]	**LOWERBD**	**HXT Clock Frequency Range Detector Lower Boundary Value** The bits define the minimum value of frequency range detector window. The HXT detected frequency value is 512 * (the frequency of HXT / the frequency of HIRC) If the HXT detected frequency value lower than this minimum frequency value (LOWERBD), the HXT Clock Frequency Range Detector Interrupt Flag (HXTFQIF(CLK_CLKDSTS[8])) will set to 1.

Power Manager Control Register (CLK_PMUCTL)

Register	Offset	R/W	Description	Reset Value
CLK_PMUCTL	CLK_BA+0x90	R/W	Power Manager Control Register	0x0000_0080

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

Bits	Description	
[31:24]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[23]	**RTCWKEN**	**RTC Wake-up Enable Bit (Write Protected)** This is a protected register. Please refer to open lock sequence to program it. 0 = RTC wake-up disable at Standby Power-down mode. 1 = RTC wake-up enabled at Standby Power-down mode. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[22:20]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[19]	**BODSPWK**	**BOD Standby Power-down Mode Wake-up Enable (Write Protected)** This is a protected register. Please refer to open lock sequence to program it. 0 = BOD wake-up disable at Standby Power-down mode. 1 = BOD wake-up enabled at Standby Power-down mode. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[18]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[17:16]	**WKPINEN**	**Wake-up Pin Enable (Write Protected)** This is a protected register. Please refer to open lock sequence to program it. 00 = Wake-up pin disable at Deep Power-down mode. 01 = Wake-up pin rising edge enabled at Deep Power-down mode. 10 = Wake-up pin falling edge enabled at Deep Power-down mode. 11 = Wake-up pin both edge enabled at Deep Power-down mode. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[15:12]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[11:9]	**WKTMRIS**	**Wake-up Timer Time-out Interval Select (Write Protected)** This is a protected register. Please refer to open lock sequence to program it.

		These bits control wake-up timer time-out interval when chip at DPD/SPD mode. 000 = Time-out interval is 128 LIRC clocks (About 12.8 ms). 001 = Time-out interval is 256 LIRC clocks (About 25.6 ms). 010 = Time-out interval is 512 LIRC clocks (About 51.2 ms). 011 = Time-out interval is 1024 LIRC clocks (About 102.4ms). 100 = Time-out interval is 4096 LIRC clocks (About 409.6ms). 101 = Time-out interval is 8192 LIRC clocks (About 819.2ms). 110 = Time-out interval is 16384 LIRC clocks (About 1638.4ms). 111 = Time-out interval is 65536 LIRC clocks (About 6553.6ms). **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[8]	**WKTMREN**	**Wake-up Timer Enable (Write Protected)** This is a protected register. Please refer to open lock sequence to program it. 0 = Wake-up timer disable at DPD/SPD mode. 1 = Wake-up timer enabled at DPD/SPD mode. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[7:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**PDMSEL**	**Power-down Mode Selection (Write Protected)** This is a protected register. Please refer to open lock sequence to program it. These bits control chip power-down mode grade selection when CPU execute WFI/WFE instruction. 000 = Power-down mode is selected. (PD) 001 = Low leakage Power-down mode is selected (LLPD). 010 = Reserved. Do not use. 011 = Reserved. Do not use. 100 = Standby Power-down mode 0 is selected (SPD0) (SRAM retention). 101 = Standby Power-down mode 1 is selected (SPD1). 110 = Deep Power-down mode is selected (DPD). 111 = Reserved. Do not use. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

Power Manager Status Register (CLK_PMUSTS)

Register	Offset	R/W	Description	Reset Value
CLK_PMUSTS	CLK_BA+0x94	R/W	Power Manager Status Register	0x0000_0001

31	30	29	28	27	26	25	24
CLRWK	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				GPDWK	GPCWK	GPBWK	GPAWK
7	6	5	4	3	2	1	0
DPD_RSTWK	SPD_TMRWK	RTCWK	BODWK	Reserved	DPD_TMRWK	PINWK	PORWK

Bits	Description	
[31]	**CLRWK**	**Clear Wake-up Flag** 0 = No clear. 1 = Clear all wake-up flag. **Note:** This bit is auto cleared by hardware.
[30:12]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[11]	**GPDWK**	**GPD Wake-up Flag (Read Only)** This flag indicates that wake-up of chip from Standby Power-down mode was requested by a transition of selected one GPD group pins. This flag is cleared when SPD mode is entered.
[10]	**GPCWK**	**GPC Wake-up Flag (Read Only)** This flag indicates that wake-up of chip from Standby Power-down mode was requested by a transition of selected one GPC group pins. This flag is cleared when SPD mode is entered.
[9]	**GPBWK**	**GPB Wake-up Flag (Read Only)** This flag indicates that wake-up of chip from Standby Power-down mode was requested by a transition of selected one GPB group pins. This flag is cleared when SPD mode is entered.
[8]	**GPAWK**	**GPA Wake-up Flag (Read Only)** This flag indicates that wake-up of chip from Standby Power-down mode was requested by a transition of selected one GPA group pins. This flag is cleared when SPD mode is entered.
[7]	**DPD_RSTWK**	**DPD Mode Reset Wake-up Flag (Read Only)** This flag indicates that wakeup of device was requested with a reset. This flag is cleared when DPD mode is entered.
[6]	**SPD_TMRWK**	**SPD Mode Wake-up Timer Wake-up Flag (Read Only)** This flag indicates that wake-up of chip was requested by wakeup timer time-out. This flag is cleared when SPD mode is entered.

[5]	**RTCWK**	**RTC Wake-up Flag (Read Only)** This flag indicates that wakeup of device from Standby Power-down mode was requested with a RTC alarm or tick time happened. This flag is cleared when SPD mode is entered.
[4]	**BODWK**	**BOD Wake-up Flag (Read Only)** This flag indicates that wakeup of device from Standby Power-down mode was requested with a BOD happened. This flag is cleared when SPD mode is entered.
[3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**DPD_TMRWK**	**DPD Mode Wake-up Timer Wake-up Flag (Read Only)** This flag indicates that wake-up of chip from Deep Power-down mode (DPD) was requested by wakeup timer time-out. This flag is cleared when DPD mode is entered.
[1]	**PINWK**	**Pin Wake-up Flag (Read Only)** This flag indicates that wake-up of chip from Deep Power-down mode was requested by a transition of the WAKEUP pin (PA.15). This flag is cleared when DPD mode is entered.
[0]	**PORWK**	**Power-on-reset Wake-up Flag (Read Only)** This flag indicates that wakeup of device was requested with a power-on reset. This flag is cleared when DPD mode is entered.

Chip LDO Register (CLK_LDOCTL)

Register	Offset	R/W	Description	Reset Value
CLK_LDOCTL	CLK_BA+0x98	R/W	Chip LDO 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							OVEN
7	6	5	4	3	2	1	0
Reserved							

Bits	Description	
[31:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8]	**OVEN**	**LDO over Drive Enable Bit** 0 = LDO keep standard voltage operating. 1 = LDO over drive voltage operating. **Note:** The bit must be set to 1 when the frequency of HCLK high than 160 MHz.
[7:0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

Standby Power-down Wake-up De-bounce Control Register (CLK_SWKDBCTL)

Register	Offset	R/W	Description	Reset Value
CLK_SWKDBCTL	CLK_BA+0x9C	R/W	Standby Power-down Wake-up De-bounce 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				SWKDBCLKSEL			

Bits	Description	
[31:4]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[3:0]	**SWKDBCLKSEL**	**Standby Power-down Wake-up De-bounce Sampling Cycle Selection** 0000 = Sample wake-up input once per 1 clocks. 0001 = Sample wake-up input once per 2 clocks. 0010 = Sample wake-up input once per 4 clocks. 0011 = Sample wake-up input once per 8 clocks. 0100 = Sample wake-up input once per 16 clocks. 0101 = Sample wake-up input once per 32 clocks. 0110 = Sample wake-up input once per 64 clocks. 0111 = Sample wake-up input once per 128 clocks. 1000 = Sample wake-up input once per 256 clocks. 1001 = Sample wake-up input once per 2*256 clocks. 1010 = Sample wake-up input once per 4*256 clocks. 1011 = Sample wake-up input once per 8*256 clocks. 1100 = Sample wake-up input once per 16*256 clocks. 1101 = Sample wake-up input once per 32*256 clocks. 1110 = Sample wake-up input once per 64*256 clocks. 1111 = Sample wake-up input once per 128*256 clocks.. **Note:** De-bounce counter clock source is the internal low speed RC oscillator (LIRC).

PA Standby Power-down Wake-up Control Register (CLK_PASWKCTL)

Register	Offset	R/W	Description	Reset Value
CLK_PASWKCTL	CLK_BA+0xA0	R/W	GPA Standby Power-down Wakeup 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							DBEN
7	6	5	4	3	2	1	0
WKPSEL				Reserved	PFWKEN	PRWKEN	WKEN

Bits	Description	
[31:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8]	**DBEN**	**PA Input Signal De-bounce Enable Bit** The DBEN bit is used to enable the de-bounce function for each corresponding I/O. 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 wakeup. The de-bounce clock source is the internal low speed RC oscillator. 0 = Standby power-down wake-up pin De-bounce function disable. 1 = Standby power-down wake-up pin De-bounce function enable. The de-bounce function is valid only for edge triggered.
[7:4]	**WKPSEL**	**PA Standby Power-down Wake-up Pin Select** 0000 = PA.0 wake-up function enabled. 0001 = PA.1 wake-up function enabled. 0010 = PA.2 wake-up function enabled. 0011 = PA.3 wake-up function enabled. 0100 = PA.4 wake-up function enabled. 0101 = PA.5 wake-up function enabled. 0110 = PA.6 wake-up function enabled. 0111 = PA.7 wake-up function enabled. 1000 = PA.8 wake-up function enabled. 1001 = PA.9 wake-up function enabled. 1010 = PA.10 wake-up function enabled. 1011 = PA.11 wake-up function enabled. 1100 = PA.12 wake-up function enabled. 1101 = PA.13 wake-up function enabled. 1110 = PA.14 wake-up function enabled. 1111 = PA.15 wake-up function enabled.
[3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with

		reset value.
[2]	**PFWKEN**	**Pin Falling Edge Wake-up Enable Bit** 0 = PA group pin falling edge wake-up function disabled. 1 = PA group pin falling edge wake-up function enabled.
[1]	**PRWKEN**	**Pin Rising Edge Wake-up Enable Bit** 0 = PA group pin rising edge wake-up function disabled. 1 = PA group pin rising edge wake-up function enabled.
[0]	**WKEN**	**Standby Power-down Pin Wake-up Enable Bit** 0 = PA group pin wake-up function disabled. 1 = PA group pin wake-up function enabled.

PB Standby Power-down Wake-up Control Register (CLK_PBSWKCTL)

Register	Offset	R/W	Description	Reset Value
CLK_PBSWKCTL	CLK_BA+0xA4	R/W	GPB Standby Power-down Wakeup 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							DBEN
7	6	5	4	3	2	1	0
WKPSEL				Reserved	PFWKEN	PRWKEN	WKEN

Bits	Description	
[31:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8]	**DBEN**	**PB Input Signal De-bounce Enable Bit** The DBEN bit is used to enable the de-bounce function for each corresponding I/O. 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 wakeup. The de-bounce clock source is the internal low speed RC oscillator. 0 = Standby power-down wake-up pin De-bounce function disable. 1 = Standby power-down wake-up pin De-bounce function enable. The de-bounce function is valid only for edge triggered.
[7:4]	**WKPSEL**	**PB Standby Power-down Wake-up Pin Select** 0000 = PB.0 wake-up function enabled. 0001 = PB.1 wake-up function enabled. 0010 = PB.2 wake-up function enabled. 0011 = PB.3 wake-up function enabled. 0100 = PB.4 wake-up function enabled. 0101 = PB.5 wake-up function enabled. 0110 = PB.6 wake-up function enabled. 0111 = PB.7 wake-up function enabled. 1000 = PB.8 wake-up function enabled. 1001 = PB.9 wake-up function enabled. 1010 = Reserved. Do not use. 1011 = Reserved. Do not use. 1100 = Reserved. Do not use. 1101 = PB.13 wake-up function enabled. 1110 = PB.14 wake-up function enabled. 1111 = PB.15 wake-up function enabled.
[3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with

		reset value.
[2]	**PFWKEN**	**Pin Falling Edge Wake-up Enable Bit** 0 = PB group pin falling edge wake-up function disabled. 1 = PB group pin falling edge wake-up function enabled.
[1]	**PRWKEN**	**Pin Rising Edge Wake-up Enable Bit** 0 = PB group pin rising edge wake-up function disabled. 1 = PB group pin rising edge wake-up function enabled.
[0]	**WKEN**	**Standby Power-down Pin Wake-up Enable Bit** 0 = PB group pin wake-up function disabled. 1 = PB group pin wake-up function enabled.

PC Standby Power-down Wake-up Control Register (CLK_PCSWKCTL)

Register	Offset	R/W	Description	Reset Value
CLK_PCSWKCTL	CLK_BA+0xA8	R/W	GPC Standby Power-down Wakeup 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							DBEN
7	6	5	4	3	2	1	0
WKPSEL				Reserved	PFWKEN	PRWKEN	WKEN

Bits	Description	
[31:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8]	**DBEN**	**PC Input Signal De-bounce Enable Bit** The DBEN bit is used to enable the de-bounce function for each corresponding I/O. 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 wakeup. The de-bounce clock source is the internal low speed RC oscillator. 0 = Standby power-down wake-up pin De-bounce function disable. 1 = Standby power-down wake-up pin De-bounce function enable. The de-bounce function is valid only for edge triggered.
[7:4]	**WKPSEL**	**PC Standby Power-down Wake-up Pin Select** 0000 = PC.0 wake-up function enabled. 0001 = PC.1 wake-up function enabled. 0010 = PC.2 wake-up function enabled. 0011 = PC.3 wake-up function enabled. 0100 = PC.4 wake-up function enabled. 0101 = PC.5 wake-up function enabled. 0110 = PC.6 wake-up function enabled. 0111 = PC.7 wake-up function enabled. 1000 = PC.8 wake-up function enabled. 1001 = PC.9 wake-up function enabled. 1010 = PC.10 wake-up function enabled. 1011 = PC.11 wake-up function enabled. 1100 = PC.12 wake-up function enabled. 1101 = PC.13 wake-up function enabled. 1110 = PC.14 wake-up function enabled. 1111 = PC.15 wake-up function enabled.
[3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with

		reset value.
[2]	**PFWKEN**	**Pin Falling Edge Wake-up Enable Bit** 0 = PC group pin falling edge wake-up function disabled. 1 = PC group pin falling edge wake-up function enabled.
[1]	**PRWKEN**	**Pin Rising Edge Wake-up Enable Bit** 0 = PC group pin rising edge wake-up function disabled. 1 = PC group pin rising edge wake-up function enabled.
[0]	**WKEN**	**Standby Power-down Pin Wake-up Enable Bit** 0 = PC group pin wake-up function disabled. 1 = PC group pin wake-up function enabled.

PD Standby Power-down Wake-up Control Register (CLK_PDSWKCTL)

Register	Offset	R/W	Description	Reset Value
CLK_PDSWKCTL	CLK_BA+0xAC	R/W	GPD Standby Power-down Wakeup 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							DBEN
7	6	5	4	3	2	1	0
WKPSEL				Reserved	PFWKEN	PRWKEN	WKEN

Bits	Description	
[31:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8]	**DBEN**	**PD Input Signal De-bounce Enable Bit** The DBEN bit is used to enable the de-bounce function for each corresponding I/O. 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 wakeup. The de-bounce clock source is the internal low speed RC oscillator. 0 = Standby power-down wake-up pin De-bounce function disable. 1 = Standby power-down wake-up pin De-bounce function enable. The de-bounce function is valid only for edge triggered.
[7:4]	**WKPSEL**	**PD Standby Power-down Wake-up Pin Select** 0000 = PD.0 wake-up function enabled. 0001 = PD.1 wake-up function enabled. 0010 = PD.2 wake-up function enabled. 0011 = PD.3 wake-up function enabled. 0100 = PD.4 wake-up function enabled. 0101 = PD.5 wake-up function enabled. 0110 = PD.6 wake-up function enabled. 0111 = PD.7 wake-up function enabled. 1000 = PD.8 wake-up function enabled. 1001 = PD.9 wake-up function enabled. 1010 = PD.10 wake-up function enabled. 1011 = PD.11 wake-up function enabled. 1100 = PD.12 wake-up function enabled. 1101 = PD.13 wake-up function enabled. 1110 = PD.14 wake-up function enabled. 1111 = PD.15 wake-up function enabled.
[3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with

		reset value.
[2]	**PFWKEN**	**Pin Falling Edge Wake-up Enable Bit** 0 = PD group pin falling edge wake-up function disabled. 1 = PD group pin falling edge wake-up function enabled.
[1]	**PRWKEN**	**Pin Rising Edge Wake-up Enable Bit** 0 = PD group pin rising edge wake-up function disabled. 1 = PD group pin rising edge wake-up function enabled.
[0]	**WKEN**	**Standby Power-down Pin Wake-up Enable Bit** 0 = PD group pin wake-up function disabled. 1 = PD group pin wake-up function enabled.

GPIO Standby Power-down Control Register (CLK_IOPDCTL)

Register	Offset	R/W	Description	Reset Value
CLK_IOPDCTL	CLK_BA+0xB0	R/W	GPIO Standby Power-down 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							IOHR

Bits	Description	
[31:1]	Reserved	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	IOHR	**GPIO Hold Release** When GPIO enter standby power-down mode, all I/O status are hold to keep normal operating status. After chip was waked up from standby power-down mode, the I/O are still keep hold status until user set this bit to release I/O hold status. This bit is auto cleared by hardware.

6.4 Flash Memory Controller (FMC)

6.4.1 Overview

The NPCA121 Series provides up to 512 KB of on-chip embedded flash for application program memory (APROM) and data flash. In-System-Programming (ISP) and In-Application-Programming (IAP) enables user to update chip embedded flash when chip is soldered on PCB. After chip powers-on, Cortex®-M4 CPU fetches code from APROM or LDROM depending on the boot select (CBS) configuration in CONFIG0. The NPCA121 Series also provides Data Flash for user to store some application dependent data to be retained when chip is powered off.

The NPCA121 Series supports configurable data flash size. The data flash size is decided by data flash enable (DFEN) in CONFIG0 and data flash base address (DFBA) in CONFIG1. When DFEN is set to 1, the data flash size is zero. When DFEN is set to 0, the APROM and data flash share 512 KB continuous address and the start address of data flash is defined by (DFBA) in CONFIG1.

6.4.2 Features

- Supports up to 512 KB of application ROM (APROM).
- Supports 4 KB loader ROM (LDROM).
- Supports Data Flash with configurable memory size.
- Supports 12 bytes User Configuration block to control system initiation.
- Supports 4 KB page erase for all embedded flash.
- Supports 32-bit/64-bit and multi-word flash programming function.
- Supports fast flash programming verification function.
- Supports CRC32 checksum calculation function.
- Supports flash all one verification function
- Supports cache memory to improve flash access performance and reduce power consumption.
- Supports In-System-Programming (ISP) / In-Application-Programming (IAP) to update embedded flash memory.
- Supports cache memory to improve flash access performance and reduce power consumption.

6.4.3 Block Diagram

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

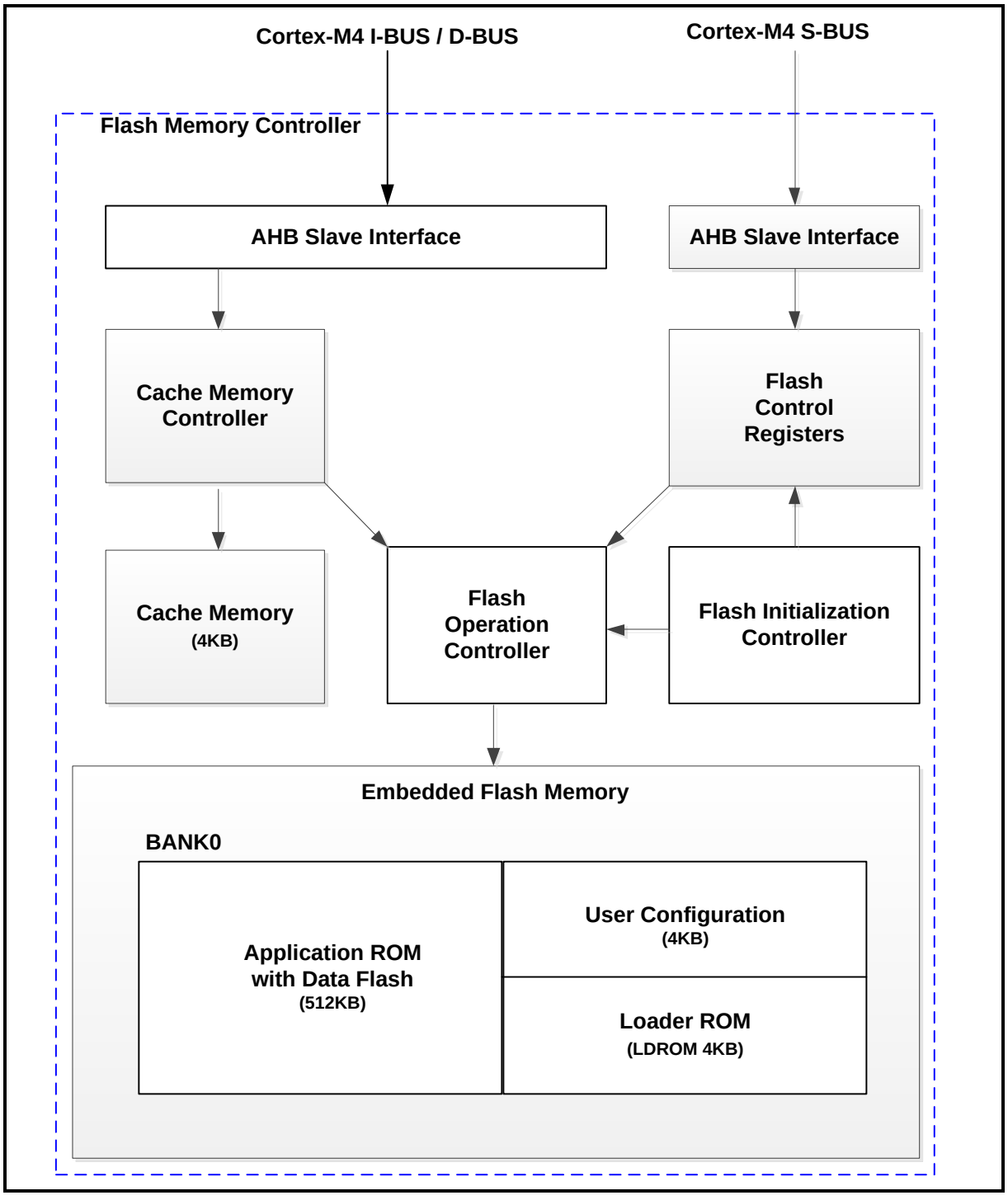


Figure 6.4-1 Flash Memory Controller Block Diagram

AHB Slave Interface

There are two AHB slave interfaces in the flash memory controller, one is from both Cortex®-M4 I-Bus and D-Bus for instruction and data fetch; the other is from Cortex®-M4 S-Bus for flash control registers access, including ISP registers.

Cache Memory Controller

A 4 KB cache memory with zero wait cycle is implemented between the Cortex®-M4 CPU and embedded flash memory. This cache memory improves the flash access performance and reduces power consumption of the embedded flash memory.

Flash Control Registers

All of ISP control and status registers are in the flash control registers. The detailed register description follows.

Flash Initialization Controller

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

Flash Operation Controller

The flash operations, such as flash erase, flash program, and flash read operation, have specific control timings for embedded flash memory. The flash operation controller generates these control timings for operations requested from the cache memory controller, 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 KB LDROM and 512 KB APROM with Data Flash. The page erase flash size is 4 KB, and minimum program bit size is 32 bits.

6.4.4 Functional Description

This section describes NPCA121 series flash memory controller's functions including memory organization, boot configuration, IAP, ISP, Flash read/write operation, checksum calculation, and so on.

6.4.4.1 Memory Organization

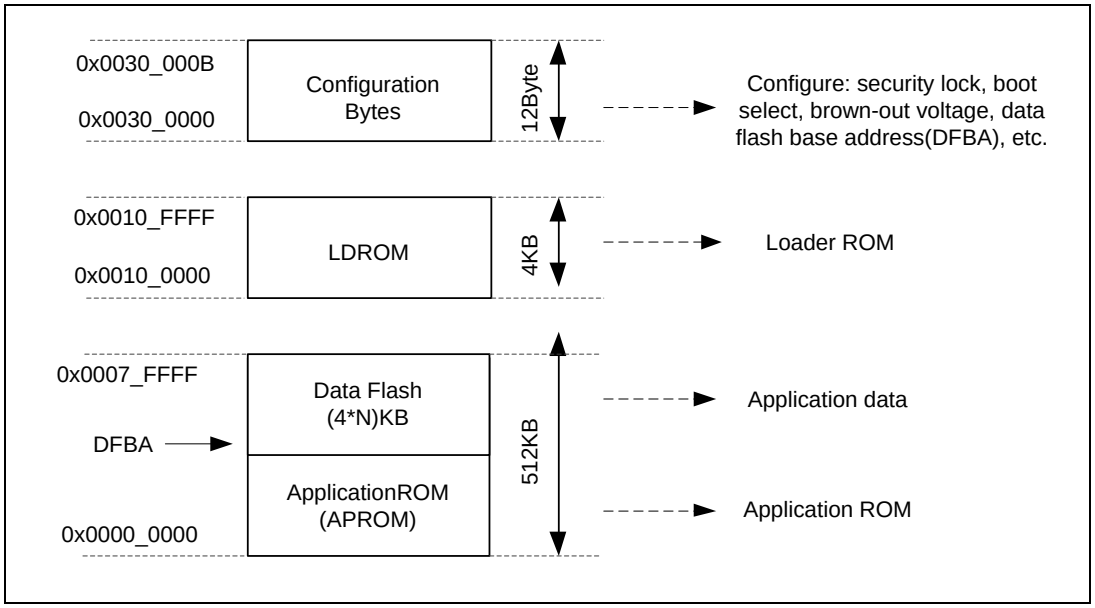


Figure 6.4-2 Memory Organization

APROM and Data Flash

APROM is main memory for user applications. 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 the embedded Flash memory is 4 KB page erased.

If DFEN bit = 1, the whole flash size will be assigned as APROM; if DFEN bit = 0, then area from 0x0000_0000 to DFBA-1 will be assigned to APROM, and the remaining flash size will be assigned as Data Flash. See Configuration Bytes description for the definition of DFEN and DFBA.

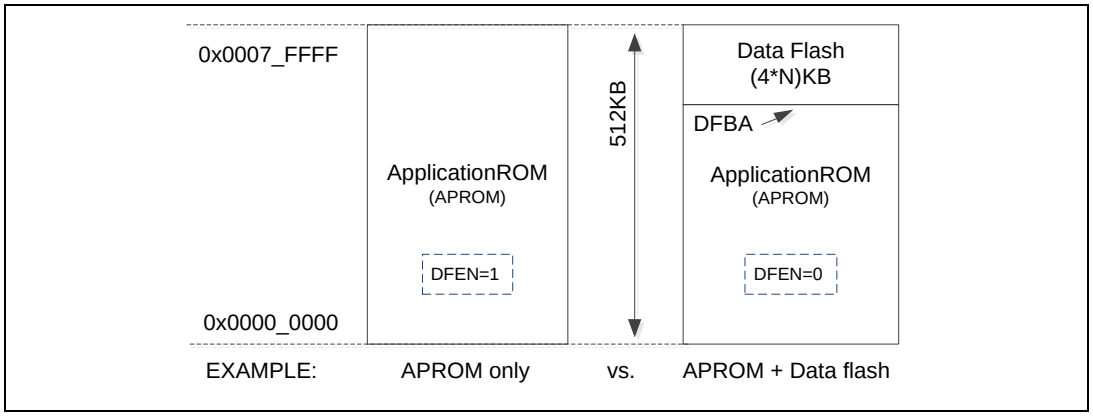


Figure 6.4-3 Data Flash Shared with 512 KB APRM example

LDROM

LDROM is designed for a loader to implement In-System-Programming (ISP) function. It is a 4 KB size flash memory region, the range of address from 0x0010_0000 to 0x0010_0FFF.

Configuration Bytes

The NPCA121 device provides 3-words of flash memory, from address 0x0030_0000 to store system configuration such as logic, flash security lock, boot select, brown-out voltage level, Data Flash base address, etc. The configuration bytes take effect after system reset.

User can call normal FMC single word read function to read these configuration bytes. To update these configuration bytes, user shall first set CFGUEN bit in FMC_ISPCTL register, and then call FMC Erase followed by FMC Write function to complete the update.

Refer to Table CONFIG0, CONFIG1 and CONFIG2 in Register Description section for details of the configuration bits.

Block Name	Size	Start Address	End Address
APROM	(512-4*N) KB	0x0000_0000	0x0007_FFFF(512 KB, if DFEN=1) DFBADR-1 (if DFEN=0)
Data Flash	4*N KB (if DFEN=0)	DFBADR (if DFEN=0)	0x0007_FFFF (512 Kbytes)
LDROM	4 KB	0x0010_0000	0x0010_0FFF
User Configuration	3 words	0x0030_0000	0x0030_000B

Note: N is the page number of configured data flash. One page size is 4096 bytes, N >= 0

Table 6.4.4-1 Flash Memory Address Map

6.4.4.2 Boot Configuration

Typically the system vector table is located in ROM space from 0x0000_0000 to 0x0000_01FF. When the processor exits from reset, it loads the MSP (Main Stack Pointer) from address 0x0000_0000, and loads the PC counter with the value pointed of the RESET Vector, which is stored at address 0x0000_0004. The processor then starts to fetch code from the address pointed to by the RESET vector and program execution begins.

The NPCA121 device has the capability of mapping vector table into APROM space from 0x0000_0000 to 0x0000_01FF, or into LDROM space from 0x0010_0000 to 0x0010_01FF. Thus provides the flexibly of booting from APROM or from LDROM.

The NPCA121 device supports In Application Programming (IAP) function that the device can update its own firmware when it is running. By booting into IAP mode, without JTAG involved, the device can re-program other parts of the ROM (APROM, LDROM or Data Flash). Under IAP mode the NPCA121 series device supports remapping function. That is, (typically after IAP programming is finished) the MCU can issue re-mapping to dynamically change the data and code read access, so that the MCU can switch execution from LDROM, APROM or SRAM.

Through CBS[1:0] bits in CONFIG0, the NPCA121 series offers four boot configurations, as shown in Table 6.4.4-2.

CBS[1:0]	Running mode	Load System Vector table from	Support Vector Re-Mapping
00	LDROM with IAP	0x0010_0000 - 0x0010_01FF	Yes
01	LDROM without IAP	0x0010_0000 - 0x0010_01FF	No
10	APROM with IAP	0x0000_0000 - 0x0010_01FF	Yes
11	APROM without IAP	0x0000_0000 - 0x0010_01FF	No

Table 6.4.4-2 Boot Configuration

6.4.4.2.1 Boot from LDROM with IAP support

By writing 0b00 into CBS[1:0] bits in CONFIG0, the NPCA121 device will map the system vector table into LRDOM space, which has physical memory address from 0x0100_0000 - 0x0100_01FF. In other words, MCU will no longer load the vector value from APROM address 0x0000_0000 to 0x0000_01FF, it loads the vector values from LDROM space 0x0010_0100 to 0x0010_01FF.

In this mode, the MCU boots from LDROM, and has the access to all memory space, including 512KB APROM and 4KB LDROM. The device can read, erase and write other parts of the memory so that the IAP function can be implemented.

This mode supports dynamic remapping so that APROM, LDROM or SRAM address can be remapped into system vector table. Remapping can be achieved by first writing the target remap-to address to FMC_ISPADDR register and then triggering ISP procedure with the “Vector Remap” command (0x2E). The targeted remapping address needs to be in alignment with 512 bytes. In VECMAP (FMC_ISPSTS[23:9]), shows the finial system vector mapping address.

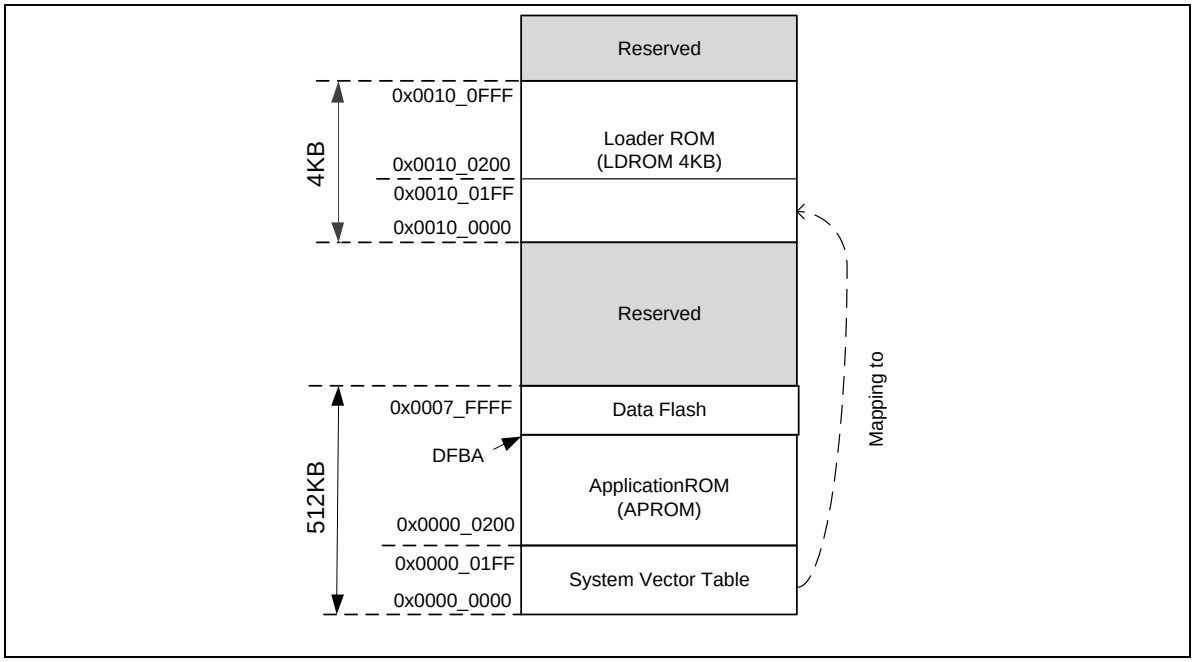


Figure 6.4-4 Boot from LDROM with IAP support

6.4.4.2.2 Boot from APROM with IAP support

By writing 0b10 into CBS[1:0] bits in CONFIG0, the NPCA121 device will load the system vector table from APROM space 0x0000_0000 - 0x0000_01FF. In this mode, except that the MCU boots from APROM, all other functions as the same as “boot from LDROM with IAP support” mode.

It has the access to all memory space, can read, erase write any other part of the APROM and LDROM. It supports IAP and remapping. Remapping can be done by first writing the target remap-to address to FMC_ISPADDR register and then triggering ISP procedure with the “Vector Remap” command (0x2E). The targeted remapping address needs to be in alignment with 512bytes.

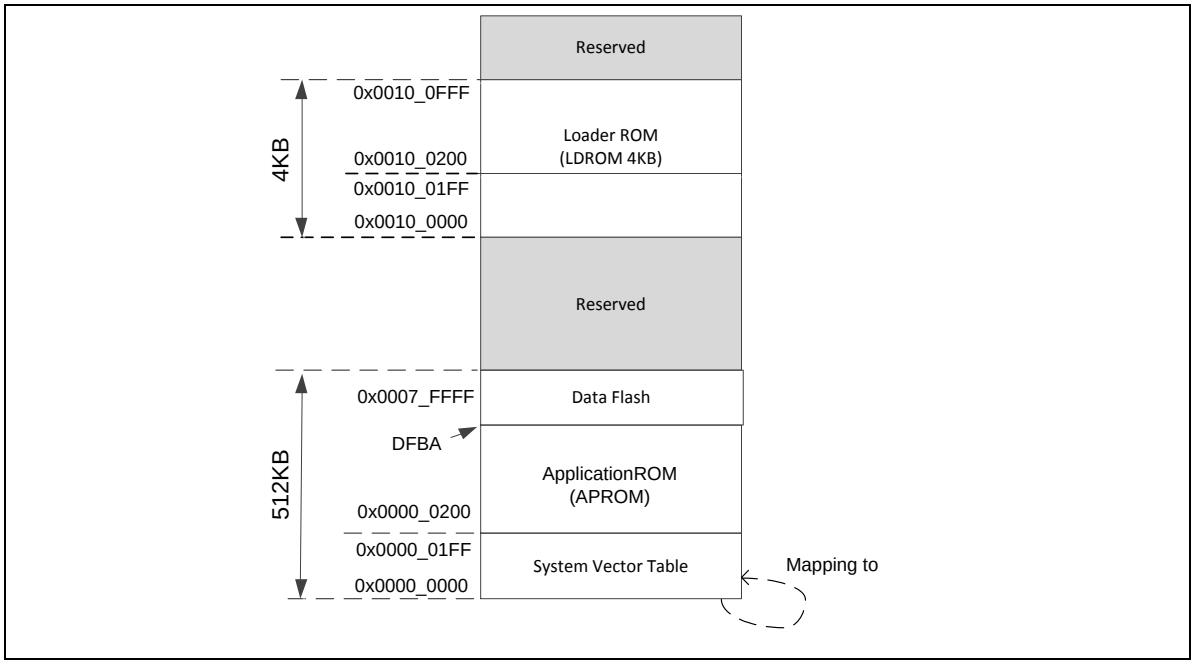


Figure 6.4-5 Boot from APROM with IAP support

6.4.4.2.3 Boot from LDROM without IAP support

By writing 0b01 into CBS[1:0] bits in CONFIG0, the NPCA121 device will load the system vector table from LDROM space 0x0010_0000 - 0x0010_01FF. In this mode the MCU boots from LDROM, and only has the access to LDROM, as shown in Figure 6.4-6.

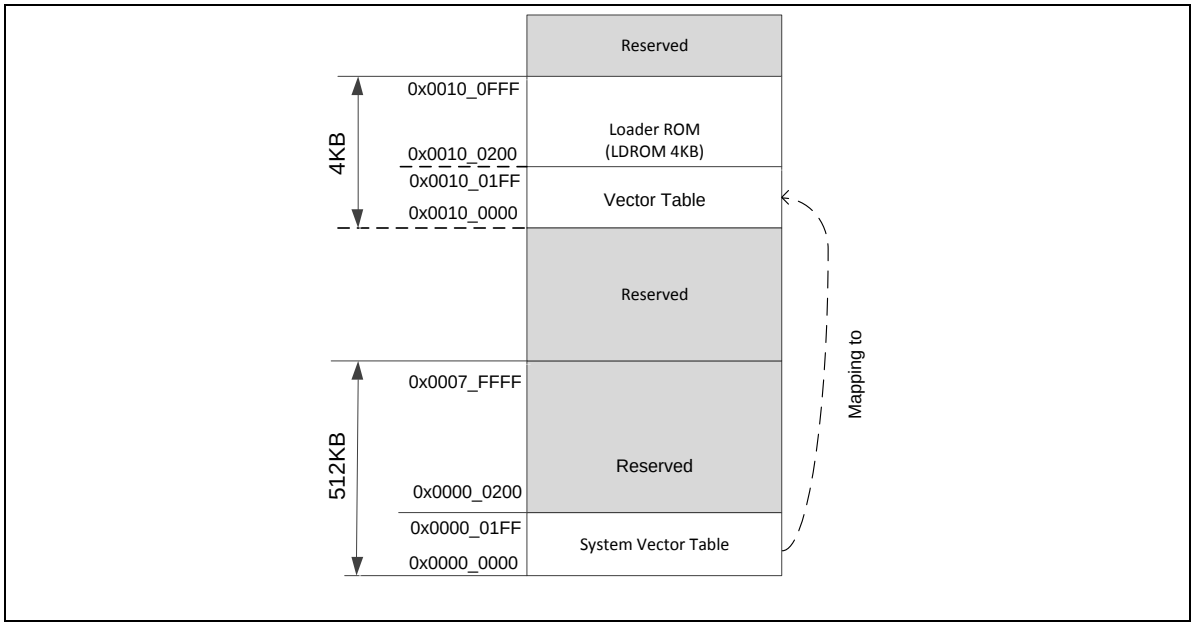


Figure 6.4-6 Boot from LDROM without IAP support

This mode does not support IAP -- because it cannot access APROM, and the LDROM has only 4 KB – one page size. In this mode remapping is not supported.

6.4.4.2.4 Boot from APROM without IAP support

By writing 0b11 into CBS[1:0] bits in CONFIG0, the NPCA121 device will load the system vector table from APROM space 0x0000_0000 - 0x0000_01FF. In this mode the MCU boots from APROM. And the MCU can only the access APROM, and data Flash if the partition for data flash exists, as shown in Figure 6.4-7.

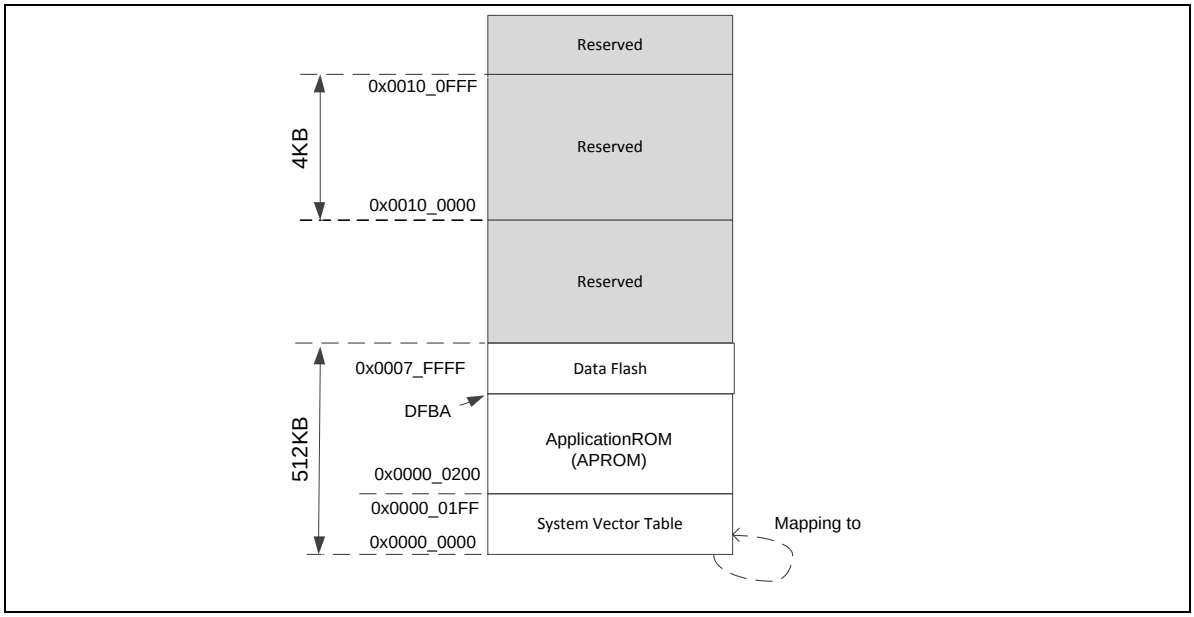


Figure 6.4-7 Boot from APROM without IAP support

In this mode the device can read, erase or write part of the APROM (or data flash). This mode

does not support remapping function.

6.4.4.3 In-Application-Programming (IAP)

The NPCA121 series provides In-Application-Programming (IAP) function for user to switch the system memory vector code executing between APROM and LDROM. 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 any time. User can change the remap address to FMC_ISPADDR and then trigger ISP procedure with the “Vector Remap” command.

6.4.4.4 In System Programming (ISP)

NPCA121 series supports In System Programming (ISP) which allows the device flash memory to be programmed by using a utility such as a NULINK JTAG device. ISP is performed without removing the microcontroller from the system, by writing data and ISP commands to the device through the on-chip connectivity interface, such as UART, I^2C or SPI.

See Table 6.4.4-3 ISP Command List for all ISP commands.

The NPCA121 series supports the following operations on the embedded flash memory:

- Page erase
- Flash data read and write (programming)
- Read company ID, Device ID, and the Unique ID
- CRC32 checksum calculation
- Flash all one verification
- Support system vector table remapping

ISP Commands

ISP Command	FMC_ISP CMD	FMC_ISPADDR	FMC_ISPDAT FMC_MPDAT0~FMC_MPDAT3
Page Erase	0x22	Valid address of flash memory organization. It must be page (4 Kbytes) alignment. Note that FMC_ISPADDR[11:0] will be ignored.	N/A
Bank Erase	0x23	Valid address of APROM of the target bank. Note that FMC_ISPADDR[15:0] will be ignored.	N/A
Block Erase	0x25	Valid address of APROM. It must be 4 pages (16 Kbytes) alignment. Note that FMC_ISPADDR[13:0] will be ignored.	N/A
32-bit Program	0x21	Valid address of flash memory organization	FMC_ISPDAT :Programming Data FMC_MPDAT0~FMC_MPDAT3 : N/A
64-bit Program	0x61	Valid address of flash memory organization	FMC_ISPDAT :N/A FMC_MPDAT0: LSB Programming Data FMC_MPDAT1: MSB Programming Data FMC_MPDAT2~FMC_MPDAT3:

			N/A
Multi-Word Program	0x27	Valid address of flash memory organization in APROM, LDROM	FMC_ISPDAT :N/A FMC_MPDAT0: 1'st Programming Data FMC_MPDAT1: 2'nd Programming Data FMC_MPDAT2: 3'rd Programming Data FMC_MPDAT3: 4'th Programming Data
FLASH Read	0x00	Valid address of flash memory organization	FMC_ISPDAT: Return Data FMC_MPDAT0~FMC_MPDAT3 : N/A
64-bit Read	0x40	Valid address of flash memory organization	FMC_ISPDAT: Return Data in FMC_ISPADDR FMC_MPDAT0: Return Data in FMC_ISPADDR FMC_MPDAT1: Return Data in FMC_ISPADDR+4 FMC_MPDAT2~FMC_MPDAT3 : N/A
Read Company ID	0x0B	0x0000_0000	FMC_ISPDAT: 0x0000_00DA FMC_MPDAT0~FMC_MPDAT3 : N/A
Read Device ID	0x0C	0x0000_0000	FMC_ISPDAT: Return Device ID FMC_MPDAT0~FMC_MPDAT3 : N/A
Read CRC32 Checksum	0x0D	0x0000_0000	FMC_ISPDAT: Return Checksum FMC_MPDAT0~FMC_MPDAT3 : N/A
Run CRC32 Checksum Calculation	0x2D	Valid start address of memory organization It must be 4 Kbytes page alignment	FMC_ISPDAT: Size It must be 4 Kbytes alignment FMC_MPDAT0~FMC_MPDAT3 : N/A
Read Flash All One Result	0x08	Keep address of "Run Flash All One Verification"	FMC_ISPDAT: Return Result 0xA110_0000 : Flash is not all one 0xA11F_FFFF: Flash is all one. FMC_MPDAT0~FMC_MPDAT3 : N/A
Run Flash All One Verification	0x28	Valid start address of memory organization It must be 4 Kbytes page alignment	FMC_ISPDAT: Size It must be 4 Kbytes alignment FMC_MPDAT0~FMC_MPDAT3 : N/A
Read Unique ID	0x04	0x0000_0000	FMC_ISPDAT: Unique ID Word 0 FMC_MPDAT0~FMC_MPDAT3 : N/A
		0x0000_0004	FMC_ISPDAT: Unique ID Word 1 FMC_MPDAT0~FMC_MPDAT3 : N/A

		0x0000_0008	FMC_ISPDAT: Unique ID Word 2 FMC_MPDAT0~FMC_MPDAT3 : N/A
Vector Remap	0x2E	Valid address in APROM,LDROM or boot loader It must be 512 bytes alignment	N/A

Table 6.4.4-3 ISP Command List

ISP Procedure

The FMC controller supports flash memory read, erase and re-programming functions. Some control bits or ISP registers are write-protected, and require unlock and lock sequence before and after access. Refer to Register Description section for detailed information.

Figure 6.4-8 illustrates the ISP flow:

Configure FMC_ISPCTL register to decide to which part of flash memory to update: LDROM, APROM or configuration bytes. Writing 1 to ISPEN (FMC_ISPCTL[0]) enables ISP module. Configure FMC_ISPCMD (for ISP command) and FMC_ISPDAT (read/write data) registers to make device ready for a specific ISP operation.

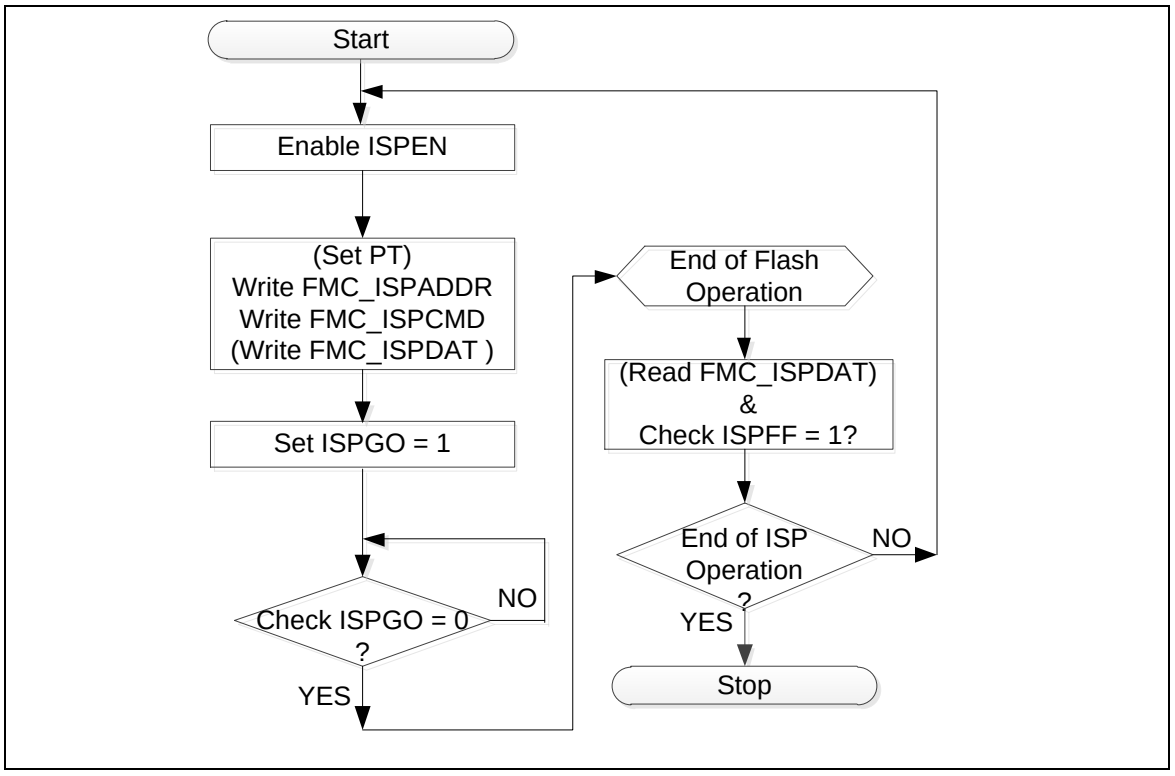


Figure 6.4-8 ISP Procedure Example

Writing 1 to ISPGO bit (FMC_ISPTRG[0]) starts the ISP function. The ISPGO (FMC_ISPTRG[0]) bit is self-cleared when ISP function finishes.

When the ISPGO (FMC_ISPTRG[0]) bit is set, CPU will wait for ISP operation to finish during this period; the peripheral still keeps working as usual. If any interrupt request occurs, CPU will not service it until ISP operation is finished. When ISP operation is finished, the ISPGO bit will be cleared by hardware automatically. Software can poll ISPGO bit to determine whether ISP operation is finished.

ISP operation status is presented in FMC_ISPSTS register. Software can check the status and do the corresponding error handling if necessary. Particularly, if ISPFF (FMC_ISPSTS[6])=1, it means ISP is actually not started and error handling is desired.

6.4.4.5 Flash Read and write Operations

The NPCA121 series supports 32-bit and 64-bit read operations. Programming wise the NPCA121 series supports 32-bit, 64-bit and multi-word flash write operations.

Table 6.4.4-4 below gives a quick reference for the registers involved in each one of the above operations. Before any read or write operation, the ISPEN bit in FMC_ISPCTRL register needs to be enabled; and after operation finishes, ISPEN bit needs to be disabled for the purpose of data security. For more detailed information about these registers, please refer to Register Description section.

Register	Description	32-Bit Read/Write	64-Bit Read/Write	Multi-Word Write
FMC_ISPCTL	ISP Control Register	✓	✓	✓
FMC_ISPADDR	ISP Address Register	✓	✓	✓
FMC_ISPDAT	ISP Data Register	✓	N/A	N/A
FMC_ISPCMD	ISP Command Register	Read: 0x00	Read: 0x40	0x27
		Write: 0x21	Write: 0x61	
FMC_ISPTRG	ISP Trigger Register	✓	✓	✓
FMC_ISPSTS	ISP Status Register	✓	✓	N/A
FMC_MPDAT0	ISP Data0 Register	N/A	✓	✓
FMC_MPDAT1	ISP Data1 Register	N/A	✓	✓
FMC_MPDAT2	ISP Data2 Register	N/A	N/A	✓
FMC_MPDAT3	ISP Data3 Register	N/A	N/A	✓
FMC_MPSTA	ISP Multi-Program status	N/A	N/A	✓
FMC_MPADDR	ISP Multi-Program Address	N/A	N/A	✓

Table 6.4.4-4 FMC control registers for Flash Read/Write

6.4.4.5.1 Read Operations

The NPCA121 series supports two kinds of flash reads: 32-bit read and 64-bit read. Please note for the purpose of data integrity, the user needs to set ISPEN bit in FMC->ISPCTL register before flash read/write operation, and clear ISPEN bit after the operation is completed.

32-bit Read

The code snippet below show how to do 32-bit read:

```
FMC->ISPCMD = FMC_ISPCMD_READ;      // op code for 32-bit read is 0x00
FMC->ISPADDR = u32Addr;             // the flash memory address where read from
FMC->ISPTRG = FMC_ISPTRG_ISPGO_Msk; // set ISPGO bit to start reading
while (FMC->ISPTRG & FMC_ISPTRG_ISPGO_Msk) ; // ISPGO bit ==1 → operation in progress
```

```
return FMC->ISPDAT;
```

64-bit Read

Code snippet below show how to do 64-bit read:

```
FMC->ISPCMD = FMC_ISPCMD_READ_64;          // op code for 64-bit read is 0x40
FMC->ISPADDR    = u32addr;
FMC->ISPDAT = 0x0;
FMC->ISPTRG = FMC_ISPTRG_ISPGO_Msk;        // set ISPGO bit to start reading
while (FMC->ISPSTS & FMC_ISPSTS_ISPBUSY_Msk) ;

…
*u32data0 = FMC->MPDAT0;
*u32data1 = FMC->MPDAT1;
```

6.4.4.5.2 Write Operations

The NPCA121 series supports 32-bit, 64-bit and multi-word flash write operations.

32-bit Write

Figure 6.4-9 illustrates the operation flow of 32-bit write. The user first writes the 32-bit data into FMC_ISPDAT register and in turn the device will write the data into the flash destination address.

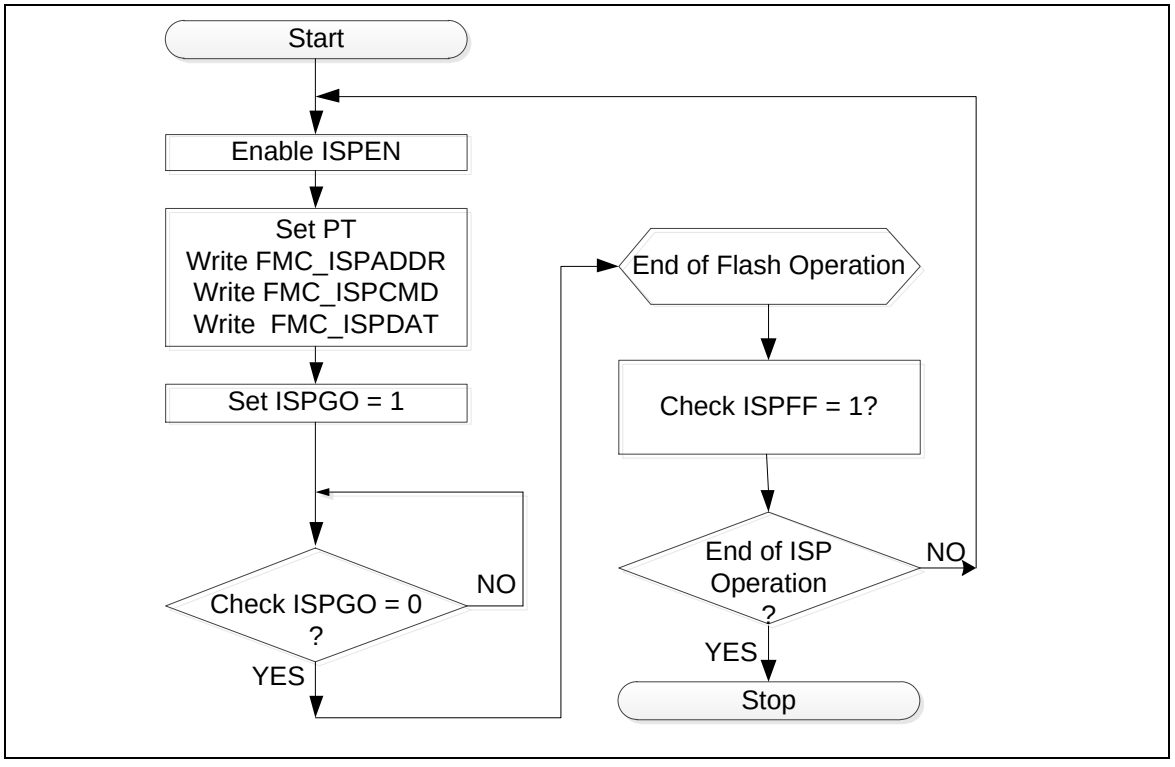


Figure 6.4-9 flash 32-bit write procedure

Code snippet for 32_bit flash write:

```
FMC->ISPCMD = FMC_ISPCMD_PROGRAM;      // op code for 32-bit write is 0x21
FMC->ISPCTL &= ~(7<<8);    // Set PT = 0
FMC->ISPADDR = u32Addr;
FMC->ISPDAT = u32Data;
FMC->ISPTRG = FMC_ISPTRG_ISPGO_Msk;
while (FMC->ISPTRG & FMC_ISPTRG_ISPGO_Msk) ;
// Check ISPFF flag (FMC_ISPCTL[6])
```

64-bit Write

Figure 6.4-10 illustrates the operation flow of 64-bit flash write. Two data registers are involved: FMC_MPDAT0 for LSB word, and FMC_MPDAT1 for MSB word. The ISP command is 0x61.

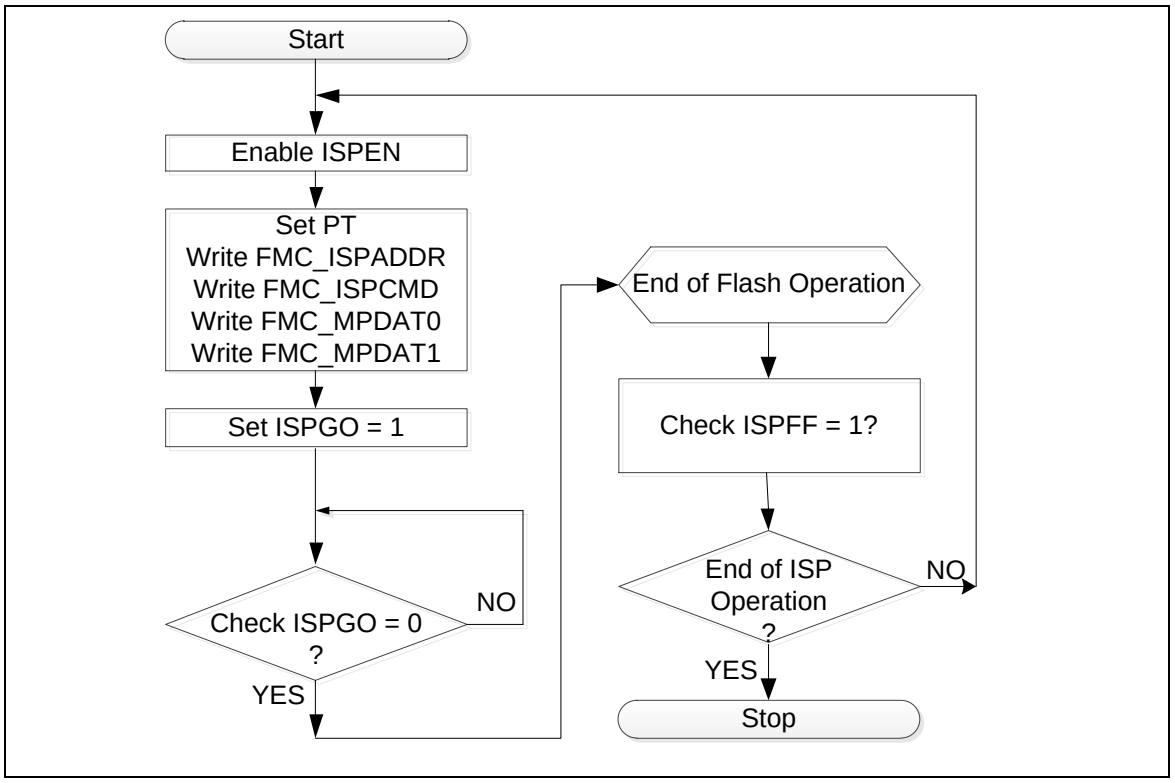


Figure 6.4-10 Flash 64-bit write procedure

Sample code for 64_bit flash write:

```
FMC->ISPCMD = FMC_ISPCMD_PROGRAM_64;              // op code for 64-bit write is 0x61
FMC->ISPCTL |= (3<<8);       //extend programming pulse
FMC->ISPADDR = u32addr;
FMC->MPDAT0 = u32data0;
FMC->MPDAT1 = u32data1;
FMC->ISPTRG = FMC_ISPTRG_ISPGO_Msk;
while (FMC->ISPSTS & FMC_ISPSTS_ISPBUSY_Msk) ;
FMC->ISPCTL &= ~(7<<8);
// Check ISPFF flag(FMC_ISPCTL[6])
…
```

Multi-word Write

The NPCA121 series supports Multi-word Write function to speed up flash programming. Each one multi-word write operation needs to write at least 16 bytes (4 words), and can write up to 512 bytes.

No matter how many bytes are to be written in one multi-word write session, there is only one flash SETUP time before writing and only one HOLD time after writing, as shown in Figure 6.4-11 below.

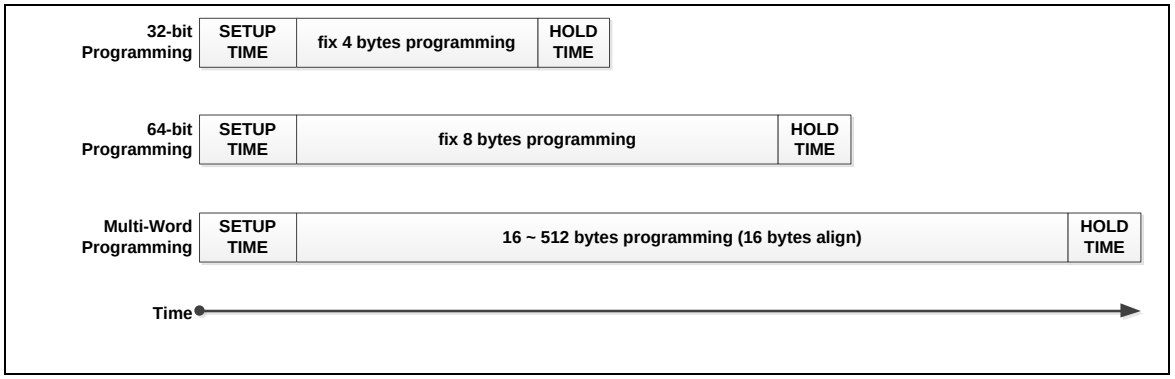


Figure 6.4-11 Timeline comparison for write operations

Within a multi-word programming session, the firmware needs to monitor buffer status, continuously prepare next batch two bytes of data in time to ensure a successful programming session. Because fetching code from APROM or LDROM cannot keep up with this pace, multi-word programing firmware code should run from SRAM.

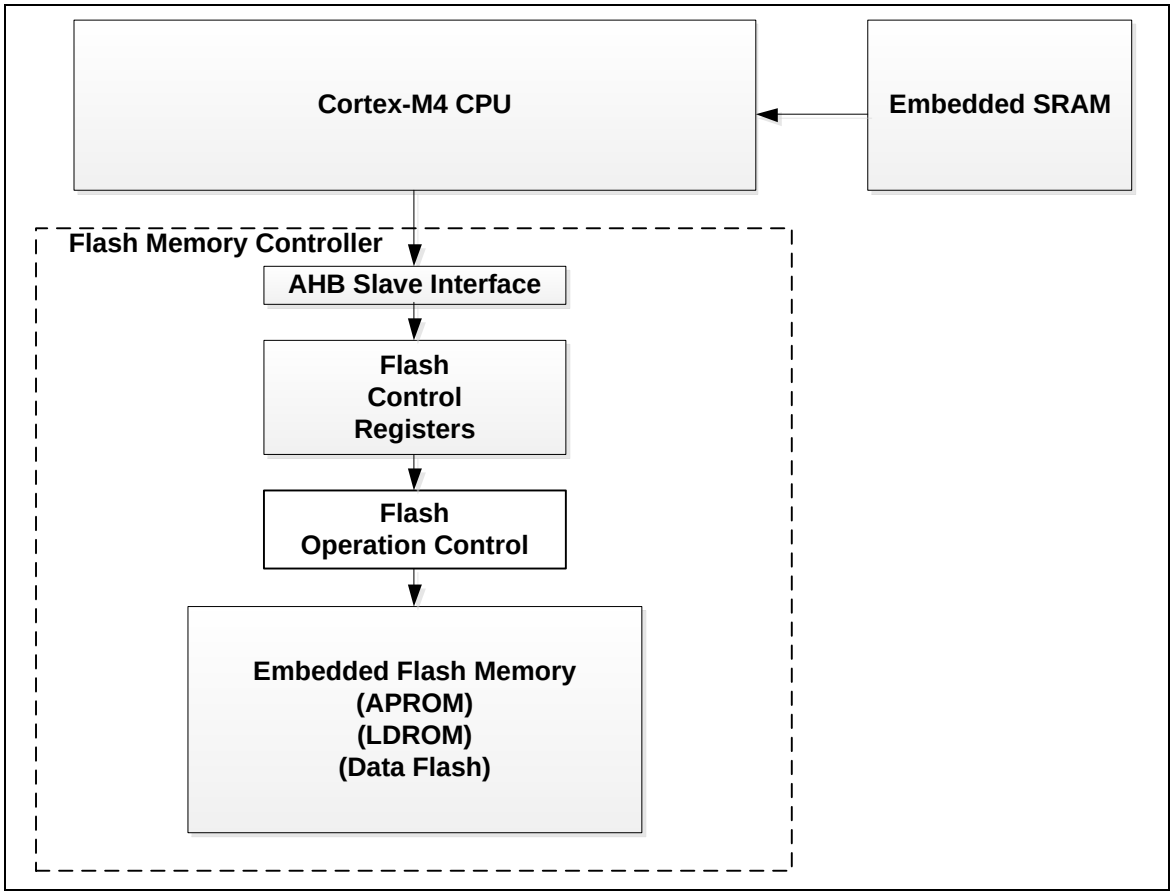


Figure 6.4-12 Firmware in SRAM for Multi-word Programming

The multi-word programming flow is shown in Figure 6.4-13. The starting ISP address (FMC_ISPADDR) has to be 16-byte align, FMC_ISPADDR[3:0] should be 0. FMC_MPDAT0 is the data word of the offset 0x0, FMC_MPDAT1 is the second word (offset 0x4), FMC_MPDAT2 is the third word (offset 0x8), and FMC_MPDAT3 is forth word (offset 0xC). If the starting ISP

address FMC_ISPADDR [3] is 0, the 1st data word should put on FMC_MPDAT0, and 2nd word is FMC_MPDAT1, 3rd word is FMC_MPDAT2, and 4th word is FMC_MPDAT3. If the starting ISP address FMC_ISPADDR [3] is 1, the 1st data word should put on FMC_MPDAT2, and 2nd word is FMC_MPDAT3, 3rd word is FMC_MPDAT0, and 4th word is FMC_MPDAT1. The maximum programming size is 512 bytes and aligns to 512-byte address. While FMC controller performs multi-word programming operation, CPU needs to monitor the buffer status D3~D0 (FMC_MPSTS[7:4]) and MPBUSY (FMC_MPSTS[0]) to wait the buffer empty ((D1,D0)=00, or (D3,D2)=00), and then CPU needs to update the next programming data (FMC_MPDAT0, FMC_MPDAT1, FMC_MPDAT2 and FMC_MPDAT3) in time. Otherwise, FMC controller will exit multi-word programming operation (MPBUSY (FMC_MPSTS[0]) = 0). If CPU cannot update the data in time (MPBUSY (FMC_MPSTS[0]) =0), CPU needs restart a new multi-word programming procedure to continue, FMC_MPADDR provides the last program address information. At the end of operation, CPU has to check ISPFF (FMC_MPSTS[2]) to confirm the multi-word operation successful complete.

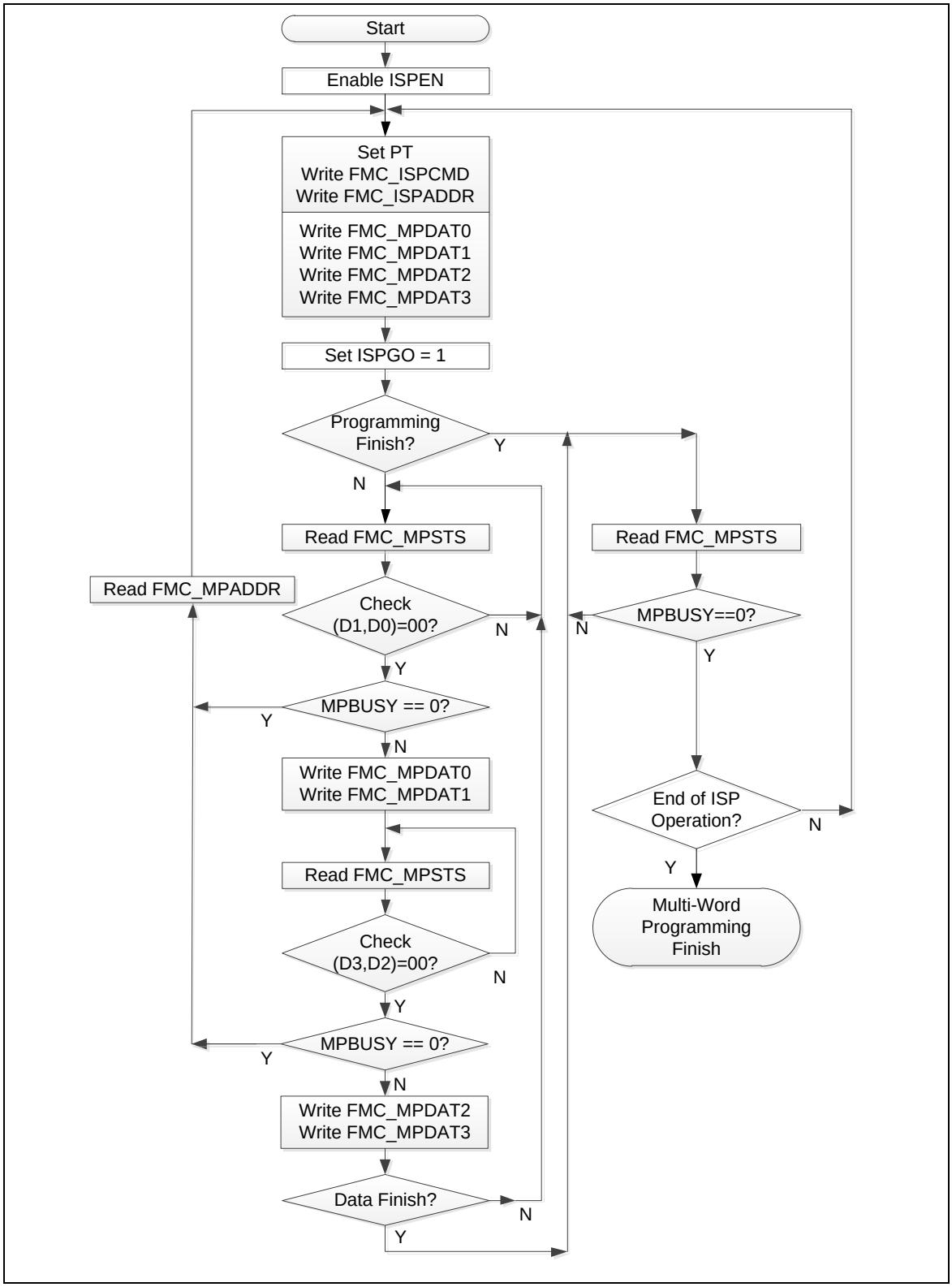


Figure 6.4-13 Multi-word programming flow chart

6.4.4.6 *Fast Flash Programming Verification*

The NPCA121 series supports fast flash programming with hardware self-verification feature, in which during programming the hardware does verification at the same time, so that it saves time for memory data read back and comparison. That is, when data is programmed to the embedded flash memory, the controller asserts the flash read operation to read data out, and performs data comparison. If there is discrepancy found, PGFF (FMC_ISPSTS[5]) flag will be set. The PGFF bit will be kept until cleared by software or a new erase operation. The NPCA121 series flash programing with self-verification feature is shown in Figure 6.4-14 below.

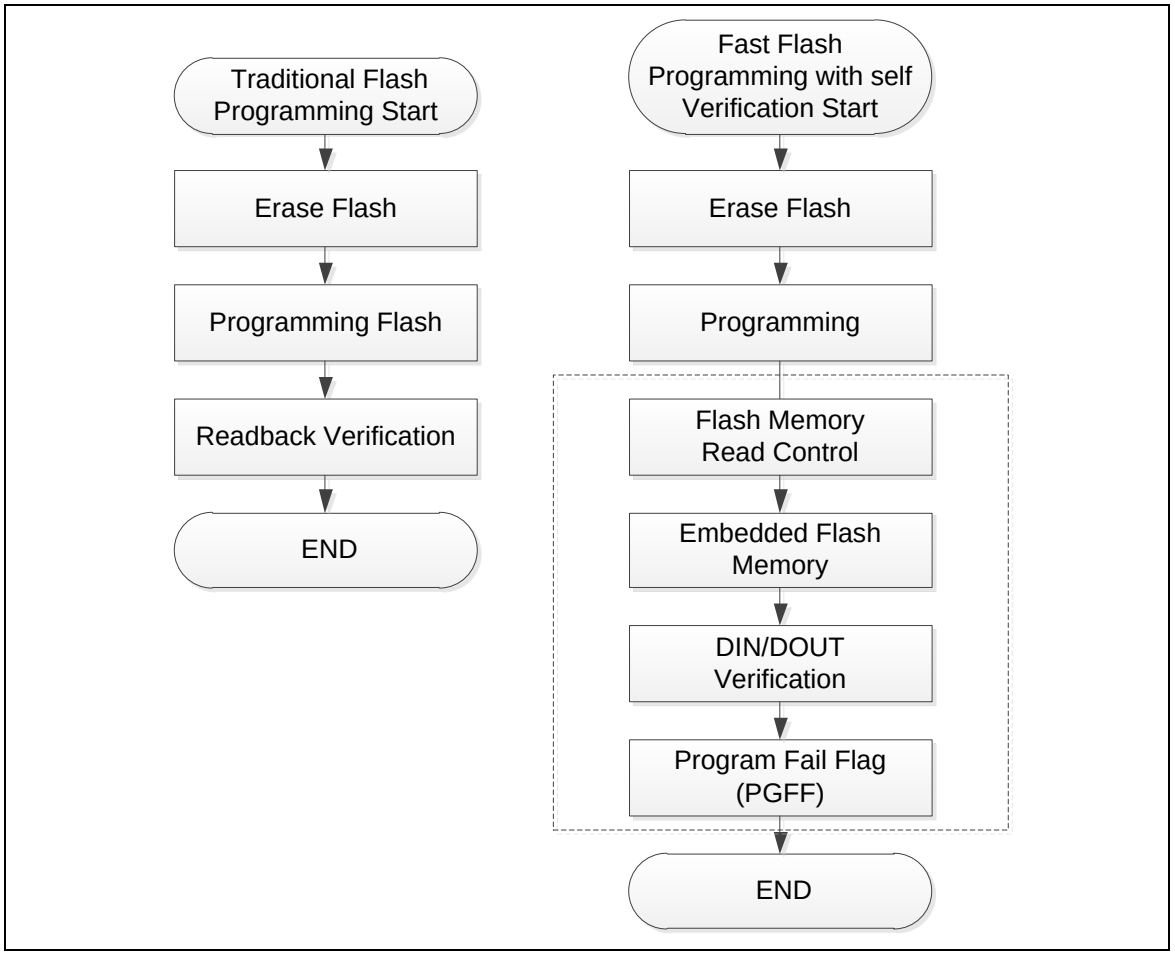


Figure 6.4-14 Fast Flash Programming Verification Flow

Compared to traditional programming operation, in which the software has to perform three-step operations to complete the programming: (1) erase flash, (2) program flash, and (3) flash data read back and compare; for NPCA121 Series software only needs to read FMC_ISPSTS to check PGFF flag in step (3) and does not have to read massive data back and compare.

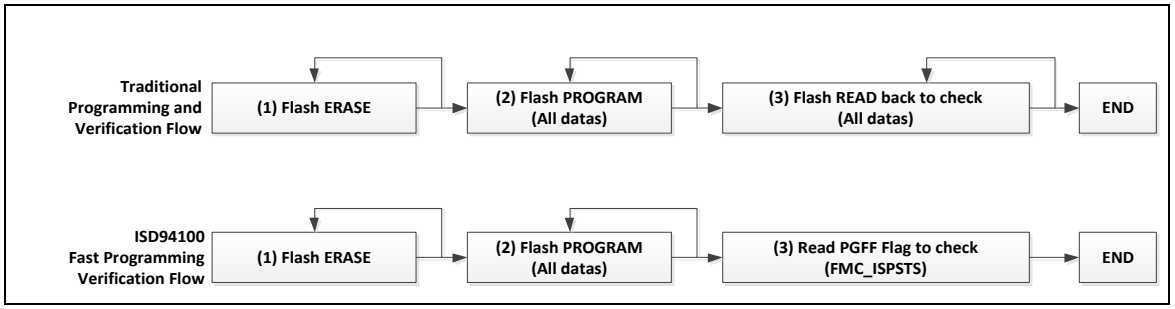


Figure 6.4-15 Verification Flow

The self-verification feature is available under 32-bit programming and 64-bit programming modes, but not available in multi-word programming mode (due to the high voltage requirement for flash programming is not suitable for continuous fast programming).

6.4.4.7 CRC32 Checksum Calculation

The NPCA121 series supports the CRC32 checksum calculation function to help user quickly check the memory content of APROM, LDROM. The CRC32 polynomial is

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

With seed = 0xFFFF_FFFF

The CRC32 checksum calculation flow is shown in Figure 6.4-16.

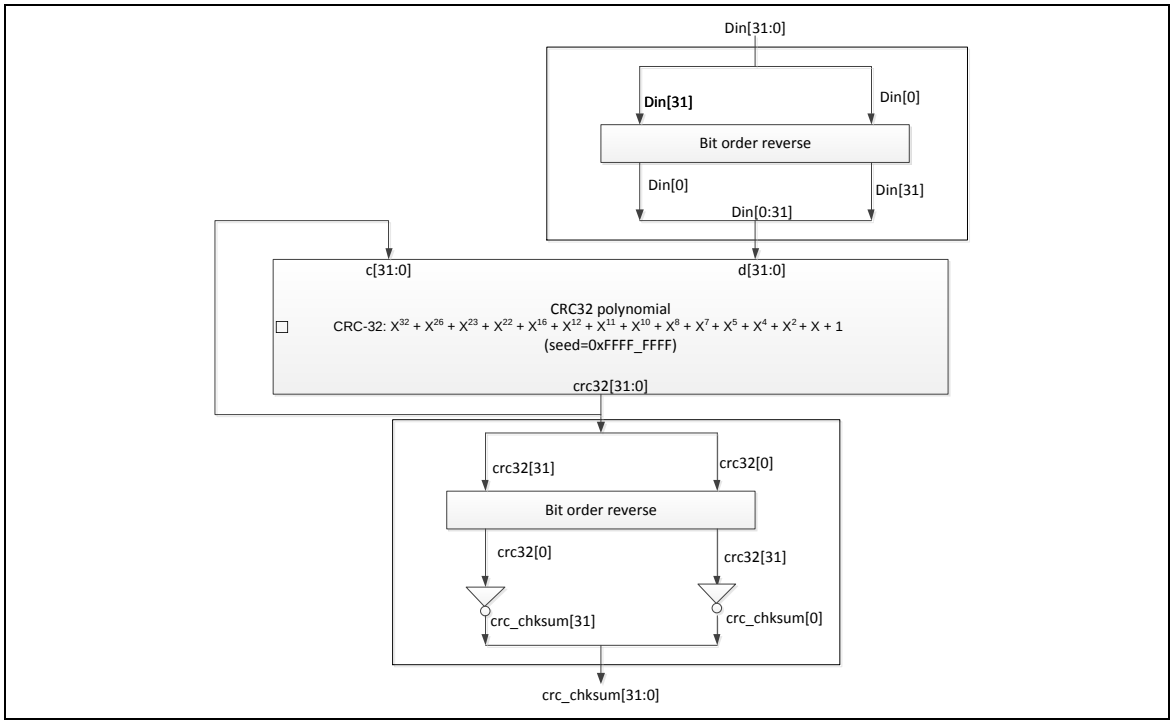


Figure 6.4-16 Flash CRC32 Checksum Calculation

Three steps complete this CRC32 checksum calculation.

Step 1: perform ISP “Run Memory CRC32 Checksum” operation

Step 2: perform ISP “Read Memory CRC32 Checksum” operation

Step 3: read FMC_ISPDAT to get checksum.

In step 1, user has to set the memory starting address (FMC_ISPADDR) and size (FMC_ISPDAT) to calculate. Both address and size have to be 4 Kbytes alignment, the size should be ≥ 4 Kbytes and the starting address includes APROM, LDROM.

In step 2, the FMC_ISPADDR should be kept as the same as step 1.

In step 3, the checksum is read from FMC_ISPDAT. If the checksum is 0x0000_0000, there is one of two conditions (1) Checksum calculation is in-progress, (2) Address and size is over device limitation

6.4.4.8 Flash All One Verification

The NPCA121 series supports the flash all one verification function which checks if a memory block are all 0xFF after a APROM or LDROM flash erase operation.

Two or Three steps complete this flash all one verification.

Two-step flow

Step 1: perform ISP “Run Flash All One Verification” operation

Step 2: read ALLONE(FMC_ISPSTA[7])bit to get the verification result

ALLONE : 1 , all of flash bits are 1 in verification block memory.

ALLONE : 0 , flash bits are not all 1 in verification block memory.

Three-step flow

Step 1: perform ISP “Run Flash All One Verification” operation

Step 2: perform ISP “Read Flash All One Result” operation

Step 3: read FMC_ISPDAT to get the verification result.

FMC_ISPDATA : 0xA11F_FFFF, all of flash bits are 1 in verification block memory.

FMC_ISPDATA : 0xA110_0000, flash bits are not all 1 in verification block memory

Before the All-One checking, write the starting address into FMC_ISPADDR register and write the memory size into FMC_ISPDAT register. Both address and size need be in 4 KB alignment. The size should be ≥ 4 KB. The starting address can point to APROM or LDROM space.

Sample code for all one checking:

```
FMC->ISPSTS = 0x80;   // clear check all one bit

FMC->ISPCMD = FMC_ISPCMD_RUN_ALL1;
FMC->ISPADDR    = u32addr;
FMC->ISPDAT = u32count;
FMC->ISPTRG = FMC_ISPTRG_ISPGO_Msk;

while (FMC->ISPSTS & FMC_ISPSTS_ISPBUSY_Msk) ;
```

6.4.4.9 Flash Access Cycle Auto-Tuning

The NPCA121 series supports the flash access cycle auto-tuning function. User don't need to set the flash access cycle by manual while system clock (HCLK) is changed, hardware will monitor the HCLK frequency and generate a optimized cycle number for flash controller to get the best performance.

Any updated registers of HCLK source and divder, are the auto-tuning trigger events, include HCLKSEL(CLK_CLKSEL0), CLK_PLLCTL, and HCLKDIV(CLK_CLKDIV0). When detecting a event, FMC will set the max number (i.e., 8) temporarily to CYCLE (FMC_CYCCTL[3:0]) register to save flash access without influence by clock changed. HIRC clock is necessary for auto-tuning to generate a exact period for HCLK counting. The HCLK detected frequency and optimized CYCLE number is showed in Table 6.4.4-5.

HCLK Clock Frequency	Optimized CYCLE Number
0 MHz ~27 MHz	1
27 MHz~51 MHz	2
51 MHz~78 MHz	3
78 MHz~105 MHz	4
105 MHz~132 MHz	5
132 MHz~159 MHz	6
159 MHz~189 MHz	7
>189 MHz	8

Table 6.4.4-5 Flash Access Optimized Cycle under auto-tuning function

The flash access cycle auto-tuning flow is showed in Figure 6.4-17.

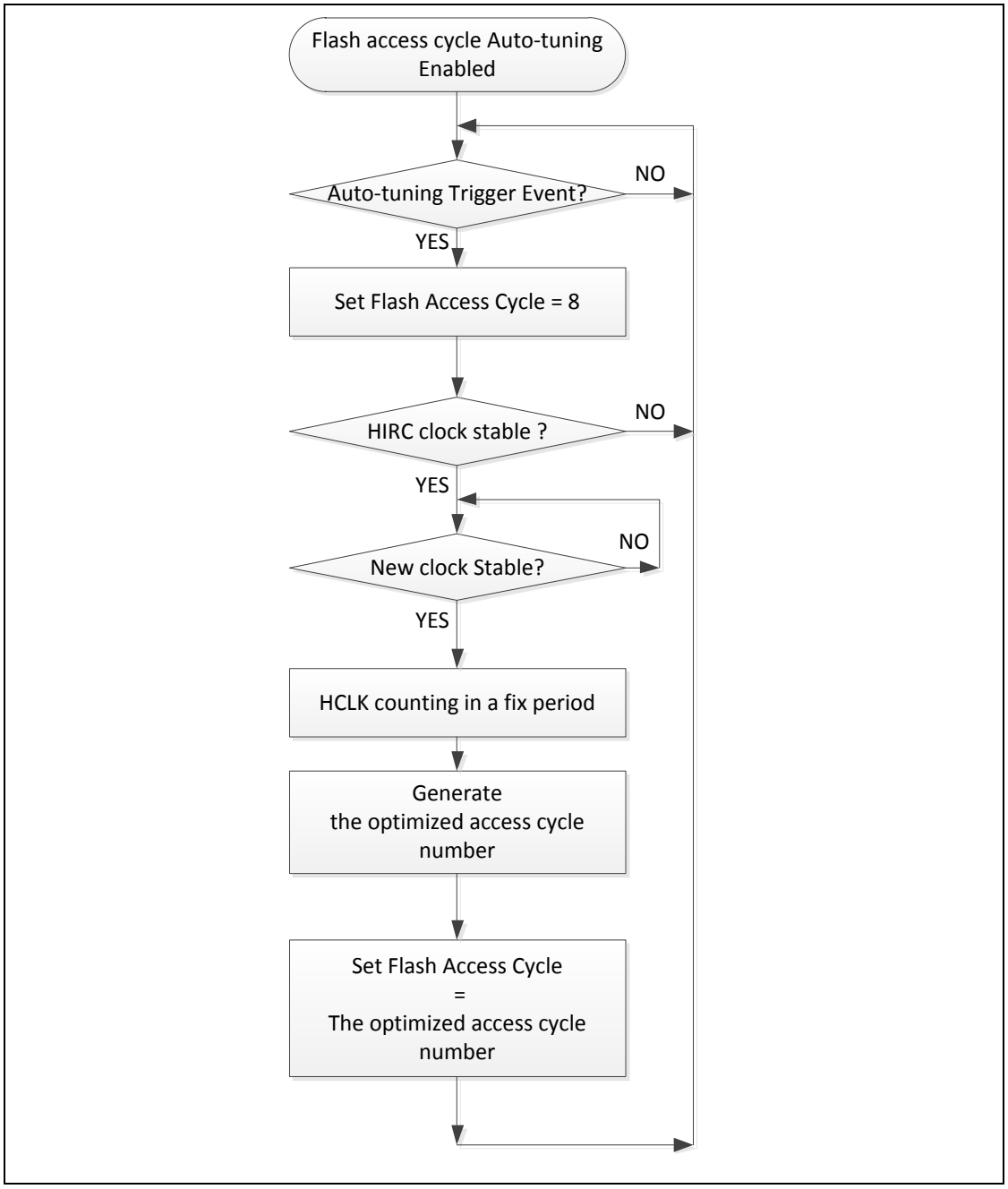


Figure 6.4-17 Flash access cycle auto-tuning flow

6.4.4.10 Lock Effect Tables

The NPCA121 series supports four kinds of protections include the Security Lock Control (i.e., LOCK in CONFIG0[1] and LOCK2 in CONFIG0[2]). In this section, NPCA121 series prepares some lock effect tables for user to understand the lock effects on APROM, LDROM and CONFIG with above two protections for CPU, SWD/ICE and ICP/Writer. Table 6.4.4-6 are the lock effect table for CPU, SWD/ICE and ICP/Writer

	LOCK2, LOCK (CONFIG0[2:1])			
	11	10	01	00
CPU, via user code present in APROM/LDROM, can erase/program/read flash memory via ISP registers, can modify registers and SRAM	YES	YES	YES	YES
Accept ICP mass erase command	YES	YES	YES	YES
SWD/ICE can use page erase/program/read flash memory by ICP	YES	NO	YES	NO
SWD/ICE can use page erase/program/read flash memory via ISP registers, can modify registers and SRAM	YES	NO	NO	NO
CortexM4 ICE can set breakpoints, read registers	YES	YES	NO	NO

Table 6.4.4-6 The lock effect table with two protections

6.4.5 Register Map

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

Configuration Bytes	Offset	R/W	Description	Reset Value
Configuration Bytes Base Address: **FMC_CONFIG_BASE = 0x0030_0000**				
CONFIG0	FMC_CONFIG_BA+0x00	R/W	Configuration byte 0x00 ~ 0x03	Flash, no change
CONFIG1	FMC_CONFIG_BA+0x04	R/W	Configuration byte 0x04 ~ 0x07	Flash, no change
CONFIG2	FMC_CONFIG_BA+0x08	R/W	Configuration byte 0x08 ~ 0x0B	Flash, no change

Note:

1. Any configuration bit not listed here is reserved and must be kept at 1.
2. The reserved field that listed in configuration bytes must be kept at 1 except CONFIG1. Programming reserved field with 0 may produce undefined results.

Register	Offset	R/W	Description	Reset Value
FMC Base Address: **FMC_BA = 0x4000_C000**				
FMC_ISPCTL	FMC_BA+0x00	R/W	ISP Control Register	0x0000_0000
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 Register	0x0000_0000
FMC_DFBA	FMC_BA+0x14	R	Data Flash Base Address	0xXXXX_XXXX
FMC_ISPSTS	FMC_BA+0x40	R	ISP Status Register	0x0000_000X
FMC_CYCCTL	FMC_BA+0x4C	R/W	Flash Access Cycle Control Register	0x0001_0000
FMC_MPDAT0	FMC_BA+0x80	R/W	ISP Multi-Word Program Data0 Register	0x0000_0000
FMC_MPDAT1	FMC_BA+0x84	R/W	ISP Multi-Word Program Data1 Register	0x0000_0000
FMC_MPDAT2	FMC_BA+0x88	R/W	ISP Multi-Word Program Data2 Register	0x0000_0000
FMC_MPDAT3	FMC_BA+0x8C	R/W	ISP Multi-Word Program Data3 Register	0x0000_0000
FMC_MPSTS	FMC_BA+0xC0	R	ISP Multi-Word Program Status Register	0x0000_0000
FMC_MPADDR	FMC_BA+0xC4	R	ISP Multi-Word Program Address Status Register	0x0000_0000

Note:

1. Any register not listed here is reserved and must not be written. The result of a read operation on these bits is undefined.
2. The reserved register fields that listed in register description must be written to their reset value. Writing reserved fields with other than reset values may produce undefined results.

6.4.6 Register Description

CONFIG0 (Address = 0x0030_0000)

31	30	29	28	27	26	25	24
CWDTEN[2]	Reserved			CFGXT1	CFOSC	Reserved	
23	22	21	20	19	18	17	16
CBOV			CBORST	CBODEN	Reserved		
15	14	13	12	11	10	9	8
Reserved				GPA8_LOW	CIOINI	Reserved	
7	6	5	4	3	2	1	0
CBS		Reserved	CWDTE[1:0]		LOCK2	LOCK	DFEN

Bits	Descriptions	
[31]	**CWDTEN[2]**	**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 forced to LIRC and LIRC can't be disabled. **CWDTEN[2:0]** is CONFIG0[31][4][3], 111 = WDT hardware enable function is inactive. Others = WDT hardware enable function is active. WDT clock is always on.
[30:28]	**Reserved**	Reserved bit should always be programmed with 1.
[27]	**CFGXT1**	**XTAL Multi-Function Select** 0 = XTAL pins are configured as GPIO pins. 1 = XTAL pins are configured as external 4~24.576 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 = External high speed crystal oscillator (HXT) 1 = Internal high speed RC oscillator (HIRC)
[25:24]	**Reserved**	Reserved bit should always be programmed with 1.
[23:21]	**CBOV**	**Brown-Out Voltage Selection** 000 = Brown-out voltage is 1.6V. 001 = Brown-out voltage is 1.8V. 010 = Brown-out voltage is 2.0V. 011 = Brown-out voltage is 2.2V. 100 = Brown-out voltage is 2.4V. 101 = Brown-out voltage is 2.6V. 110 = Brown-out voltage is 2.8V. 111 = Brown-out voltage is 3.0V.
[20]	**CBORST**	**Brown-Out Reset Enable Bit** 0 = Brown-out reset Enabled after power on. 1 = Brown-out reset Disabled after power on.

[19]	**CBODEN**	**Brown-Out Detector Enable Bit** 0 = Brown-out detect Enabled after powered on. 1 = Brown-out detect Disabled after power on.
[18:12]	**Reserved**	Reserved bit should always be programmed with 1.
[11]	**GPA8_LOW**	**Initial State of GPA8 Selection** 0 = GPA8 is low after chip power on or reset. 1 = GPA8 follows CIOINI.
[10]	**CIOINI**	**I/O Initial State Selection** 0 = All GPIO set as Quasi-bidirectional mode after chip power on. 1 = All GPIO set as input tri-state mode after power on.
[9:8]	**Reserved**	Reserved bit should always be programmed with 1.
[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[23:9]) is only be used to remap 0x0~0x1ff when CBS[0] = 0.
[5]	**Reserved**	Reserved bit should always be programmed with 1.
[4:3]	**CWDTEN**	**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], 111 = WDT hardware enable function is inactive. Others = WDT hardware enable function is active. WDT clock is always on.
[2]	**LOCK2**	**Security Lock Control 2** 0 = ICE interface is disabled. 1 = ICE operation not restricted except by LOCK (CONFIG0[1]) bit.
[1]	**LOCK**	**Security Lock Control** 0 = Flash memory content is locked. 1 = Flash memory content is not locked. When flash data is locked by LOCK, the only operation permissible on flash memory is mass erase ensuring that APROM cannot be read or configuration changed before APROM erased. LOCK will disable: Read from FLASH, Program FLASH, Page ERASE flash (APROM or CONFIG), write SRAM, access to DMA registers.
[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.

Note: The bits of configure should be 1 if reserved.

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	Descriptions	
[31:20]	**Reserved**	Reserved bit should always be programmed with 0.
[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 4 Kbytes, it is mandatory to keep bit 11-0 as 0.

Note: The bits of configure should be 0 if reserved.

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
ALOCK							
7	6	5	4	3	2	1	0
ALOCK							

Bits	Descriptions	
[31:16]	**Reserved**	Reserved bit should always be programmed with 1.
[15:0]	**ALOCK**	**Advance Security Lock Control** Must be set to 0x5A5A for access to flash memory content. Any other value, flash memory is locked. To unlock system a whole chip erase must be performed by ICP tool. Or user code can perform ISP commands to set ALOCK to 0x5A5A. Purpose of ALOCK is to ensure configuration is fully read from flash memory before ICE/ICP functions are enabled. ALOCK provides multiple protection bits to protect against over/under voltage attack or UV attack.

Note: The bits of configure should be 1 if reserved.

ISP Control Register (FMC_ISPCTL)

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

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

Bits	Description	
[31:11]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[10:8]	**PT**	**ISP Flash Program Time (Write Protected)** Leave at 000 for FLASH 32-bit program and FLASH multi-word program. Leave at 011 for FLASH 64-bit program.
[7]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[6]	**ISPFF**	**ISP Fail Flag (Write Protected)** 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) Destination address is illegal, such as over an available range. **Note:** This bit needs to be cleared by writing 1 to it.
[5]	**LDUEN**	**LDROM Update Enable Bit (Write Protected)** LDROM update enable bit. 0 = LDROM cannot be updated. 1 = LDROM can be updated.
[4]	**CFGUEN**	**Config-Bits Update By ISP Enable Bit(Write Protected)** 0 = ISP Disabled to update config-bits. 1 = ISP Enabled to update config-bits.
[3]	**APUEN**	**APROM Update Enable Bit (Write Protected)** 0 = APROM cannot be updated when the chip runs in APROM. 1 = APROM can be updated when the chip runs in APROM.
[2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[1]	**BS**	**Boot Select (Write Protected)** Set/clear this bit to select next booting from LDROM/APROM, respectively. This bit also functions as chip booting status flag, which can be used to check where chip booted from. This bit is initiated with the inversed value of CBS in CONFIG0 after any reset is happened except CPU reset (RSTS_CPU is 1) or system reset (RSTS_SYS) is happened 0 = Boot from APROM. 1 = Boot from LDROM.
[0]	**ISPEN**	**ISP Enable Bit (Write Protected)** ISP function enable bit. Set this bit to enable ISP function. 0 = ISP function Disabled. 1 = ISP function Enabled.

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 NPCA121 series is equipped with an embedded . ISPADDR[1:0] must be kept 00 for ISP 32-bit operation. ISPADDR[2:0] must be kept 000 for ISP 64-bit operation. ISPADDR[3:0] must be kept 0000 for ISP multi-word operation. For CRC32 Checksum Calculation command, this field is the flash starting address for checksum calculation, 4 Kbytes alignment is necessary for CRC32 checksum calculation. For FLASH 32-bit Program, ISP address needs word alignment (4-byte). For FLASH 64-bit Program, ISP address needs double word alignment (8-byte). For FLASH multi-word Program, ISP address needs four word alignment (16-byte).

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. When ISPFF (FMC_ISPCTL[6]) is 1, ISPDAT = 0xffff_ffff. For Run CRC32 Checksum Calculation command, ISPDAT is the memory size (byte) and 4 Kbytes alignment. For ISP Read CRC32 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 Register (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. Any values read should be ignored. When writing to this field always write with reset value.
[6:0]	CMD	**ISP Command** ISP command table is shown below: 0x00= FLASH Read. 0x04= Read Unique ID. 0x08= Read Flash All-One Result. 0x0B= Read Company ID. 0x0C= Read Device ID. 0x0D= Read Checksum. 0x21= FLASH 32-bit Program. 0x22= FLASH Page Erase. Erase page 0x23= FLASH Bank Erase. Erase all pages of APROM. 0x25= FLASH Block Erase. Erase four pages alignment of APROM. 0x27= FLASH Multi-Word Program. 0x28= Run Flash All-One Verification. 0x2D= Run Checksum Calculation. 0x2E= Vector Remap. 0x40= FLASH 64-bit Read. 0x61= FLASH 64-bit Program. 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 Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	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. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**ISPGO**	**ISP Start Trigger (Write Protected)** 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 .

ISP Status Register (FMC_ISPSTS)

Register	Offset	R/W	Description	Reset Value
FMC_ISPSTS	FMC_BA+0x40	R	ISP Status Register	0x0000_000X

31	30	29	28	27	26	25	24
Reserved							
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
ALLONE	ISPFF	PGFF	FCYCDIS	Reserved	CBS		ISPBUSY

Bits	Description	
[31:24]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[23:9]	**VECMAP**	**Vector Page Mapping Address (Read Only)** The current flash address space 0x0000_0000~0x0000_01FF is mapping to address {VECMAP[14:0], 9'h000} ~ {VECMAP[14:0], 9'h1FF}
[8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7]	**ALLONE**	**Flash All-one Verification Flag** This bit is set by hardware if all of flash bits are 1, and clear if flash bits are not all 1 after "Run Flash All-One Verification" complete; this bit also can be clear by writing 1 0 = All of flash bits are 1 after "Run Flash All-One Verification" complete. 1 = Flash bits are not all 1 after "Run Flash All-One Verification" complete.
[6]	**ISPFF**	**ISP Fail Flag (Read Only)** 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) Destination address is illegal, such as over an available range.
[5]	**PGFF**	**Flash Program with Fast Verification Flag (Read Only)** This bit is set if data is mismatched at ISP programming verification. This bit is cleared by performing ISP flash erase or ISP read CID operation 0 = Flash Program is success. 1 = Flash Program has failed. Program data is different with data in the flash memory
[4]	**FCYCDIS**	**Flash Access Cycle Auto-tuning Disabled Flag (Read Only)** This bit is set if flash access cycle auto-tunning function is disabled. The auto-tunning function is disabled by FADIS(FMC_CYCCTL[8]) or HIRC clock is not ready. 0 = Flash access cycle auto-tuning is Enabled. 1 = Flash access cyle auto-tuning is Disabled.

[3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:1]	**CBS**	**Chip Boot Selection Mode (Read Only)** This CBS field is just a copy of flash controller user configuration register CBS (CONFIG0 [7:6]). **Note:** The reset value of FMC_ISPSTS[3:0] is 1xx0b.
[0]	**ISPBUSY**	**ISP Busy Flag (Read Only)** 0 = ISP operation is finished. 1 = ISP is progressed. **Note:** The reset value of FMC_ISPSTS[3:0] is 1xx0b.

Flash Access Cyce Control Register (FMC_CYCCTL)

Register	Offset	R/W	Description	Reset Value
FMC_CYCCTL	FMC_BA+0x4C	R/W	Flash Access Cycle Control Register	0x0001_0000

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

Bits	Description	
[31:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8]	**FADIS**	**Flash Access Cycle Auto-tuning Disabled Control (Write Protect)** Set this bit to disable flash access cycle auto-tuning function 0 = Flash access cycle auto-tuning is enabled. 1 = Flash access cycle auto-tuning is disabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[7:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[3:0]	**CYCLE**	**Flash Access Cycle Control (Write Protect)** This register is updated automaticly by hardware while FCYCDIS (FMC_ISPSTS[4]) is 0, and updated by software while auto-tuning function disabled (FADIS (FMC_CYCCTL[8]) is 1) 0000 = CPU access with zero wait cycle ; flash access cycle is 1;. The HCLK working frequency range is <27 MHz; Cache is disabled by hardware. 0001 = CPU access with one wait cycle if cache miss; flash access cylcle is 1;. The HCLK working frequency range range is<27 MHz 0010 = CPU access wtih two wait cycles if cahce miss; flash access cycle is 2;. The optimized HCLK working frequency range is 27~54 MHz 0011 = CPU access with three wait cycles if cache miss; flash access cylcle is 3;. The optimized HCLK working frequency range is 54~81 MHz 0100 = CPU access with four wait cycles if cahce miss; flash access cycle is 4;. The optimized HCLK working frequency range is81~108 MHz 0101 = CPU access with five wait cycles if cache miss; flash access cylcle is 5;. The optimized HCLK working frequency range is 108~135 MHz 0110 = CPU access with six wait cycles if cahce miss; flash access cycle is 6;. The optimized HCLK working frequency range is 135~162 MHz 0111 = CPU access with seven wait cycles if cahce miss; flash access cycle is 7;. The optimized HCLK working frequency range is 162~192 MHz 1000 = CPU access with eight wait cycles if cache miss; flash access cylcle is 8;. The optimized HCLK working frequency range is >192 MHz **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

ISP Data 0 Register (FMC_MPDAT0)

Register	Offset	R/W	Description	Reset Value
FMC_MPDAT0	FMC_BA+0x80	R/W	ISP Multi-Word Program Data0 Register	0x0000_0000

31	30	29	28	27	26	25	24
ISPDAT0							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ISPDAT0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ISPDAT0							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ISPDAT0							

Bits	Description	
[31:0]	**ISPDAT0**	**ISP Data 0** This register is the first 32-bit data for 32-bit/64-bit/multi-word programming, and it is also the mirror of FMC_ISPDAT, both registers keep the same data.

ISP Data 1 Register (FMC_MPDAT1)

Register	Offset	R/W	Description	Reset Value
FMC_MPDAT1	FMC_BA+0x84	R/W	ISP Multi-Word Program Data1 Register	0x0000_0000

31	30	29	28	27	26	25	24
ISPDAT1							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ISPDAT1							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ISPDAT1							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ISPDAT1							

Bits	Description	
[31:0]	**ISPDAT1**	**ISP Data 1** This register is the second 32-bit data for 64-bit/multi-word programming.

ISP Data 2 Register (FMC_MPDAT2)

Register	Offset	R/W	Description	Reset Value
FMC_MPDAT2	FMC_BA+0x88	R/W	ISP Multi-Word Program Data2 Register	0x0000_0000

31	30	29	28	27	26	25	24
ISPDAT2							
23	22	21	20	19	18	17	16
ISPDAT2							
15	14	13	12	11	10	9	8
ISPDAT2							
7	6	5	4	3	2	1	0
ISPDAT2							

Bits	Description	
[31:0]	ISPDAT2	ISP Data 2 This register is the third 32-bit data for multi-word programming.

ISP Data 3 Register (FMC_MPDAT3)

Register	Offset	R/W	Description	Reset Value
FMC_MPDAT3	FMC_BA+0x8C	R/W	ISP Multi-Word Program Data3 Register	0x0000_0000

31	30	29	28	27	26	25	24
ISPDAT3							
23	22	21	20	19	18	17	16
ISPDAT3							
15	14	13	12	11	10	9	8
ISPDAT3							
7	6	5	4	3	2	1	0
ISPDAT3							

Bits	Description	
[31:0]	ISPDAT3	ISP Data 3 This register is the fourth 32-bit data for multi-word programming.

ISP Multi-Program Status Register (FMC_MPSTS)

Register	Offset	R/W	Description	Reset Value
FMC_MPSTS	FMC_BA+0xC0	R	ISP Multi-Word Program 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
D3	D2	D1	D0	Reserved	ISPFF	PPGO	MPBUSY

Bits	Description	
[31:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7]	**D3**	**ISP DATA 3 Flag (Read Only)** This bit is set when FMC_MPDAT3 is written and auto-clear to 0 when the FMC_MPDAT3 data is programmed to flash complete. 0 = FMC_MPDAT3 register is empty, or program to flash complete. 1 = FMC_MPDAT3 register has been written, and not program to flash complete.
[6]	**D2**	**ISP DATA 2 Flag (Read Only)** This bit is set when FMC_MPDAT2 is written and auto-clear to 0 when the FMC_MPDAT2 data is programmed to flash complete. 0 = FMC_MPDAT2 register is empty, or program to flash complete. 1 = FMC_MPDAT2 register has been written, and not program to flash complete.
[5]	**D1**	**ISP DATA 1 Flag (Read Only)** This bit is set when FMC_MPDAT1 is written and auto-clear to 0 when the FMC_MPDAT1 data is programmed to flash complete. 0 = FMC_MPDAT1 register is empty, or program to flash complete. 1 = FMC_MPDAT1 register has been written, and not program to flash complete.
[4]	**D0**	**ISP DATA 0 Flag (Read Only)** This bit is set when FMC_MPDAT0 is written and auto-clear to 0 when the FMC_MPDAT0 data is programmed to flash complete. 0 = FMC_MPDAT0 register is empty, or program to flash complete. 1 = FMC_MPDAT0 register has been written, and not program to flash complete.
[3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[2]	**ISPFF**	**ISP Fail Flag (Read Only)** 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) Page Erase command at LOCK mode with ICE connection (5) Erase or Program command at brown-out detected (6) Destination address is illegal, such as over an available range. (7) Invalid ISP commands
[1]	**PPGO**	**ISP Multi-program Status (Read Only)** 0 = ISP multi-word program operation is not active. 1 = ISP multi-word program operation is in progress.
[0]	**MPBUSY**	**ISP Multi-word Program Busy Flag (Read Only)** Write 1 to start ISP Multi-Word program operation and this bit will be cleared to 0 by hardware automatically when ISP Multi-Word program operation is finished. This bit is the mirror of ISPGO(FMC_ISPTRG[0]). 0 = ISP Multi-Word program operation is finished. 1 = ISP Multi-Word program operation is progressed.

ISP Multi-Word Program Address Register (FMC_MPADDR)

Register	Offset	R/W	Description	Reset Value
FMC_MPADDR	FMC_BA+0xC4	R	ISP Multi-Word Program Address Status Register	0x0000_0000

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

Bits	Description	
[31:0]	**MPADDR**	**ISP Multi-word Program Address** MPADDR is the address of ISP multi-word program operation when ISPGO flag is 1. MPADDR will keep the final ISP address when ISP multi-word program is complete.

6.5 General Purpose I/O (GPIO)

6.5.1 Overview

The NPCA121 series device has up to 58 General Purpose I/O (GPIO) pins, grouped in 4 ports PA, PB, PC and PD. Port PA, PC and PD each has 16 pins, and there are 13 pins in Port PB.

All the GPIO pins are multi-functional pins, in that they can be I/O pins or they can work as alternate function pins. Each pin can be individually configured. Pin function are defined in MFP registers, for example PA0~7 pin functions are defined in SYS_GPA_MFPL register.

When working as an I/O pin, each pin can be configured by software in several modes:

- Input
- Push-pull Output
- Open-Drain Output
- Quasi-bidirectional

After a power-on or reset event, all GPIO pins' default working mode are determined by CIOINI bit (CONFIG0[10]) except PA.8. PA.8 pin default I/O mode is determined by GPA8_LOW bit (CONFIG0[11]). Every I/O pin has a weak pull-up resistor with value ~50 kΩ when I/O pin configured as quasi-bidirectional output low.

6.5.2 Features

- Four I/O modes:
  - Quasi-bidirectional mode
  - Push-Pull Output mode
  - Open-Drain Output mode
  - Input only with high impendence mode
- TTL/Schmitt trigger input capability
- I/O pin can be configured as interrupt source with edge/level trigger option
- Supports High Drive and High Slew Rate I/O mode
- CIOINI bit (CONFIG0[10]) configures all GPIO pins' default I/O mode except PA.8 after power-on or reset:
  - CIOINI = 0: Quasi-bidirectional mode,
  - CIOINI = 1: input mode.
- GPA8_LOW (CONFIG[11]) configures PA.8 pin's default I/O mode after power-on or reset:
  - GPA8_LOW = 0: Push-Pull mode and output low,
  - GPA8_LOW = 1: PA.8 follows CIOINI setting.
- I/O pin internal pull-up only available in Quasi-bidirectional I/O mode
- Enabling the pin interrupt function will also enable the wake-up function
- PB0 ~ PB4, PB7 ~ PB9, PB13 ~ PB15, PC2 ~ PC15 and PD0 ~ PD15 support 5V-tolerance functions

6.5.3 Block Diagram

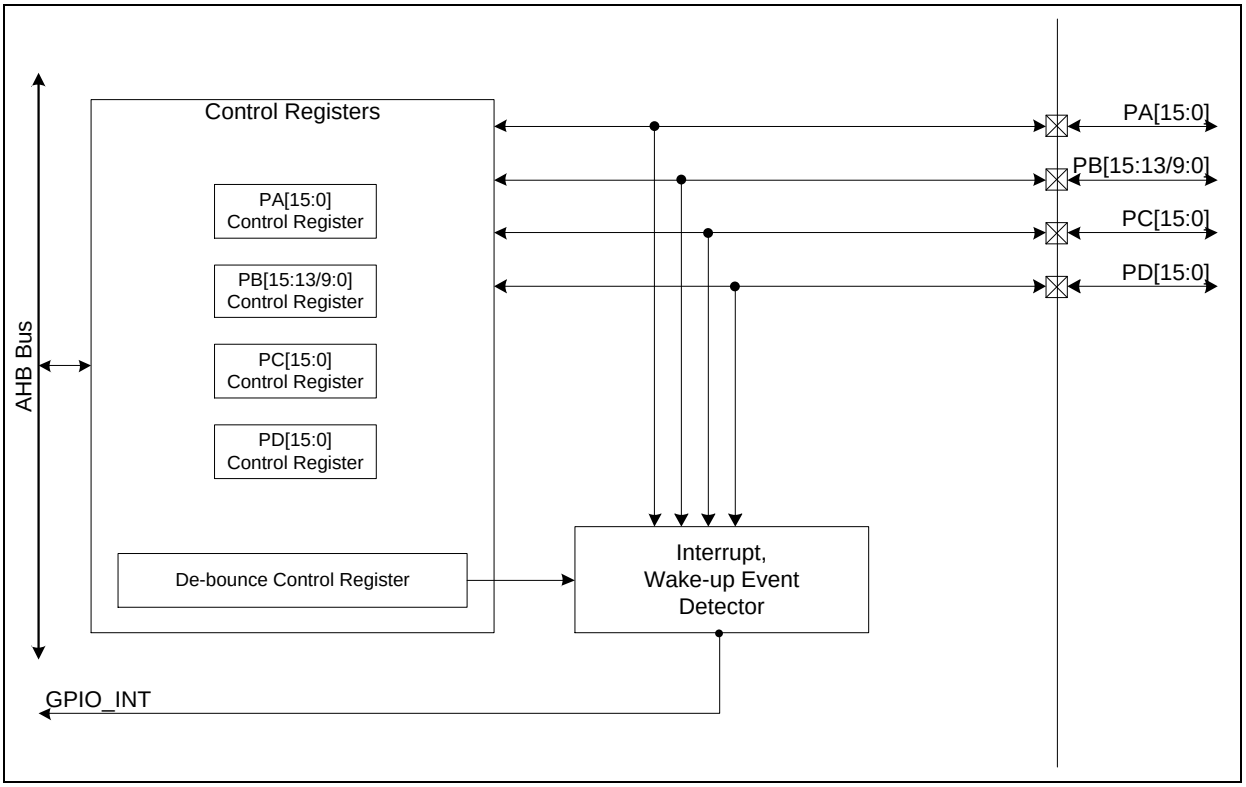


Figure 6.5-1 GPIO Controller Block Diagram

6.5.4 Basic Configuration

- Reset configuration
  - Writing 1 to GPIORST bit (SYS_IPRST1[1]) resets GPIO controller
- Pin configuration

Group	Pin Name	GPIO	MFP
INT0	INT0	PA.13	MFP2
		PA.15	MFP1
INT1	INT1	PC.5	MFP1
		PD.7	MFP2
INT2	INT2	PC.6	MFP1
		PD.11	MFP2
INT3	INT3	PD.0	MFP1
		PD.12	MFP2
INT4	INT4	PD.1	MFP1
INT5	INT5	PD.10	MFP1

6.5.5 Functional Description

6.5.5.1 Input Mode

Writing 0b00 into MODEn bits (Px_MODE[2n+1:2n]) puts the corresponding Px.n pin in Input mode, and the pin will be in tri-state (high impedance). The input pin's status is reflected in PX PIN[n] bit. For example if PA.0 is an input pin, the input level can be read by reading PA_PIN register, and PA_PIN[0] has the input value for PA.0 pin.

6.5.5.2 Push-pull Output Mode

Writing 0b01 into MODEn bits (Px_MODE[2n+1:2n]) puts the corresponding Px.n pin in in Push-pull Output mode, and the pin supports digital output function with source/sink current capability. As shown in Figure 6.5-2, the bit value in the corresponding DOUT (Px_DOUT[n]) is driven on the pin.

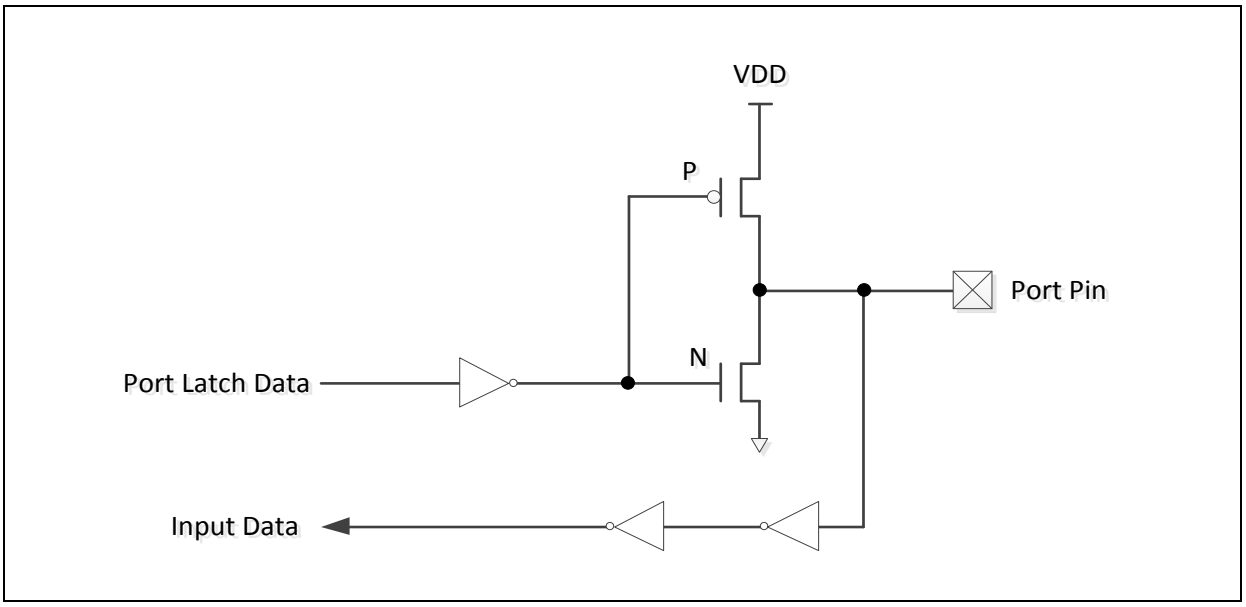


Figure 6.5-2 Push-Pull Output

6.5.5.3 Open-drain Mode

Writing 0b10 to MODEn bits (Px_MODE[2n+1:2n]) configures the corresponding Px.n pin as Open-drain mode I/O pin. External pull-up register is required to drive high state. If DOUT (Px_DOUT[n]) bit is 0, the pin drives low. If DOUT (Px_DOUT[n]) bit is 1, the pin drives high assuming there is external pull high.

Open-drain Mode I/O function is shown in Figure 6.5-3.

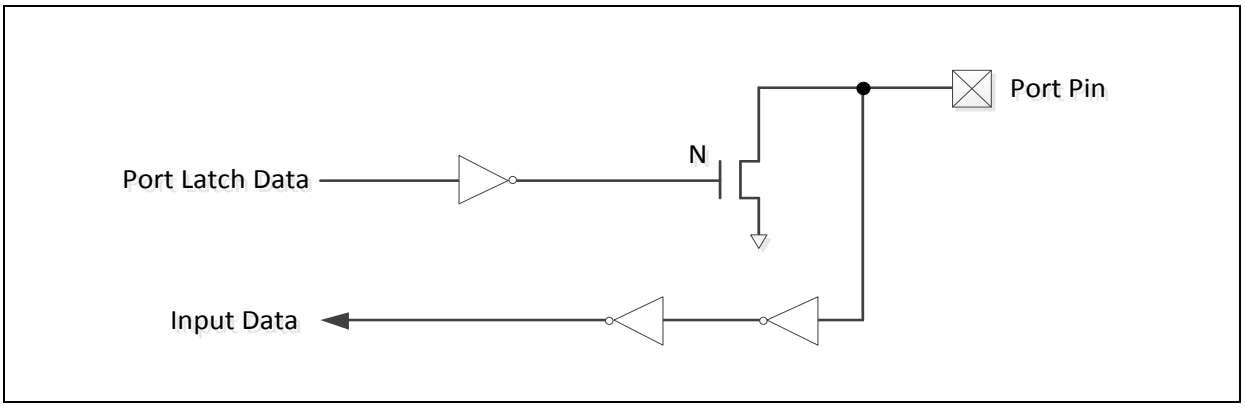


Figure 6.5-3 Open-Drain Output

6.5.5.4 Quasi-bidirectional Mode

Writing 0b11 to MODEn bits (Px_MODE[2n+1:2n]) configures the corresponding Px.n pin as Quasi-bidirectional mode I/O pin. Under this mode, the I/O pin supports digital output and input function at the same time however the max source current is only less than 100uA.

For input operation, before the read operation is performed the DOUT (Px_DOUT[n]) bit must be set to 1. For output operation, if DOUT (Px_DOUT[n]) bit is 0, the pin outputs low to the pin. If DOUT (Px_DOUT[n]) bit is 1, the device will drives high and also check the pin value. If pin value is high, there will be no action taken. If pin state is low, then the device will drive strong high for 2 clock cycles 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 30~65 μA.

Figure 6.5-4 shows the Quasi-bidirectional Mode I/O pin architecture.

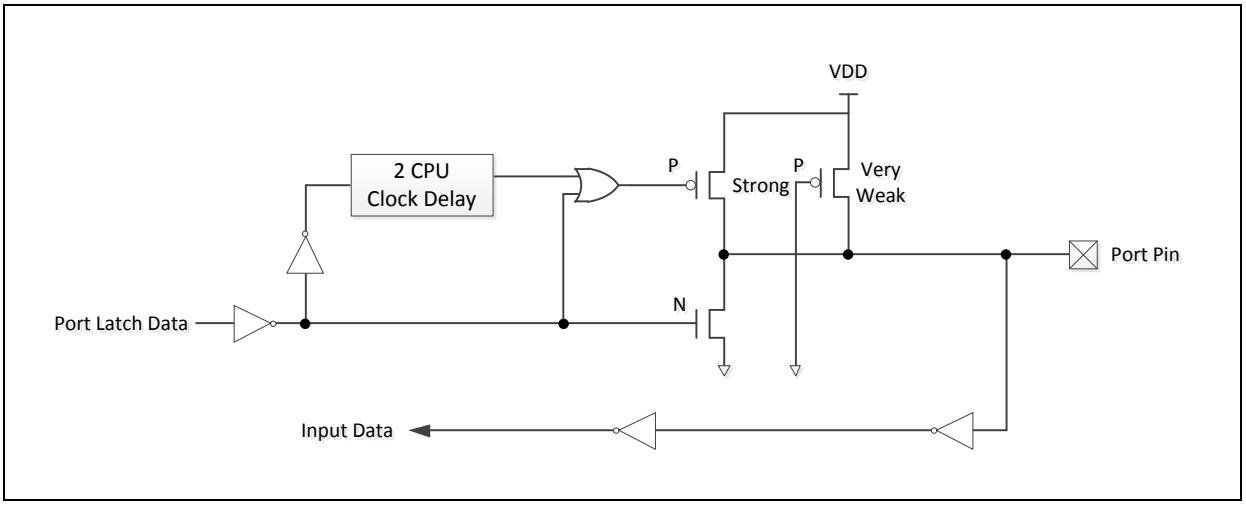


Figure 6.5-4 Quasi-Bidirectional I/O Mode

6.5.5.5 GPIO Interrupt and Wake-up Function

Each GPIO pin can be configured to generate interrupt by configuring its RHIEN (Px_INTEN[n+16])/ FLIEN (Px_INTEN[n]) bit and TYPE (Px_INTTYPE[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.

The GPIO pins can also be the chip wake-up source from Idle/Power-down state. The configuration of wake-up trigger condition is the same as GPIO interrupt configuration.

6.5.5.6 GPIO De-bounce Function

GPIO de-bounce function can be used to sample interrupt input and prevent unexpected interrupt due to noise. GPIO de-bounce function only applies to edge trigger interrupts, including falling edge trigger, rising edge trigger and both rising and falling edge trigger.

Writing 1 to Px_DBEN bit enables the de-bounce function for the corresponding pin. The de-bounce clock source can be from HCLK or LIRC (10kHz) by configuring DBCLKSRC (Px_DBCTL[4]) register. DBCLKSEL (Px_DBCTL[3:0]) register configures the interrupt sampling rate.

Figure 6.5-5 shows the triggering condition for a GPIO rising edge trigger interrupt. The interval of time between the two valid sample signal is determined by DBCLKSRC (Px_DBCTL[4]) and DBCLKSEL (Px_DBCTL[3:0]). Each valid data from GPIO pin need to be sampled twice. For rising edge trigger:

- if the pin status was low before DBEN (Px_DBEN) is enabled, an interrupt will be generated when a valid high pin status is met.
- If the pin status was high before DBEN (Px_DBEN) is enabled, an interrupt can only happen after first a valid low status detected, and then a valid high status detected.

For falling edge trigger, the opposite triggering condition is shown in Figure 6.5-6.

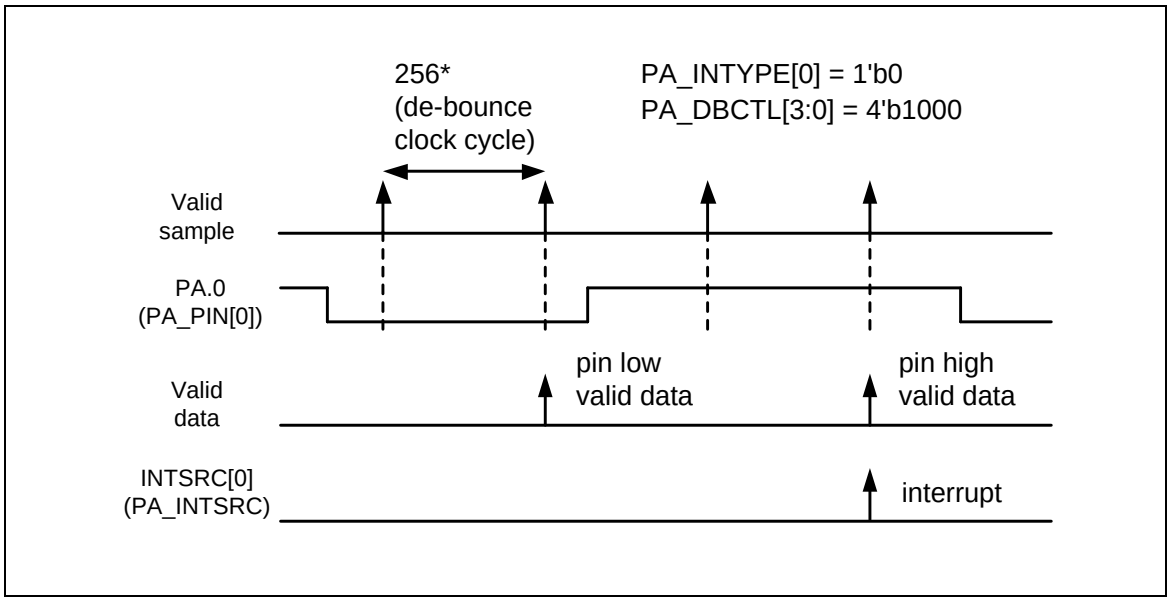


Figure 6.5-5 GPIO Rising Edge Trigger Interrupt

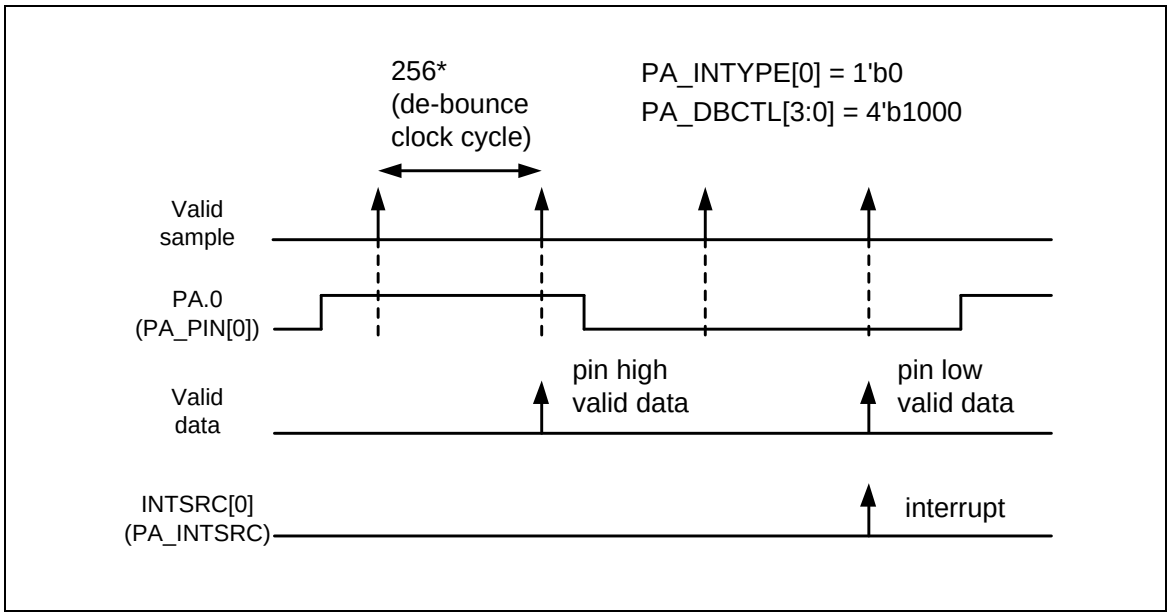


Figure 6.5-6 GPIO Falling Edge Trigger Interrupt

Table 6.5.5-1 shows the pre-condition for de-bounce support.

System Status	DBEN	DBCLKSRC	Description
Normal Mode / Idle Mode	0	0	No de-bounce function
		1	No de-bounce function
	1	0	De-bounce function using HCLK
		1	De-bounce function using LIRC (10 kHz)
Power Down Mode	0	0	No de-bounce function
		1	No de-bounce function
	1	0	No de-bounce function
		1	De-bounce function using LIRC (10 kHz)

Table 6.5.5-1 De-Bounce Function Setting Table

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 = 0x4000_4000**				
PA_MODE	GPIO_BA+0x000	R/W	PA I/O Mode Control	0xXXXX_XXXX
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_FXFF
PA_DATMSK	GPIO_BA+0x00C	R/W	PA Data Output Write Mask	0x0000_0000
PA_PIN	GPIO_BA+0x010	R	PA Pin Value	0x0000_XXXX
PA_DBEN	GPIO_BA+0x014	R/W	PA De-Bounce Enable Control Register	0x0000_0000
PA_INTTYPE	GPIO_BA+0x018	R/W	PA Interrupt Trigger Type Control	0x0000_0000
PA_INTEN	GPIO_BA+0x01C	R/W	PA Interrupt Enable Control Register	0x0000_0000
PA_INTSRC	GPIO_BA+0x020	R/W	PA Interrupt Source Flag	0x0000_0000
PA_SMTEN	GPIO_BA+0x024	R/W	PA Input Schmitt Trigger Enable Register	0x0000_0000
PA_SLEWCTL	GPIO_BA+0x028	R/W	PA High Slew Rate Control Register	0x0000_0000
PA_PUSEL	GPIO_BA+0x030	R/W	PA Pull-up and Pull-down Selection Register	0x0000_0000
PB_MODE	GPIO_BA+0x040	R/W	PB I/O Mode Control	0xXX0X_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_F3FF
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 Register	0x0000_0000
PB_INTTYPE	GPIO_BA+0x058	R/W	PB Interrupt Trigger Type Control	0x0000_0000
PB_INTEN	GPIO_BA+0x05C	R/W	PB Interrupt Enable Control Register	0x0000_0000
PB_INTSRC	GPIO_BA+0x060	R/W	PB Interrupt Source Flag	0x0000_0000
PB_SMTEN	GPIO_BA+0x064	R/W	PB Input Schmitt Trigger Enable Register	0x0000_0000
PB_SLEWCTL	GPIO_BA+0x068	R/W	PB High Slew Rate Control Register	0x0000_0000
PB_PUSEL	GPIO_BA+0x070	R/W	PB Pull-up and Pull-down Selection Register	0x0000_0000
PC_MODE	GPIO_BA+0x080	R/W	PC I/O Mode Control	0xXXXX_XXXX
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_FFFF
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 Register	0x0000_0000

Register	Offset	R/W	Description	Reset Value
GPIO Base Address: **GPIO_BA = 0x4000_4000**				
PC_INTTYPE	GPIO_BA+0x098	R/W	PC Interrupt Trigger Type Control	0x0000_0000
PC_INTEN	GPIO_BA+0x09C	R/W	PC Interrupt Enable Control Register	0x0000_0000
PC_INTSRC	GPIO_BA+0x0A0	R/W	PC Interrupt Source Flag	0x0000_0000
PC_SMTEN	GPIO_BA+0x0A4	R/W	PC Input Schmitt Trigger Enable Register	0x0000_0000
PC_SLEWCTL	GPIO_BA+0x0A8	R/W	PC High Slew Rate Control Register	0x0000_0000
PC_PUSEL	GPIO_BA+0x0B0	R/W	PC Pull-up and Pull-down Selection Register	0x0000_0000
PD_MODE	GPIO_BA+0x0C0	R/W	PD I/O Mode Control	0xXXXX_XXXX
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_FFFF
PD_DATMSK	GPIO_BA+0x0CC	R/W	PD Data Output Write Mask	0x0000_0000
PD_PIN	GPIO_BA+0x0D0	R	PD Pin Value	0x0000_XXXX
PD_DBEN	GPIO_BA+0x0D4	R/W	PD De-Bounce Enable Control Register	0x0000_0000
PD_INTTYPE	GPIO_BA+0x0D8	R/W	PD Interrupt Trigger Type Control	0x0000_0000
PD_INTEN	GPIO_BA+0x0DC	R/W	PD Interrupt Enable Control Register	0x0000_0000
PD_INTSRC	GPIO_BA+0x0E0	R/W	PD Interrupt Source Flag	0x0000_0000
PD_SMTEN	GPIO_BA+0x0E4	R/W	PD Input Schmitt Trigger Enable Register	0x0000_0000
PD_SLEWCTL	GPIO_BA+0x0E8	R/W	PD High Slew Rate Control Register	0x0000_0000
PD_PUSEL	GPIO_BA+0x0F0	R/W	PD Pull-up and Pull-down Selection Register	0x0000_0000
GPIO_DBCTL	GPIO_BA+0x440	R/W	Interrupt De-bounce Control Register	0x0000_0020
PAn_PDIO **n=0,1..15**	GPIO_BA+0x800+ (0x04 * n)	R/W	GPIO PA.n Pin Data Input/Output Register	0x0000_000X
PBn_PDIO **n=0,1..9,13..15**	GPIO_BA+0x840+ (0x04 * n)	R/W	GPIO PB.n Pin Data Input/Output Register	0x0000_000X
PCn_PDIO **n=0,1..15**	GPIO_BA+0x880+ (0x04 * n)	R/W	GPIO PC.n Pin Data Input/Output Register	0x0000_000X
PDn_PDIO **n=0,1..15**	GPIO_BA+0x8C0+ (0x04 * n)	R/W	GPIO PD.n Pin Data Input/Output Register	0x0000_000X

Note:

1. Any register not listed here is reserved and must not be written. The result of a read operation on these bits is undefined.
2. The reserved register fields that listed in register description must be written to their reset value. Writing reserved fields with other than reset values may produce undefined results.

6.5.7 Register Description

Port A-D 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	0xXXXX_XXXX
PB_MODE	GPIO_BA+0x040	R/W	PB I/O Mode Control	0xXX0X_XXXX
PC_MODE	GPIO_BA+0x080	R/W	PC I/O Mode Control	0xXXXX_XXXX
PD_MODE	GPIO_BA+0x0C0	R/W	PD I/O Mode Control	0xXXXX_XXXX

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-D 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. **Note 1:** The reset value of this field is defined by CIOINI (CONFIG0[10]) except PA_MODE[17:16]. If CIOINI is set to 0, the reset value is 0xFFFF_FFFF and all pins will be quasi-bidirectional mode after chip power-on or reset period. If CIOINI is set to 1, the reset value is 0x0000_0000 and all pins will be input mode after chip power-on or reset period. **Note 2:** The reset value of PA_MODE[17:16] field is defined by GPA8_LOW (CONFIG0[11]). If GPA8_LOW is set to 0, the reset value of PA_MODE[17:16] is 0x1 and PA.8 will be push-pull output mode after chip power-on or reset period. If GPA8_LOW is set to 1, the reset value of PA_MODE[17:16] follows CIOINI setting. **Note 3:** Max. n=15 for port A/C/D n=0..9, 13, 14, 15 for port B

Port A-D 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

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[n]**	**Port A-D Pin[n] Digital Input Path Disable Control** Each of these bits is used to control if the digital input path of corresponding Px.n pin is disabled. If input is analog signal, users can disable Px.n digital input path to avoid input current leakage. 0 = Px.n digital input path Enabled. 1 = Px.n digital input path Disabled (digital input tied to low). **Note:** Max. n=15 for port A/C/D n=0..9, 13, 14, 15 for port B
[15:0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

Port A-D Data Output Value (Px_DOUT)

Register	Offset	R/W	Description	Reset Value
PA_DOUT	GPIO_BA+0x008	R/W	PA Data Output Value	0x0000_FXFF
PB_DOUT	GPIO_BA+0x048	R/W	PB Data Output Value	0x0000_F3FF
PC_DOUT	GPIO_BA+0x088	R/W	PC Data Output Value	0x0000_FFFF
PD_DOUT	GPIO_BA+0x0C8	R/W	PD Data Output Value	0x0000_FFFF

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[n] n=0,1..15	**DOUT[n]**	**Port A-D Pin[n] Output Value** Each of these bits controls the status of a Px.n pin when the Px.n is configured as Push-pull output, Open-drain output or Quasi-bidirectional mode. 0 = Px.n will drive Low if the Px.n pin is configured as Push-pull output, Open-drain output or Quasi-bidirectional mode. 1 = Px.n will drive High if the Px.n pin is configured as Push-pull output or Quasi-bidirectional mode. **Note 1:** The reset value of PA_DOUT[8] field is defined by GPA8_LOW (CONFIG0[11]). If GPA8_LOW is set to 0, the reset value of PA_DOUT[8] is 0 and PA.8 pin will be drive low state after chip power-on or reset period. If GPA8_LOW is set to 1, the reset value of PA_DOUT[8] is 1. **Note 2:** The reset value of PA_DOUT[11:8] is 111Xb in binary form, the reset value of PA_DOUT[8] is based on the setting of GPA_LOW (CONFIG0[11]). **Note 3:** Max. n=15 for port A/C/D n=0..9, 13, 14, 15 for port B

Port A-D 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

31	30	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. Any values read should be ignored. When writing to this field always write with reset value.
[n] n=0,1..15	**DATMSK[n]**	**Port A-D 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. **Note 1:** This function only protects the corresponding DOUT (Px_DOUT[n]) bit, and will not protect the corresponding PDIO (Pxn_PDIO[0]) bit. **Note 2:** Max. n=15 for port A/C/D n=0..9, 13, 14, 15 for port B

Port A-D Pin Value (Px_PIN)

Register	Offset	R/W	Description	Reset Value
PA_PIN	GPIO_BA+0x010	R	PA Pin Value	0x0000_XXXX
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_XXXX

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[n] n=0,1..15	**PIN[n]**	**Port A-D Pin[n] Pin Value** Each bit of the register reflects the actual status of the respective Px.n pin. If the bit is 1, it indicates the corresponding pin status is high; else the pin status is low. **Note 1:** The reset value of PB_PIN[15:8] is XXXX_00XXb in binary form. **Note 2:** Max. n=15 for port A/C/D n=0..9, 13, 14, 15 for port B

Port A-D De-bounce Enable Control Register (Px_DBEN)

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

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[n] n=0,1..15	**DBEN[n]**	**Port A-D 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. **Note:** Max. n=15 for port A/C/D n=0..9, 13, 14, 15 for port B

Port A-D Interrupt Type Control (Px_INTTYPE)

Register	Offset	R/W	Description	Reset Value
PA_INTTYPE	GPIO_BA+0x018	R/W	PA Interrupt Trigger Type Control	0x0000_0000
PB_INTTYPE	GPIO_BA+0x058	R/W	PB Interrupt Trigger Type Control	0x0000_0000
PC_INTTYPE	GPIO_BA+0x098	R/W	PC Interrupt Trigger Type Control	0x0000_0000
PD_INTTYPE	GPIO_BA+0x0D8	R/W	PD Interrupt 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. Any values read should be ignored. When writing to this field always write with reset value.
[n] n=0,1..15	**TYPE[n]**	**Port A-D 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. **Note:** Max. n=15 for port A/C/D n=0..9, 13, 14, 15 for port B

Port A-D Interrupt Enable Control Register (Px_INTEN)

Register	Offset	R/W	Description	Reset Value
PA_INTEN	GPIO_BA+0x01C	R/W	PA Interrupt Enable Control Register	0x0000_0000
PB_INTEN	GPIO_BA+0x05C	R/W	PB Interrupt Enable Control Register	0x0000_0000
PC_INTEN	GPIO_BA+0x09C	R/W	PC Interrupt Enable Control Register	0x0000_0000
PD_INTEN	GPIO_BA+0x0DC	R/W	PD Interrupt Enable Control Register	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[n]**	**Port A-D 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. **Note:** Max. n=15 for port A/C/D n=0..9, 13, 14, 15 for port B
[n] n=0,1..15	**FLIEN[n]**	**Port A-D 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. **Note:**

		Max. n=15 for port A/C/D n=0..9, 13, 14, 15 for port B

Port A-D Interrupt Source Flag (Px_INTSRC)

Register	Offset	R/W	Description	Reset Value
PA_INTSRC	GPIO_BA+0x020	R/W	PA Interrupt Source Flag	0x0000_0000
PB_INTSRC	GPIO_BA+0x060	R/W	PB Interrupt Source Flag	0x0000_0000
PC_INTSRC	GPIO_BA+0x0A0	R/W	PC Interrupt Source Flag	0x0000_0000
PD_INTSRC	GPIO_BA+0x0E0	R/W	PD Interrupt Source Flag	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[n] n=0,1..15	**INTSRC[n]**	**Port A-D Pin[n] Interrupt Source Flag** Write Operation : 0 = No action. 1 = Clear the corresponding pending interrupt. Read Operation : 0 = No interrupt at Px.n. 1 = Px.n generates an interrupt. **Note 1:** The reset value of PB_INTSRC[15:8] is 0XXX_00XXb in binary form. **Note 2:** Max. n=15 for port A/C/D n=0..9, 13, 14, 15 for port B

Port A-D Input Schmitt Trigger Enable Register (Px_SMTEN)

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[n] n=0,1..15	**SMTEN[n]**	**Port A-D Pin[n] Input Schmitt Trigger Enable Bit** 0 = Px.n input schmitt trigger function Disabled. 1 = Px.n input schmitt trigger function Enabled. **Note:** Max. n=15 for port A/C/D n=0..9, 13, 14, 15 for port B

Port A-D High Slew Rate Control Register (Px_SLEWCTL)

Register	Offset	R/W	Description	Reset Value
PA_SLEWCTL	GPIO_BA+0x028	R/W	PA High Slew Rate Control Register	0x0000_0000
PB_SLEWCTL	GPIO_BA+0x068	R/W	PB High Slew Rate Control Register	0x0000_0000
PC_SLEWCTL	GPIO_BA+0x0A8	R/W	PC High Slew Rate Control Register	0x0000_0000
PD_SLEWCTL	GPIO_BA+0x0E8	R/W	PD High Slew Rate Control Register	0x0000_0000

31	30	29	28	27	26	25	24
HSREN15		HSREN14		HSREN13		HSREN12	
23	22	21	20	19	18	17	16
HSREN11		HSREN10		HSREN9		HSREN8	
15	14	13	12	11	10	9	8
HSREN07		HSREN6		HSREN5		HSREN4	
7	6	5	4	3	2	1	0
HSREN03		HSREN2		HSREN1		HSREN0	

Bits	Description	
[2n+1:2n] n=0,1..15	**HSRENn**	**Port A-D Pin[n] High Slew Rate Control** 00 = Px.n output with normal slew rate mode. 01 = Px.n output with high slew rate mode. 10 = Px.n output with fast slew rate mode. 11 = Reserved. Do not use. **Note:** Max. n=15 for port A/C/D n=0..9, 13, 14, 15 for port B

Port A-D Pull-up and Pull-down Selection Register (Px_PUSEL)

Register	Offset	R/W	Description	Reset Value
PA_PUSEL	GPIO_BA+0x030	R/W	PA Pull-up and Pull-down Selection Register	0x0000_0000
PB_PUSEL	GPIO_BA+0x070	R/W	PB Pull-up and Pull-down Selection Register	0x0000_0000
PC_PUSEL	GPIO_BA+0x0B0	R/W	PC Pull-up and Pull-down Selection Register	0x0000_0000
PD_PUSEL	GPIO_BA+0x0F0	R/W	PD Pull-up and Pull-down Selection Register	0x0000_0000

31	30	29	28	27	26	25	24
PUSEL15		PUSEL14		PUSEL13		PUSEL12	
23	22	21	20	19	18	17	16
PUSEL11		PUSEL10		PUSEL9		PUSEL8	
15	14	13	12	11	10	9	8
PUSEL7		PUSEL6		PUSEL5		PUSEL4	
7	6	5	4	3	2	1	0
PUSEL3		PUSEL2		PUSEL1		PUSEL0	

Bits	Description	
[2n+1:2n] n=0,1..15	**PUSELn**	**Port A-D Pin[n] Pull-up and Pull-down Enable Register** Determine each I/O Pull-up/pull-down of Px.n pins. 00 = Px.n pull-up and pull- down disable. 01 = Px.n pull-up enable. 10 = Px.n pull-down enable. 11 = Px.n pull-up and pull-down disable. **Note 1:** Basically, the pull-up control and pull-down control has following behavior limitation The independent pull-up control register only valid when MODEn set as tri-state and open-drain mode The independent pull-down control register only valid when MODEn set as tri-state mode When both pull-up and pull-down is set as 1 at “tri-state” mode, keep I/O in tri-state mode **Note 2:** Max. n=15 for port A/C/D n=0..9, 13, 14, 15 for port B

Interrupt De-bounce Control Register (GPIO_DBCTL)

Register	Offset	R/W	Description	Reset Value
GPIO_DBCTL	GPIO_BA+0x440	R/W	Interrupt De-bounce Control Register	0x0000_0020

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

Bits	Description	
[31:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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 10 kHz internal low speed RC oscillator (LIRC).

Bits	Description	
[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/Output Register (Pxn_PDIO)

Register	Offset	R/W	Description	Reset Value
PAn_PDIO **n=0,1..15**	GPIO_BA+0x800+ (0x04 * n)	R/W	GPIO PA.n Pin Data Input/Output Register	0x0000_000X
PBn_PDIO **n=0,1..9,13..15**	GPIO_BA+0x840+ (0x04 * n)	R/W	GPIO PB.n Pin Data Input/Output Register	0x0000_000X
PCn_PDIO **n=0,1..15**	GPIO_BA+0x880+ (0x04 * n)	R/W	GPIO PC.n Pin Data Input/Output Register	0x0000_000X
PDn_PDIO **n=0,1..15**	GPIO_BA+0x8C0+ (0x04 * n)	R/W	GPIO PD.n Pin Data Input/Output 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							**PDIO**

Bits	Description	
[31:1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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]). **Note 1:** The writing operation will not be affected by register DATMSK (Px_DATMSK[n]). **Note 2:** The reset value of PAn_PDIO[3:0], PBn_PDIO[3:0], PCn_PDIO[3:0] and PDn_PDIO[3:0] are 000Xb in binary form. **Note 3:** Max. n=15 for port A/C/D n=0..9, 13, 14, 15 for port B

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 16 channels and each channel can perform transfer between memory and peripherals or between memory and memory.

6.6.2 Features

- Supports 16 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, ADC and PWM 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 on channel 0 and channel1
- Supports stride function from channel 0 to channel 5

6.6.3 Block Diagram

The block diagram about PDMA controller is shown as follows.

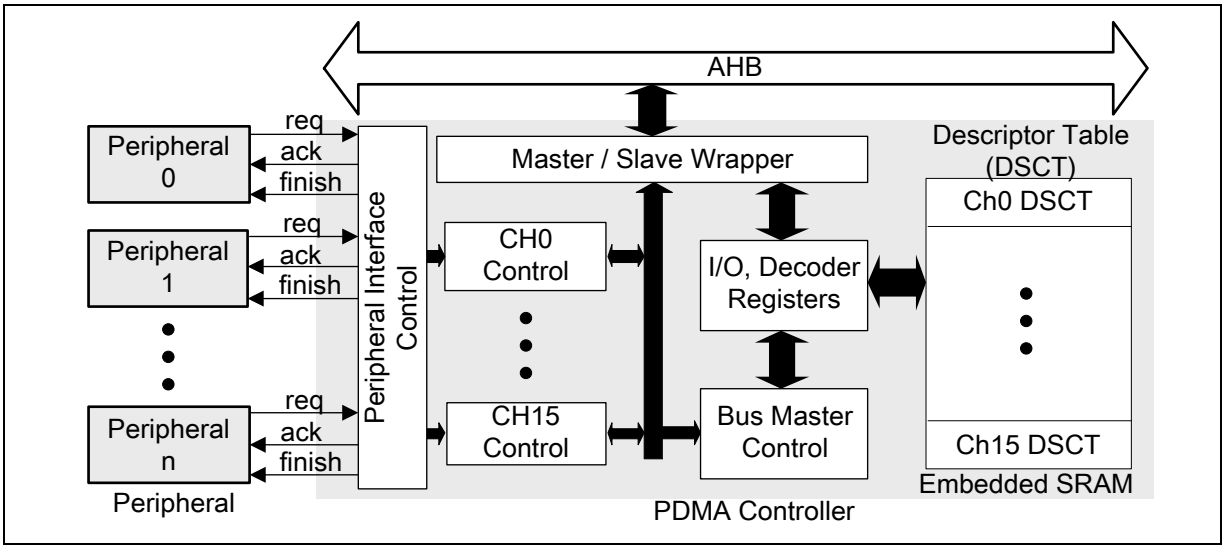


Figure 6.6-1 PDMA Controller Block Diagram

6.6.4 Basic Configuration

- Clock source configuration
  - Enable PDMA controller clock in PDMACKEN (CLK_AHBCLK [1]).
- Reset configuration
  - Reset PDMA controller in PDMARST (SYS_IPRST0[2]).

6.6.5 Functional Description

The PDMA controller transfers data from one address to another without CPU intervention. The PDMA controller supports 16 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. The Figure 6.6-2 shows the diagram of descriptor table (DSCT) data structure.

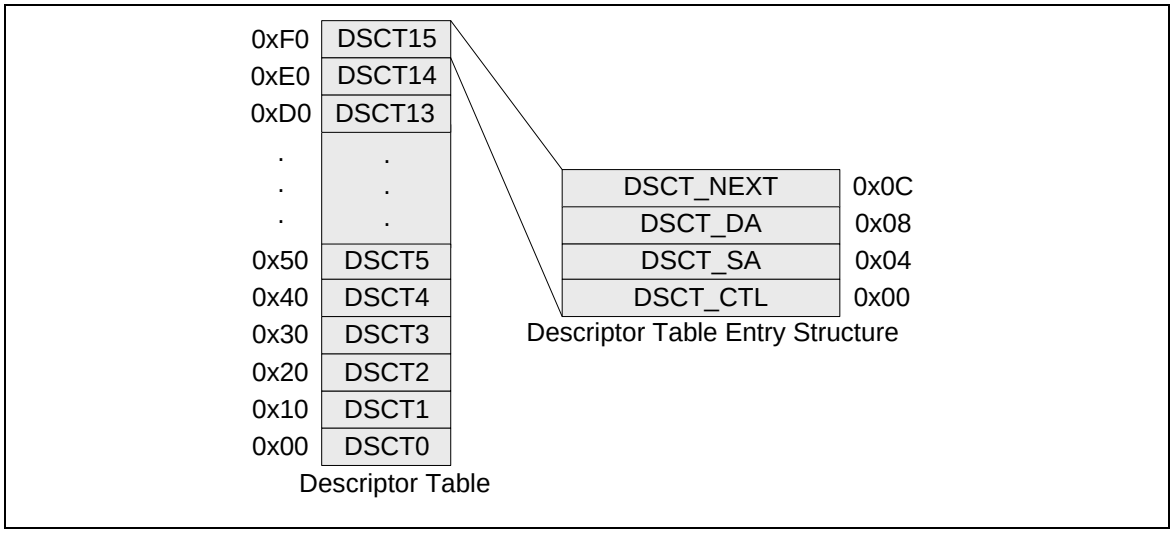


Figure 6.6-2 Descriptor Table Entry Structure

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).

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. The priority order is listed in Table 6.6.5-1.

PDMA_PRISET	Channel Number	Priority Setting	Arbitration Priority In Descending Order
1	15	Channel15, Fixed Priority	Highest
1	14	Channel14, Fixed Priority	---
---	---	---	---
1	0	Channel0, Fixed Priority	---
0	15	Channel15, Round-Robin Priority	---
0	14	Channel14, Round-Robin Priority	---
---	---	---	---
0	0	Channel0, Round-Robin Priority	Lowest

Table 6.6.5-1 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. 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), default setting is in idle state (OPMODE (PDMA_DSCTn_CTL[1:0]) = 0x0) 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 fill the transfer count TXCNT (PDMA_DSCTn_CTL[31:16]) 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 corresponding PDMA interrupt bit INTENn (PDMA_INTEN[15:0]) is enabled and the operation mode will be updated to idle state as shown in Figure 6.6-3. 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 corresponding PDMA interrupt bit is enabled.

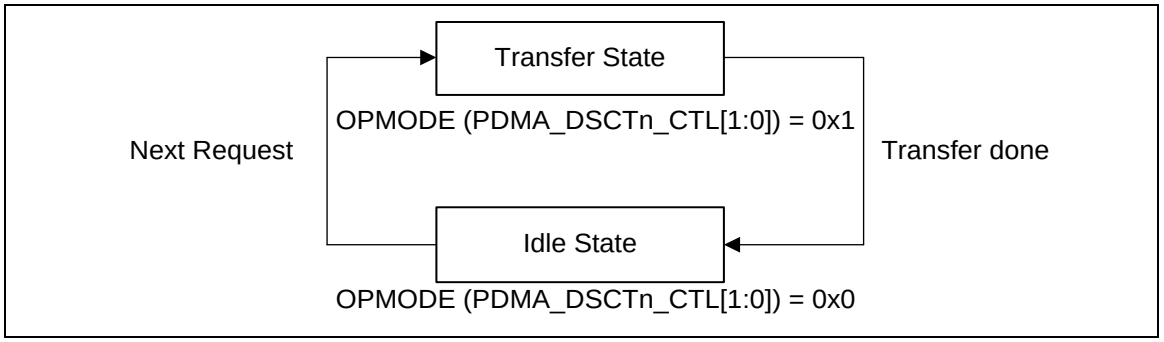


Figure 6.6-3 Basic Mode Finite State Machine

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 6.6-4. Through operation mode user can perform peripheral wrapper-around, multiple PDMA task or can be used for data transfer between varied locations in system memory instead of a set of contiguous locations. Scatter-gather mode only need a request to finish all table entries task until the last task with OPMODE (PDMA_DSCTn_CTL[1:0]) is idle state without ack, it also means scatter-gather mode can only be use to transfer data between memory to memory without handshaking.

In Scatter-Gather mode, the table 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[15:0]) flag will be asserted and if this bit is “1” TDIFn will not be active).

If channel 7 has been triggered, and the operation mode is in Scatter-Gather mode (OPMODE (PDMA_DSCTn_CTL[1:0]) = 0x2), the hardware will load the real PDMA information task from the address generated by adding PDMA_DSCTn_NEXT (link address) and PDMA_SCATBA (base address) registers. For example, base address is 0x2000_0000 (only MSB 16bits valid in PDMA_SCATBA), current link address is 0x0000_0100 (only LSB 16bits without last two bits [1:0] valid in PDMA_DSCTn_NEXT), then next DSCT entry start address is 0x2000_0100. Note that the address of descriptor tables must be in the same 64KB area.

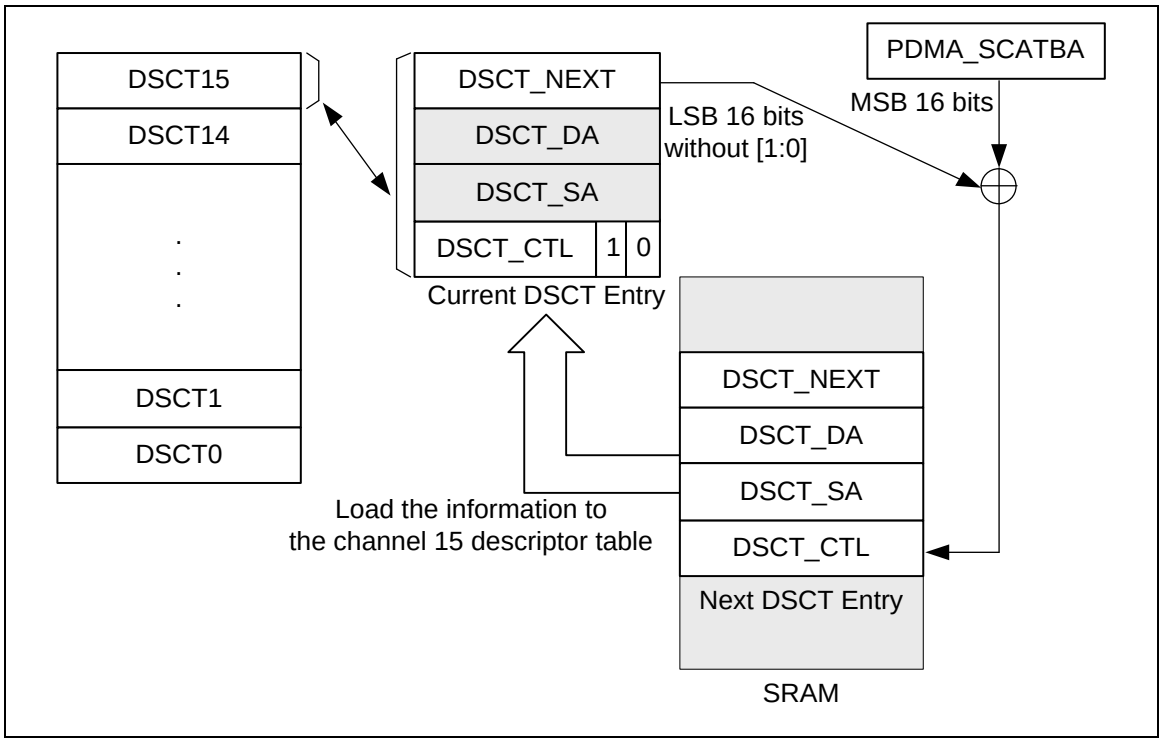


Figure 6.6-4 Descriptor Table Link List Structure

The above link list table operation is DSCT state in Scatter-Gather Mode as shown in Figure 6.6-5. 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.

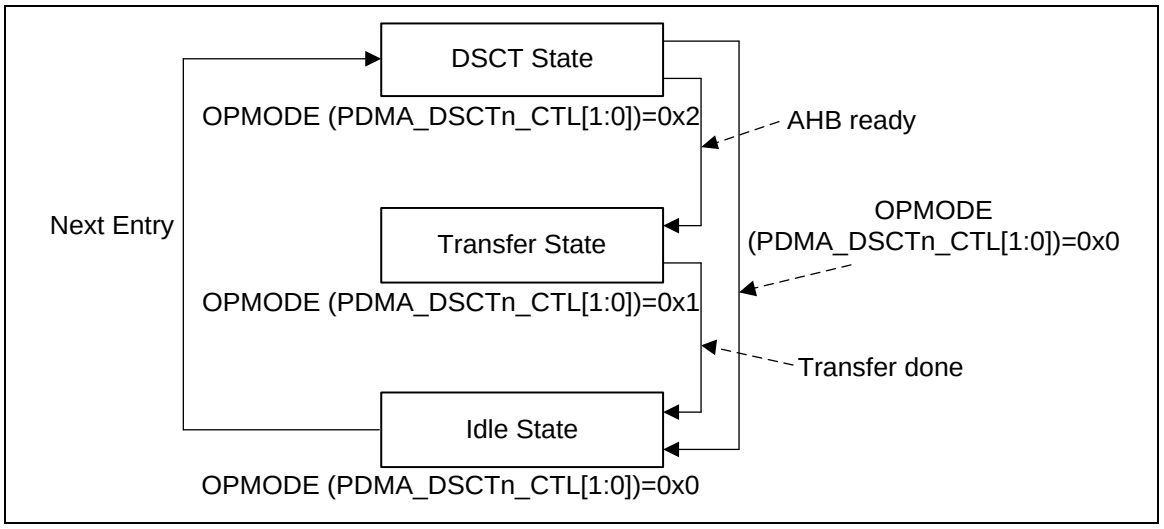


Figure 6.6-5 Scatter-Gather Mode Finite State Machine

6.6.5.3 Transfer Type

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

When PDMA controller operated in single transfer type, each transfer data needs one request signal for one transfer, after transferred data, TXCNT (PDMA_DSCTn_CTL[31:16]) will decrease 1. Transfer will finish until the TXCNT (PDMA_DSCTn_CTL[31:16]) decrease to 0. In this mode, the BURSIZE (PDMA_DSCTn_CTL[6:4]) is not useful to control the transfer size. The BURSIZE (PDMA_DSCTn_CTL[6:4]) will be fixed as one.

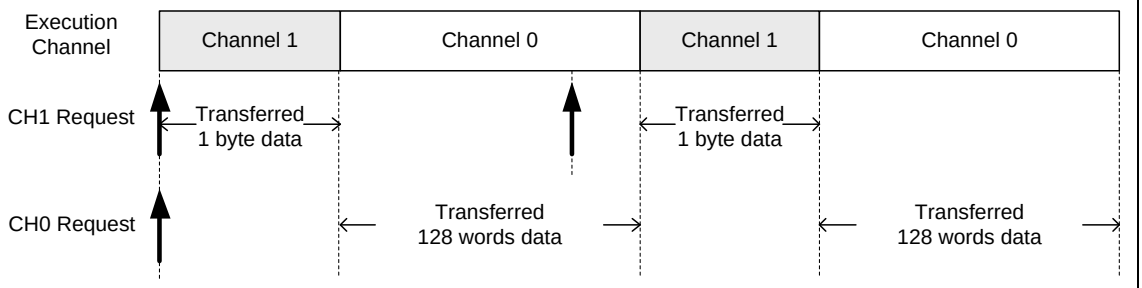
For the burst transfer type, PDMA controller transfers TXCNT (PDMA_DSCTn_CTL[31:16]) of data and need only one request signal. After transferred BURSIZE (PDMA_DSCTn_CTL[6:4]) of data, TXCNT (PDMA_DSCTn_CTL[31:16]) will decrease BURSIZE number. Transfer will done until the transfer count TXCNT (PDMA_DSCTn_CTL[31:16]) 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.

The Figure 6.6-6 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[31:16]) = 127. Channel 0 uses burst transfer type, BURSIZE (PDMA_DSCTn_CTL[6:4]) = 0 (128 transfers) and TXCNT (PDMA_DSCTn_CTL[31:16]) = 255. 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, so PDMA controller will only transfer one transfer data.
3. Then, PDMA controller turns to the channel 0 and loads channel 0's descriptor table. The channel 0 is burst transfer type and the burst size selected to 128. Therefore, PDMA controller will transfer 128 transfer data.
4. When channel 0 transfers 128 data, channel 1 gets another request signal, then after channel 0 finishes 128 transfer data, the PDMA controller will turn to channel 1 and transfer

next one data.

5. After channel 1 transfers data, PDMA controller switches to low priority channel 0 to continuous next 128 data transfer. If no channel 1 request receives, PDMA will start next channel 0, 128 data transfer.
6. PDMA controller will complete transfer when channel 0 finishes data transfer 256 times, and channel 1 finishes transferring 128 times.



	CH1	CH0
TXCNT (PDMA_DSCTn_CTL[31:16])	127	255
TXWIDTH (PDMA_DSCTn_CTL[13:12])	0 (1 byte)	2 (1 word)
BURSIZE (PDMA_DSCTn_CTL[6:4])	Non-useful	0 (128 transfers)
TXTYPE (PDMA_DSCTn_CTL[2])	1 (Single Transfer)	0 (Burst Transfer)

Figure 6.6-6 Example of Single Transfer Type and Burst Transfer Type in Basic Mode

6.6.5.4 Channel Time-out

Only PDMA 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 count up from 0 while the channel has received trigger signal from the peripheral.

The time-out counter is based on output of HCLK prescaler, which 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, time-out function disabled or chip enter power-down mode.

Figure 6.6-7 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.

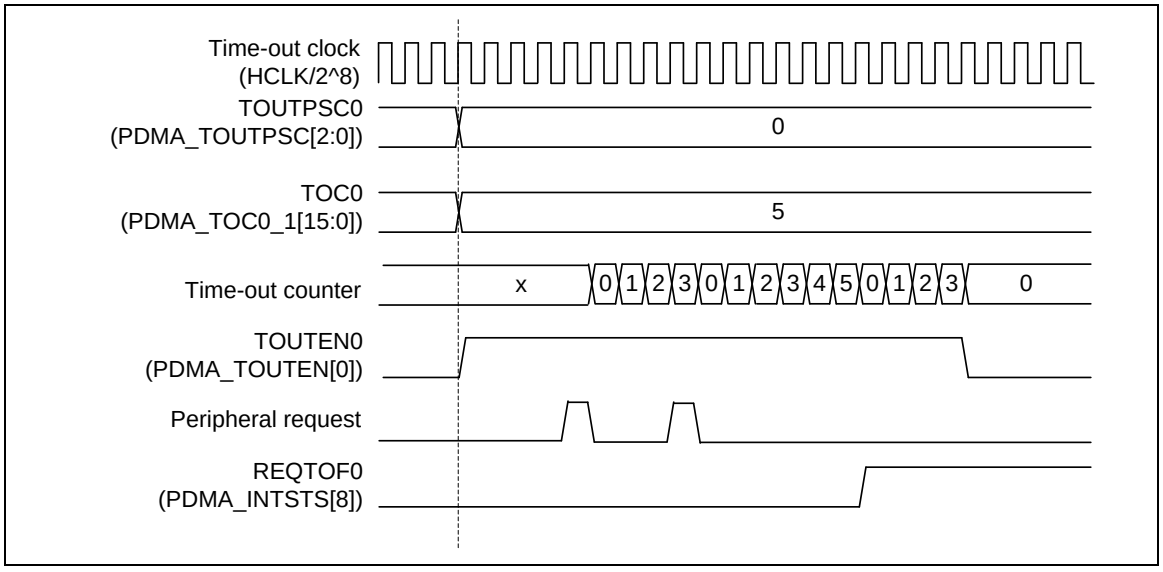


Figure 6.6-7 Example of PDMA Channel 0 Time-out Counter Operation

6.6.5.5 Stride Function

The PDMA support channel 0 to channel 5 six channels with stride function. The stride function can transfers data from one address to another address and can support block transfer with stride. When operating in stride function, the transfer address can be fixed or incremented successively.

Setting STRIDE_EN (PDMA_DSCTn_CTL[15]) to enable stride function, and then write a valid source address to the PDMA_DSCTn_SA register and a source address offset count to SASOL (PDMA_ASOCRn[15:0]) register, a destination address to the PDMA_DSCTn_DA register and a destination address offset count to DASOL (PDMA_ASOCRn[31:16]), and a transfer count to the TXCNT (PDMA_DSCTn_CTL) register and a stride transfer count to STC (PDMA_STCn[15:0]). Next, trigger the SWREQn (PDMA_SWREQ[5:0]). The DMA will continue the transfer until TXCNT (PDMA_DSCTn_CTL) count down to zero. The Figure 6.6-8 shows the block transfer relationship between source memory and destination memory. Stride function also supports peripheral to memory or memory to peripheral transfer.

Please note that stride mode must be operated in burst mode (TXTYPE (PDMA_DSCTx_CTL[2]) = 0), and the transfer count (TXCNT (PDMA_DSCTx_CTL[31:16])) should be less than burst size (BURSIZE (PDMA_DSCTx_CTL[6:4])), and STC (PDMA_STCRx[15:0]) must be greater than or equal to 1.

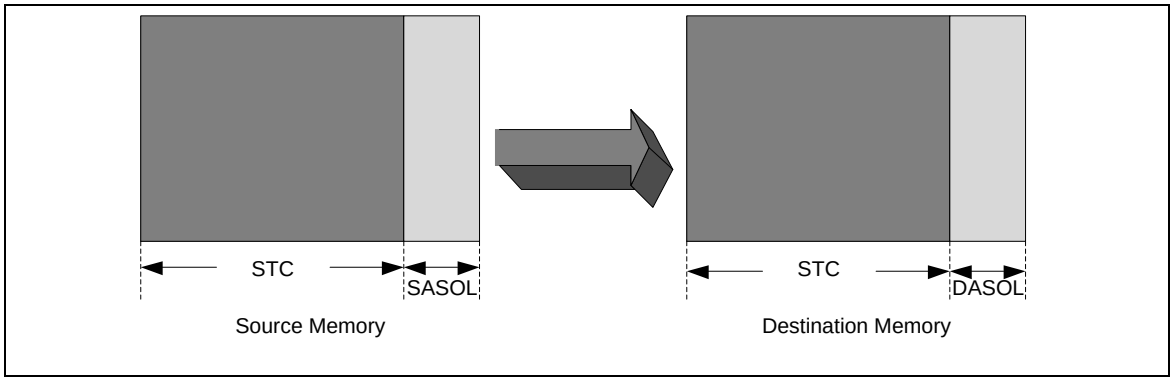


Figure 6.6-8 Stride Function Block Transfer

6.6.5.6 PDMA Controller Transfer Bandwidth

The PDMA fits into the memory system of the NPCA121 as a master on the AHB bus. Only a finite number of PDMA transactions can occur on the AHB per second, hence there is a maximum bandwidth possible for PDMA transfers. The PDMA controller takes multiple clock cycles to handle requests to/from peripherals or SRAM. The PDMA controller uses round-robin priority scheme to handle each channel. Each PDMA channel transfer (a data transaction of width byte/half-word/word based on PDMA setting) between the peripheral's FIFO and system SRAM takes 9 AHB clock cycles plus 5 APB clock cycles. And the AHB clock cycles for memory-to-memory burst transfers are listed in the table below. The user must consider the worst case time to complete outstanding PDMA requests of all high priority channels in assessing whether there is sufficient PDMA bandwidth for the application. If memory bandwidth utilization is too high then FIFO overflow on peripherals can occur when PDMA request cannot be serviced in a timely manner.

Burst Size	AHB Clock Cycles	Burst Size	AHB Clock Cycles
1	14	16	74
2	18	32	144
4	26	64	284
8	42	128	564

Example:

If the frequency of HCLK is 49.152 MHz, the frequency of PCLK0 and PCLK1 is 24.576 MHz, for each memory-to-peripheral (M2P) and peripheral-to-memory (P2M) transfer takes 19 AHB (9 + 5 * (F_{AHB}/F_{APB})) clock cycles.

Suppose I2S interface is used with a sampling rate of 48 kHz, data width of 32 bits and stereo transmission, PDMA controller needs 48000 * 2(stereo) * 2(RX/TX) * 19 = 3,648,000 AHB clock cycles per second to handle the data transfer rate.

If there is also a 2560000 word, 128 burst size, memory-to-memory (M2M) transfer per second scheduled. Then this takes 2560000/128 * 564 = 11,280,000 AHB clock cycles per second.

The remaining 34,224,000 (49,152,000 – 11,280,000 – 3,648,000) AHB clock cycles can be used for other peripheral data transfer.

Supposed that 2 channels of SPI are enabled with PDMA transfer, the maximum frequency for each SPI bus clock should be less than 14 MHz. The calculation being:

34,224,000 (available cycles/sec) / 4(PDMA channels for SPI TX and RX) / 19 (P2M or M2P AHB clock cycles) * 32 (FIFO width) = 14.41 MHz.

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 = 0x4000_8000**				
PDMA_DSCT0_CTL	PDMA_BA + 0x00	R/W	Descriptor Table Control Register of PDMA Channel 0	0xXXXX_XXXX
PDMA_DSCT0_SA	PDMA_BA + 0x04	R/W	Source Address Register of PDMA Channel 0	0xXXXX_XXXX
PDMA_DSCT0_DA	PDMA_BA + 0x08	R/W	Destination Address Register of PDMA Channel 0	0xXXXX_XXXX
PDMA_DSCT0_NEXT	PDMA_BA + 0x0C	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 0	0xXXXX_XXXX
PDMA_DSCT1_CTL	PDMA_BA + 0x10	R/W	Descriptor Table Control Register of PDMA Channel 1	0xXXXX_XXXX
PDMA_DSCT1_SA	PDMA_BA + 0x14	R/W	Source Address Register of PDMA Channel 1	0xXXXX_XXXX
PDMA_DSCT1_DA	PDMA_BA + 0x18	R/W	Destination Address Register of PDMA Channel 1	0xXXXX_XXXX
PDMA_DSCT1_NEXT	PDMA_BA + 0x1C	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 1	0xXXXX_XXXX
PDMA_DSCT2_CTL	PDMA_BA + 0x20	R/W	Descriptor Table Control Register of PDMA Channel 2	0xXXXX_XXXX
PDMA_DSCT2_SA	PDMA_BA + 0x24	R/W	Source Address Register of PDMA Channel 2	0xXXXX_XXXX
PDMA_DSCT2_DA	PDMA_BA + 0x28	R/W	Destination Address Register of PDMA Channel 2	0xXXXX_XXXX
PDMA_DSCT2_NEXT	PDMA_BA + 0x2C	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 2	0xXXXX_XXXX
PDMA_DSCT3_CTL	PDMA_BA + 0x30	R/W	Descriptor Table Control Register of PDMA Channel 3	0xXXXX_XXXX
PDMA_DSCT3_SA	PDMA_BA + 0x34	R/W	Source Address Register of PDMA Channel 3	0xXXXX_XXXX
PDMA_DSCT3_DA	PDMA_BA + 0x38	R/W	Destination Address Register of PDMA Channel 3	0xXXXX_XXXX
PDMA_DSCT3_NEXT	PDMA_BA + 0x3C	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 3	0xXXXX_XXXX
PDMA_DSCT4_CTL	PDMA_BA + 0x40	R/W	Descriptor Table Control Register of PDMA Channel 4	0xXXXX_XXXX
PDMA_DSCT4_SA	PDMA_BA + 0x44	R/W	Source Address Register of PDMA Channel 4	0xXXXX_XXXX
PDMA_DSCT4_DA	PDMA_BA + 0x48	R/W	Destination Address Register of PDMA Channel 4	0xXXXX_XXXX
PDMA_DSCT4_NEXT	PDMA_BA + 0x4C	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 4	0xXXXX_XXXX
PDMA_DSCT5_CTL	PDMA_BA + 0x50	R/W	Descriptor Table Control Register of PDMA Channel 5	0xXXXX_XXXX
PDMA_DSCT5_SA	PDMA_BA + 0x54	R/W	Source Address Register of PDMA Channel 5	0xXXXX_XXXX
PDMA_DSCT5_DA	PDMA_BA + 0x58	R/W	Destination Address Register of PDMA Channel 5	0xXXXX_XXXX
PDMA_DSCT5_NEXT	PDMA_BA + 0x5C	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 5	0xXXXX_XXXX

Register	Offset	R/W	Description	Reset Value
PDMA Base Address: **PDMA_BA = 0x4000_8000**				
PDMA_DSCT6_CTL	PDMA_BA + 0x60	R/W	Descriptor Table Control Register of PDMA Channel 6	0xXXXX_XXXX
PDMA_DSCT6_SA	PDMA_BA + 0x64	R/W	Source Address Register of PDMA Channel 6	0xXXXX_XXXX
PDMA_DSCT6_DA	PDMA_BA + 0x68	R/W	Destination Address Register of PDMA Channel 6	0xXXXX_XXXX
PDMA_DSCT6_NEXT	PDMA_BA + 0x6C	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 6	0xXXXX_XXXX
PDMA_DSCT7_CTL	PDMA_BA + 0x70	R/W	Descriptor Table Control Register of PDMA Channel 7	0xXXXX_XXXX
PDMA_DSCT7_SA	PDMA_BA + 0x74	R/W	Source Address Register of PDMA Channel 7	0xXXXX_XXXX
PDMA_DSCT7_DA	PDMA_BA + 0x78	R/W	Destination Address Register of PDMA Channel 7	0xXXXX_XXXX
PDMA_DSCT7_NEXT	PDMA_BA + 0x7C	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 7	0xXXXX_XXXX
PDMA_DSCT8_CTL	PDMA_BA + 0x80	R/W	Descriptor Table Control Register of PDMA Channel 8	0xXXXX_XXXX
PDMA_DSCT8_SA	PDMA_BA + 0x84	R/W	Source Address Register of PDMA Channel 8	0xXXXX_XXXX
PDMA_DSCT8_DA	PDMA_BA + 0x88	R/W	Destination Address Register of PDMA Channel 8	0xXXXX_XXXX
PDMA_DSCT8_NEXT	PDMA_BA + 0x8C	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 8	0xXXXX_XXXX
PDMA_DSCT9_CTL	PDMA_BA + 0x90	R/W	Descriptor Table Control Register of PDMA Channel 9	0xXXXX_XXXX
PDMA_DSCT9_SA	PDMA_BA + 0x94	R/W	Source Address Register of PDMA Channel 9	0xXXXX_XXXX
PDMA_DSCT9_DA	PDMA_BA + 0x98	R/W	Destination Address Register of PDMA Channel 9	0xXXXX_XXXX
PDMA_DSCT9_NEXT	PDMA_BA + 0x9C	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 9	0xXXXX_XXXX
PDMA_DSCT10_CTL	PDMA_BA + 0xA0	R/W	Descriptor Table Control Register of PDMA Channel 10	0xXXXX_XXXX
PDMA_DSCT10_SA	PDMA_BA + 0xA4	R/W	Source Address Register of PDMA Channel 10	0xXXXX_XXXX
PDMA_DSCT10_DA	PDMA_BA + 0xA8	R/W	Destination Address Register of PDMA Channel 10	0xXXXX_XXXX
PDMA_DSCT10_NEXT	PDMA_BA + 0xAC	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 10	0xXXXX_XXXX
PDMA_DSCT11_CTL	PDMA_BA + 0xB0	R/W	Descriptor Table Control Register of PDMA Channel 11	0xXXXX_XXXX
PDMA_DSCT11_SA	PDMA_BA + 0xB4	R/W	Source Address Register of PDMA Channel 11	0xXXXX_XXXX
PDMA_DSCT11_DA	PDMA_BA + 0xB8	R/W	Destination Address Register of PDMA Channel 11	0xXXXX_XXXX
PDMA_DSCT11_NEXT	PDMA_BA + 0xBC	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 11	0xXXXX_XXXX
PDMA_DSCT12_CTL	PDMA_BA + 0xC0	R/W	Descriptor Table Control Register of PDMA Channel 12	0xXXXX_XXXX

Register	Offset	R/W	Description	Reset Value
PDMA Base Address: **PDMA_BA = 0x4000_8000**				
PDMA_DSCT12_SA	PDMA_BA + 0xC4	R/W	Source Address Register of PDMA Channel 12	0xXXXX_XXXX
PDMA_DSCT12_DA	PDMA_BA + 0xC8	R/W	Destination Address Register of PDMA Channel 12	0xXXXX_XXXX
PDMA_DSCT12_NEXT	PDMA_BA + 0xCC	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 12	0xXXXX_XXXX
PDMA_DSCT13_CTL	PDMA_BA + 0xD0	R/W	Descriptor Table Control Register of PDMA Channel 13	0xXXXX_XXXX
PDMA_DSCT13_SA	PDMA_BA + 0xD4	R/W	Source Address Register of PDMA Channel 13	0xXXXX_XXXX
PDMA_DSCT13_DA	PDMA_BA + 0xD8	R/W	Destination Address Register of PDMA Channel 13	0xXXXX_XXXX
PDMA_DSCT13_NEXT	PDMA_BA + 0xDC	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 13	0xXXXX_XXXX
PDMA_DSCT14_CTL	PDMA_BA + 0xE0	R/W	Descriptor Table Control Register of PDMA Channel 14	0xXXXX_XXXX
PDMA_DSCT14_SA	PDMA_BA + 0xE4	R/W	Source Address Register of PDMA Channel 14	0xXXXX_XXXX
PDMA_DSCT14_DA	PDMA_BA + 0xE8	R/W	Destination Address Register of PDMA Channel 14	0xXXXX_XXXX
PDMA_DSCT14_NEXT	PDMA_BA + 0xEC	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 14	0xXXXX_XXXX
PDMA_DSCT15_CTL	PDMA_BA + 0xF0	R/W	Descriptor Table Control Register of PDMA Channel 15	0xXXXX_XXXX
PDMA_DSCT15_SA	PDMA_BA + 0xF4	R/W	Source Address Register of PDMA Channel 15	0xXXXX_XXXX
PDMA_DSCT15_DA	PDMA_BA + 0xF8	R/W	Destination Address Register of PDMA Channel 15	0xXXXX_XXXX
PDMA_DSCT15_NEXT	PDMA_BA + 0xFC	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 15	0xXXXX_XXXX
PDMA_CURSCAT0	PDMA_BA + 0x100	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 0	0xXXXX_XXXX
PDMA_CURSCAT1	PDMA_BA + 0x104	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 1	0xXXXX_XXXX
PDMA_CURSCAT2	PDMA_BA + 0x108	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 2	0xXXXX_XXXX
PDMA_CURSCAT3	PDMA_BA + 0x10C	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 3	0xXXXX_XXXX
PDMA_CURSCAT4	PDMA_BA + 0x110	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 4	0xXXXX_XXXX
PDMA_CURSCAT5	PDMA_BA + 0x114	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 5	0xXXXX_XXXX
PDMA_CURSCAT6	PDMA_BA + 0x118	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 6	0xXXXX_XXXX
PDMA_CURSCAT7	PDMA_BA + 0x11C	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 7	0xXXXX_XXXX
PDMA_CURSCAT8	PDMA_BA + 0x120	R	Current Scatter-Gather Descriptor Table Address of	0xXXXX_XXXX

Register	Offset	R/W	Description	Reset Value
PDMA Base Address: **PDMA_BA = 0x4000_8000**				
			PDMA Channel 8	
PDMA_CURSCAT9	PDMA_BA + 0x124	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 9	0xXXXX_XXXX
PDMA_CURSCAT10	PDMA_BA + 0x128	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 10	0xXXXX_XXXX
PDMA_CURSCAT11	PDMA_BA + 0x12C	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 11	0xXXXX_XXXX
PDMA_CURSCAT12	PDMA_BA + 0x130	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 12	0xXXXX_XXXX
PDMA_CURSCAT13	PDMA_BA + 0x134	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 13	0xXXXX_XXXX
PDMA_CURSCAT14	PDMA_BA + 0x138	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 14	0xXXXX_XXXX
PDMA_CURSCAT15	PDMA_BA + 0x13C	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 15	0xXXXX_XXXX
PDMA_CHCTL	PDMA_BA + 0x400	R/W	PDMA Channel Control Register	0x0000_0000
PDMA_STOP	PDMA_BA + 0x404	W	PDMA Transfer Stop 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_ALIGN	PDMA_BA + 0x428	R/W	PDMA Transfer Alignment 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 Time-out Counter Ch1 and Ch0 Register	0xFFFF_FFFF
PDMA_CHRST	PDMA_BA + 0x460	R/W	PDMA Channel Reset Register	0x0000_0000

Register	Offset	R/W	Description	Reset Value
PDMA Base Address: **PDMA_BA = 0x4000_8000**				
PDMA_REQSEL0_3	PDMA_BA + 0x480	R/W	PDMA Request Source Select Register 0	0x0000_0000
PDMA_REQSEL4_7	PDMA_BA + 0x484	R/W	PDMA Request Source Select Register 1	0x0000_0000
PDMA_REQSEL8_11	PDMA_BA + 0x488	R/W	PDMA Request Source Select Register 2	0x0000_0000
PDMA_REQSEL12_15	PDMA_BA + 0x48C	R/W	PDMA Request Source Select Register 3	0x0000_0000
PDMA_STCR0	PDMA_BA + 0x500	R/W	Stride Transfer Count Register of PDMA Channel 0	0x0000_0000
PDMA_ASOCR0	PDMA_BA + 0x504	R/W	Address Stride Offset Register of PDMA Channel 0	0x0000_0000
PDMA_STCR1	PDMA_BA + 0x508	R/W	Stride Transfer Count Register of PDMA Channel 1	0x0000_0000
PDMA_ASOCR1	PDMA_BA + 0x50C	R/W	Address Stride Offset Register of PDMA Channel 1	0x0000_0000
PDMA_STCR2	PDMA_BA + 0x510	R/W	Stride Transfer Count Register of PDMA Channel 2	0x0000_0000
PDMA_ASOCR2	PDMA_BA + 0x514	R/W	Address Stride Offset Register of PDMA Channel 2	0x0000_0000
PDMA_STCR3	PDMA_BA + 0x518	R/W	Stride Transfer Count Register of PDMA Channel 3	0x0000_0000
PDMA_ASOCR3	PDMA_BA + 0x51C	R/W	Address Stride Offset Register of PDMA Channel 3	0x0000_0000
PDMA_STCR4	PDMA_BA + 0x520	R/W	Stride Transfer Count Register of PDMA Channel 4	0x0000_0000
PDMA_ASOCR4	PDMA_BA + 0x524	R/W	Address Stride Offset Register of PDMA Channel 4	0x0000_0000
PDMA_STCR5	PDMA_BA + 0x528	R/W	Stride Transfer Count Register of PDMA Channel 5	0x0000_0000
PDMA_ASOCR5	PDMA_BA + 0x52C	R/W	Address Stride Offset Register of PDMA Channel 5	0x0000_0000

Note:

1. Any register not listed here is reserved and must not be written. The result of a read operation on these bits is undefined.
2. The reserved register fields that listed in register addresses PDMA_BA+0x0 to PDMA_BA+0x140 must be written as 0. Writing reserved fields with other than 0 may produce undefined results.
3. The reserved register fields that listed in register addresses greater than PDMA_BA+0x400 must be written with reset value. Writing reserved fields with other than reset value may produce undefined results.

6.6.7 Register Description

Descriptor Table Control Register (PDMA_DSCTn_CTL)

Register	Offset	R/W	Description	Reset Value
PDMA_DSCT0_CTL	PDMA_BA + 0x00	R/W	Descriptor Table Control Register of PDMA Channel 0	0xXXXX_XXXX
PDMA_DSCT1_CTL	PDMA_BA + 0x10	R/W	Descriptor Table Control Register of PDMA Channel 1	0xXXXX_XXXX
PDMA_DSCT2_CTL	PDMA_BA + 0x20	R/W	Descriptor Table Control Register of PDMA Channel 2	0xXXXX_XXXX
PDMA_DSCT3_CTL	PDMA_BA + 0x30	R/W	Descriptor Table Control Register of PDMA Channel 3	0xXXXX_XXXX
PDMA_DSCT4_CTL	PDMA_BA + 0x40	R/W	Descriptor Table Control Register of PDMA Channel 4	0xXXXX_XXXX
PDMA_DSCT5_CTL	PDMA_BA + 0x50	R/W	Descriptor Table Control Register of PDMA Channel 5	0xXXXX_XXXX
PDMA_DSCT6_CTL	PDMA_BA + 0x60	R/W	Descriptor Table Control Register of PDMA Channel 6	0xXXXX_XXXX
PDMA_DSCT7_CTL	PDMA_BA + 0x70	R/W	Descriptor Table Control Register of PDMA Channel 7	0xXXXX_XXXX
PDMA_DSCT8_CTL	PDMA_BA + 0x80	R/W	Descriptor Table Control Register of PDMA Channel 8	0xXXXX_XXXX
PDMA_DSCT9_CTL	PDMA_BA + 0x90	R/W	Descriptor Table Control Register of PDMA Channel 9	0xXXXX_XXXX
PDMA_DSCT10_CTL	PDMA_BA + 0xA0	R/W	Descriptor Table Control Register of PDMA Channel 10	0xXXXX_XXXX
PDMA_DSCT11_CTL	PDMA_BA + 0xB0	R/W	Descriptor Table Control Register of PDMA Channel 11	0xXXXX_XXXX
PDMA_DSCT12_CTL	PDMA_BA + 0xC0	R/W	Descriptor Table Control Register of PDMA Channel 12	0xXXXX_XXXX
PDMA_DSCT13_CTL	PDMA_BA + 0xD0	R/W	Descriptor Table Control Register of PDMA Channel 13	0xXXXX_XXXX
PDMA_DSCT14_CTL	PDMA_BA + 0xE0	R/W	Descriptor Table Control Register of PDMA Channel 14	0xXXXX_XXXX
PDMA_DSCT15_CTL	PDMA_BA + 0xF0	R/W	Descriptor Table Control Register of PDMA Channel 15	0xXXXX_XXXX

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

Bits	Description	
[31:16]	**TXCNT**	**Transfer Count** The TXCNT represents the required number of PDMA transfer, the real transfer count is (TXCNT + 1); The maximum transfer count is 65536, 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]	**STRIDEEN**	**Stride Mode Enable Bit**

		0 = Stride transfer mode Disabled. 1 = Stride transfer mode Enabled.
[14]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with 0.
[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. Do not use. **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. 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. This bit is only used for scatter-gather mode. If the TBINTDIS bit is enabled when PDMA controller finishes transfer task, it will not generates transfer done interrupt. 0 = Table interrupt Enabled. 1 = Table interrupt Disabled.
[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. Any values read should be ignored. When writing to this field always write with 0.
[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 next descriptor table address in PDMA_DSCT_NEXT register; PDMA controller will ignore this task, then load the next task to execute. 11 = Reserved. Do not use. **Note:** Before filling transfer task in the Descriptor Table, user must check if the descriptor table is complete.

Note: The reset value of this field will be different every time powered chip.

Start Source Address Register (PDMA_DSCTn_SA)

Register	Offset	R/W	Description	Reset Value
PDMA_DSCT0_SA	PDMA_BA + 0x04	R/W	Source Address Register of PDMA Channel 0	0xXXXX_XXXX
PDMA_DSCT1_SA	PDMA_BA + 0x14	R/W	Source Address Register of PDMA Channel 1	0xXXXX_XXXX
PDMA_DSCT2_SA	PDMA_BA + 0x24	R/W	Source Address Register of PDMA Channel 2	0xXXXX_XXXX
PDMA_DSCT3_SA	PDMA_BA + 0x34	R/W	Source Address Register of PDMA Channel 3	0xXXXX_XXXX
PDMA_DSCT4_SA	PDMA_BA + 0x44	R/W	Source Address Register of PDMA Channel 4	0xXXXX_XXXX
PDMA_DSCT5_SA	PDMA_BA + 0x54	R/W	Source Address Register of PDMA Channel 5	0xXXXX_XXXX
PDMA_DSCT6_SA	PDMA_BA + 0x64	R/W	Source Address Register of PDMA Channel 6	0xXXXX_XXXX
PDMA_DSCT7_SA	PDMA_BA + 0x74	R/W	Source Address Register of PDMA Channel 7	0xXXXX_XXXX
PDMA_DSCT8_SA	PDMA_BA + 0x84	R/W	Source Address Register of PDMA Channel 8	0xXXXX_XXXX
PDMA_DSCT9_SA	PDMA_BA + 0x94	R/W	Source Address Register of PDMA Channel 9	0xXXXX_XXXX
PDMA_DSCT10_SA	PDMA_BA + 0xA4	R/W	Source Address Register of PDMA Channel 10	0xXXXX_XXXX
PDMA_DSCT11_SA	PDMA_BA + 0xB4	R/W	Source Address Register of PDMA Channel 11	0xXXXX_XXXX
PDMA_DSCT12_SA	PDMA_BA + 0xC4	R/W	Source Address Register of PDMA Channel 12	0xXXXX_XXXX
PDMA_DSCT13_SA	PDMA_BA + 0xD4	R/W	Source Address Register of PDMA Channel 13	0xXXXX_XXXX
PDMA_DSCT14_SA	PDMA_BA + 0xE4	R/W	Source Address Register of PDMA Channel 14	0xXXXX_XXXX
PDMA_DSCT15_SA	PDMA_BA + 0xF4	R/W	Source Address Register of PDMA Channel 15	0xXXXX_XXXX

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

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

Note: The reset value of this field will be different every time powered chip.

Destination Address Register (PDMA_DSCTn_DA)

Register	Offset	R/W	Description	Reset Value
PDMA_DSCT0_DA	PDMA_BA + 0x08	R/W	Destination Address Register of PDMA Channel 0	0xXXXX_XXXX
PDMA_DSCT1_DA	PDMA_BA + 0x18	R/W	Destination Address Register of PDMA Channel 1	0xXXXX_XXXX
PDMA_DSCT2_DA	PDMA_BA + 0x28	R/W	Destination Address Register of PDMA Channel 2	0xXXXX_XXXX
PDMA_DSCT3_DA	PDMA_BA + 0x38	R/W	Destination Address Register of PDMA Channel 3	0xXXXX_XXXX
PDMA_DSCT4_DA	PDMA_BA + 0x48	R/W	Destination Address Register of PDMA Channel 4	0xXXXX_XXXX
PDMA_DSCT5_DA	PDMA_BA + 0x58	R/W	Destination Address Register of PDMA Channel 5	0xXXXX_XXXX
PDMA_DSCT6_DA	PDMA_BA + 0x68	R/W	Destination Address Register of PDMA Channel 6	0xXXXX_XXXX
PDMA_DSCT7_DA	PDMA_BA + 0x78	R/W	Destination Address Register of PDMA Channel 7	0xXXXX_XXXX
PDMA_DSCT8_DA	PDMA_BA + 0x88	R/W	Destination Address Register of PDMA Channel 8	0xXXXX_XXXX
PDMA_DSCT9_DA	PDMA_BA + 0x98	R/W	Destination Address Register of PDMA Channel 9	0xXXXX_XXXX
PDMA_DSCT10_DA	PDMA_BA + 0xA8	R/W	Destination Address Register of PDMA Channel 10	0xXXXX_XXXX
PDMA_DSCT11_DA	PDMA_BA + 0xB8	R/W	Destination Address Register of PDMA Channel 11	0xXXXX_XXXX
PDMA_DSCT12_DA	PDMA_BA + 0xC8	R/W	Destination Address Register of PDMA Channel 12	0xXXXX_XXXX
PDMA_DSCT13_DA	PDMA_BA + 0xD8	R/W	Destination Address Register of PDMA Channel 13	0xXXXX_XXXX
PDMA_DSCT14_DA	PDMA_BA + 0xE8	R/W	Destination Address Register of PDMA Channel 14	0xXXXX_XXXX
PDMA_DSCT15_DA	PDMA_BA + 0xF8	R/W	Destination Address Register of PDMA Channel 15	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.

Note: The reset value of this field will be different every time powered chip.

First Scatter-gather Descriptor Table Offset Address (PDMA_DSCTn_NEXT)

Register	Offset	R/W	Description	Reset Value
PDMA_DSCT0_NEXT	PDMA_BA + 0x0C	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 0	0xXXXX_XXXX
PDMA_DSCT1_NEXT	PDMA_BA + 0x1C	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 1	0xXXXX_XXXX
PDMA_DSCT2_NEXT	PDMA_BA + 0x2C	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 2	0xXXXX_XXXX
PDMA_DSCT3_NEXT	PDMA_BA + 0x3C	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 3	0xXXXX_XXXX
PDMA_DSCT4_NEXT	PDMA_BA + 0x4C	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 4	0xXXXX_XXXX
PDMA_DSCT5_NEXT	PDMA_BA + 0x5C	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 5	0xXXXX_XXXX
PDMA_DSCT6_NEXT	PDMA_BA + 0x6C	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 6	0xXXXX_XXXX
PDMA_DSCT7_NEXT	PDMA_BA + 0x7C	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 7	0xXXXX_XXXX
PDMA_DSCT8_NEXT	PDMA_BA + 0x8C	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 8	0xXXXX_XXXX
PDMA_DSCT9_NEXT	PDMA_BA + 0x9C	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 9	0xXXXX_XXXX
PDMA_DSCT10_NEXT	PDMA_BA + 0xAC	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 10	0xXXXX_XXXX
PDMA_DSCT11_NEXT	PDMA_BA + 0xBC	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 11	0xXXXX_XXXX
PDMA_DSCT12_NEXT	PDMA_BA + 0xCC	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 12	0xXXXX_XXXX
PDMA_DSCT13_NEXT	PDMA_BA + 0xDC	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 13	0xXXXX_XXXX
PDMA_DSCT14_NEXT	PDMA_BA + 0xEC	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 14	0xXXXX_XXXX
PDMA_DSCT15_NEXT	PDMA_BA + 0xFC	R/W	First Scatter-Gather Descriptor Table Offset Address of PDMA Channel 15	0xXXXX_XXXX

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

Bits	Description	
[31:16]	**EXENEXT**	**PDMA Execution Next Descriptor Table Offset** This field indicates the offset of next descriptor table address of current execution descriptor table in system memory. **Note:** write operation is useless in this field.
[15:0]	**NEXT**	**PDMA Next Descriptor Table Offset** This field indicates the offset of the next descriptor table address in system memory. **Write Operation:** If the system memory based address is 0x2000_0000 (PDMA_SCATBA), and the next 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 NEXT[1:0] will become reserved, and indicate the first next address of system memory. **Note 1:** The first descriptor table address must be word boundary. **Note 2:** Before filled transfer task in the descriptor table, user must check if the descriptor table is complete. **Note 3:** The address of descriptor tables must be in the same 64KB area.

Note: The reset value of this field will be different every time powered chip.

Current Scatter-gather Descriptor Table Address (PDMA_CURSCATn)

Register	Offset	R/W	Description	Reset Value
PDMA_CURSCAT0	PDMA_BA + 0x100	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 0	0xXXXX_XXXX
PDMA_CURSCAT1	PDMA_BA + 0x104	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 1	0xXXXX_XXXX
PDMA_CURSCAT2	PDMA_BA + 0x108	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 2	0xXXXX_XXXX
PDMA_CURSCAT3	PDMA_BA + 0x10C	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 3	0xXXXX_XXXX
PDMA_CURSCAT4	PDMA_BA + 0x110	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 4	0xXXXX_XXXX
PDMA_CURSCAT5	PDMA_BA + 0x114	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 5	0xXXXX_XXXX
PDMA_CURSCAT6	PDMA_BA + 0x118	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 6	0xXXXX_XXXX
PDMA_CURSCAT7	PDMA_BA + 0x11C	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 7	0xXXXX_XXXX
PDMA_CURSCAT8	PDMA_BA + 0x120	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 8	0xXXXX_XXXX
PDMA_CURSCAT9	PDMA_BA + 0x124	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 9	0xXXXX_XXXX
PDMA_CURSCAT10	PDMA_BA + 0x128	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 10	0xXXXX_XXXX
PDMA_CURSCAT11	PDMA_BA + 0x12C	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 11	0xXXXX_XXXX
PDMA_CURSCAT12	PDMA_BA + 0x130	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 12	0xXXXX_XXXX
PDMA_CURSCAT13	PDMA_BA + 0x134	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 13	0xXXXX_XXXX
PDMA_CURSCAT14	PDMA_BA + 0x138	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 14	0xXXXX_XXXX
PDMA_CURSCAT15	PDMA_BA + 0x13C	R	Current Scatter-Gather Descriptor Table Address of PDMA Channel 15	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.

Note: The reset value of this field will be different every time powered chip.

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
CHEN15	CHEN14	CHEN13	CHEN12	CHEN11	CHEN10	CHEN9	CHEN8
7	6	5	4	3	2	1	0
CHEN7	CHEN6	CHEN5	CHEN4	CHEN3	CHEN2	CHEN1	CHEN0

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15]	**CHEN15**	**PDMA Channel 15 Enable Bit** Set this bit to 1 to enable PDMA channel 15 operation. Channel 15 cannot be active if it is not set as enabled. 0 = PDMA Channel 15 Disabled. 1 = PDMA Channel 15 Enabled. **Note:** Set corresponding bit of PDMA_STOP or PDMA_CHRST register will also clear this bit.
[14]	**CHEN14**	**PDMA Channel 14 Enable Bit** Set this bit to 1 to enable PDMA channel 14 operation. Channel 14 cannot be active if it is not set as enabled. 0 = PDMA Channel 14 Disabled. 1 = PDMA Channel 14 Enabled. **Note:** Set corresponding bit of PDMA_STOP or PDMA_CHRST register will also clear this bit.
[13]	**CHEN13**	**PDMA Channel 13 Enable Bit** Set this bit to 1 to enable PDMA channel 13 operation. Channel 13 cannot be active if it is not set as enabled. 0 = PDMA Channel 13 Disabled. 1 = PDMA Channel 13 Enabled. **Note:** Set corresponding bit of PDMA_STOP or PDMA_CHRST register will also clear this bit.
[12]	**CHEN12**	**PDMA Channel 12 Enable Bit** Set this bit to 1 to enable PDMA channel 12 operation. Channel 12 cannot be active if it is not set as enabled. 0 = PDMA Channel 12 Disabled. 1 = PDMA Channel 12 Enabled. **Note:** Set corresponding bit of PDMA_STOP or PDMA_CHRST register will also clear

Bits	Description	
		this bit.
[11]	**CHEN11**	**PDMA Channel 11 Enable Bit** Set this bit to 1 to enable PDMA channel 11 operation. Channel 11 cannot be active if it is not set as enabled. 0 = PDMA Channel 11 Disabled. 1 = PDMA Channel 11 Enabled. **Note:** Set corresponding bit of PDMA_STOP or PDMA_CHRST register will also clear this bit.
[10]	**CHEN10**	**PDMA Channel 10 Enable Bit** Set this bit to 1 to enable PDMA channel 10 operation. Channel 10 cannot be active if it is not set as enabled. 0 = PDMA Channel 10 Disabled. 1 = PDMA Channel 10 Enabled. **Note:** Set corresponding bit of PDMA_STOP or PDMA_CHRST register will also clear this bit.
[9]	**CHEN9**	**PDMA Channel 9 Enable Bit** Set this bit to 1 to enable PDMA channel 9 operation. Channel 9 cannot be active if it is not set as enabled. 0 = PDMA channel 9 Disabled. 1 = PDMA channel 9 Enabled. **Note:** Set corresponding bit of PDMA_STOP or PDMA_CHRST register will also clear this bit.
[8]	**CHEN8**	**PDMA Channel 8 Enable Bit** Set this bit to 1 to enable PDMA channel 8 operation. Channel 8 cannot be active if it is not set as enabled. 0 = PDMA Channel 8 Disabled. 1 = PDMA Channel 8 Enabled. **Note:** Set corresponding bit of PDMA_STOP or PDMA_CHRST register will also clear this bit.
[7]	**CHEN7**	**PDMA Channel 7 Enable Bit** Set this bit to 1 to enable PDMA channel 7 operation. Channel 7 cannot be active if it is not set as enabled. 0 = PDMA Channel 7 Disabled. 1 = PDMA Channel 7 Enabled. **Note:** Set corresponding bit of PDMA_STOP or PDMA_CHRST register will also clear this bit.
[6]	**CHEN6**	**PDMA Channel 6 Enable Bit** Set this bit to 1 to enable PDMA channel 6 operation. Channel 6 cannot be active if it is not set as enabled. 0 = PDMA Channel 6 Disabled. 1 = PDMA Channel 6 Enabled. **Note:** Set corresponding bit of PDMA_STOP or PDMA_CHRST register will also clear this bit.
[5]	**CHEN5**	**PDMA Channel 5 Enable Bit** Set this bit to 1 to enable PDMA channel 5 operation. Channel 5 cannot be active if it is not set as enabled. 0 = PDMA Channel 5 Disabled. 1 = PDMA Channel 5 Enabled. **Note:** Set corresponding bit of PDMA_STOP or PDMA_CHRST register will also clear

Bits	Description	
		this bit.
[4]	**CHEN4**	**PDMA Channel 4 Enable Bit** Set this bit to 1 to enable PDMA channel 4 operation. Channel 4 cannot be active if it is not set as enabled. 0 = PDMA Channel 4 Disabled. 1 = PDMA Channel 4 Enabled. **Note:** Set corresponding bit of PDMA_STOP or PDMA_CHRST register will also clear this bit.
[3]	**CHEN3**	**PDMA Channel 3 Enable Bit** Set this bit to 1 to enable PDMA channel 3 operation. Channel 3 cannot be active if it is not set as enabled. 0 = PDMA Channel 3 Disabled. 1 = PDMA Channel 3 Enabled. **Note:** Set corresponding bit of PDMA_STOP or PDMA_CHRST register will also clear this bit.
[2]	**CHEN2**	**PDMA Channel 2 Enable Bit** Set this bit to 1 to enable PDMA channel 2 operation. Channel 2 cannot be active if it is not set as enabled. 0 = PDMA Channel 2 Disabled. 1 = PDMA Channel 2 Enabled. **Note:** Set corresponding bit of PDMA_STOP or PDMA_CHRST register will also clear this bit.
[1]	**CHEN1**	**PDMA Channel 1 Enable Bit** Set this bit to 1 to enable PDMA channel 1 operation. Channel 1 cannot be active if it is not set as enabled. 0 = PDMA Channel 1 Disabled. 1 = PDMA Channel 1 Enabled. **Note:** Set corresponding bit of PDMA_STOP or PDMA_CHRST register will also clear this bit.
[0]	**CHEN0**	**PDMA Channel 0 Enable Bit** Set this bit to 1 to enable PDMA channel 0 operation. Channel 0 cannot be active if it is not set as enabled. 0 = PDMA Channel 0 Disabled. 1 = PDMA Channel 0 Enabled. **Note:** Set corresponding bit of PDMA_STOP or PDMA_CHRST register will also clear this bit.

PDMA Transfer Stop Control Register (PDMA_STOP)

Register	Offset	R/W	Description	Reset Value
PDMA_STOP	PDMA_BA + 0x404	W	PDMA Transfer Stop 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
STOP15	STOP14	STOP13	STOP12	STOP11	STOP10	STOP9	STOP8
7	6	5	4	3	2	1	0
STOP7	STOP6	STOP5	STOP4	STOP3	STOP2	STOP1	STOP0

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15]	**STOP15**	**PDMA Channel 15 Transfer Stop Control Register (Write Only)** User can set this bit to stop the PDMA channel 15 transfer. When user sets STOP15 bit, the PDMA controller will stop the on-going transfer, then clear the channel enable bit CHEN15 (PDMA_CHCTL [15]) and clear request active flag. If re-enable the stopped channel again, the remaining transfers will be processed. 0 = No effect. 1 = Stop PDMA channel 15 transfer.
[14]	**STOP14**	**PDMA Channel 14 Transfer Stop Control Register (Write Only)** User can set this bit to stop the PDMA channel 14 transfer. When user sets STOP14 bit, the PDMA controller will stop the on-going transfer, then clear the channel enable bit CHEN14 (PDMA_CHCTL [14]) and clear request active flag. If re-enable the stopped channel again, the remaining transfers will be processed. 0 = No effect. 1 = Stop PDMA channel 14 transfer.
[13]	**STOP13**	**PDMA Channel 13 Transfer Stop Control Register (Write Only)** User can set this bit to stop the PDMA channel 13 transfer. When user sets STOP13 bit, the PDMA controller will stop the on-going transfer, then clear the channel enable bit CHEN13 (PDMA_CHCTL [13]) and clear request active flag. If re-enable the stopped channel again, the remaining transfers will be processed. 0 = No effect. 1 = Stop PDMA channel 13 transfer.
[12]	**STOP12**	**PDMA Channel 12 Transfer Stop Control Register (Write Only)** User can set this bit to stop the PDMA channel 12 transfer. When user sets STOP12 bit, the PDMA controller will stop the on-going transfer, then clear the channel enable bit CHEN12 (PDMA_CHCTL [12]) and clear request active flag. If re-enable the stopped channel again, the remaining transfers will be processed. 0 = No effect. 1 = Stop PDMA channel 12 transfer.

Bits	Description	
[11]	**STOP11**	**PDMA Channel 11 Transfer Stop Control Register (Write Only)** User can set this bit to stop the PDMA channel 11 transfer. When user sets STOP11 bit, the PDMA controller will stop the on-going transfer, then clear the channel enable bit CHEN11 (PDMA_CHCTL [11]) and clear request active flag. If re-enable the stopped channel again, the remaining transfers will be processed. 0 = No effect. 1 = Stop PDMA channel 11 transfer.
[10]	**STOP10**	**PDMA Channel 10 Transfer Stop Control Register (Write Only)** User can set this bit to stop the PDMA channel 10 transfer. When user sets STOP10 bit, the PDMA controller will stop the on-going transfer, then clear the channel enable bit CHEN10 (PDMA_CHCTL [10]) and clear request active flag. If re-enable the stopped channel again, the remaining transfers will be processed. 0 = No effect. 1 = Stop PDMA channel 10 transfer.
[9]	**STOP9**	**PDMA Channel 9 Transfer Stop Control Register (Write Only)** User can set this bit to stop the PDMA channel 9 transfer. When user sets STOP9 bit, the PDMA controller will stop the on-going transfer, then clear the channel enable bit CHEN9 (PDMA_CHCTL [9]) and clear request active flag. If re-enable the stopped channel again, the remaining transfers will be processed. 0 = No effect. 1 = Stop PDMA channel 9 transfer.
[8]	**STOP8**	**PDMA Channel 8 Transfer Stop Control Register (Write Only)** User can set this bit to stop the PDMA channel 8 transfer. When user sets STOP8 bit, the PDMA controller will stop the on-going transfer, then clear the channel enable bit CHEN8 (PDMA_CHCTL [8]) and clear request active flag. If re-enable the stopped channel again, the remaining transfers will be processed. 0 = No effect. 1 = Stop PDMA channel 8 transfer.
[7]	**STOP7**	**PDMA Channel 7 Transfer Stop Control Register (Write Only)** User can set this bit to stop the PDMA channel 7 transfer. When user sets STOP7 bit, the PDMA controller will stop the on-going transfer, then clear the channel enable bit CHEN7 (PDMA_CHCTL [7]) and clear request active flag. If re-enable the stopped channel again, the remaining transfers will be processed. 0 = No effect. 1 = Stop PDMA channel 7 transfer.
[6]	**STOP6**	**PDMA Channel 6 Transfer Stop Control Register (Write Only)** User can set this bit to stop the PDMA channel 6 transfer. When user sets STOP6 bit, the PDMA controller will stop the on-going transfer, then clear the channel enable bit CHEN6 (PDMA_CHCTL [6]) and clear request active flag. If re-enable the stopped channel again, the remaining transfers will be processed. 0 = No effect. 1 = Stop PDMA channel 6 transfer.
[5]	**STOP5**	**PDMA Channel 5 Transfer Stop Control Register (Write Only)** User can set this bit to stop the PDMA channel 5 transfer. When user sets STOP5 bit, the PDMA controller will stop the on-going transfer, then clear the channel enable bit CHEN5 (PDMA_CHCTL [5]) and clear request active flag. If re-enable the stopped channel again, the remaining transfers will be processed. 0 = No effect. 1 = Stop PDMA channel 5 transfer.
[4]	**STOP4**	**PDMA Channel 4 Transfer Stop Control Register (Write Only)** User can set this bit to stop the PDMA channel 4 transfer. When user sets STOP4 bit, the

Bits	Description	
		PDMA controller will stop the on-going transfer, then clear the channel enable bit CHEN4 (PDMA_CHCTL [4]) and clear request active flag. If re-enable the stopped channel again, the remaining transfers will be processed. 0 = No effect. 1 = Stop PDMA channel 4 transfer.
[3]	**STOP3**	**PDMA Channel 3 Transfer Stop Control Register (Write Only)** User can set this bit to stop the PDMA channel 3 transfer. When user sets STOP3 bit, the PDMA controller will stop the on-going transfer, then clear the channel enable bit CHEN3 (PDMA_CHCTL [3]) and clear request active flag. If re-enable the stopped channel again, the remaining transfers will be processed. 0 = No effect. 1 = Stop PDMA channel 3 transfer.
[2]	**STOP2**	**PDMA Channel 2 Transfer Stop Control Register (Write Only)** User can set this bit to stop the PDMA channel 2 transfer. When user sets STOP2 bit, the PDMA controller will stop the on-going transfer, then clear the channel enable bit CHEN2 (PDMA_CHCTL [2]) and clear request active flag. If re-enable the stopped channel again, the remaining transfers will be processed. 0 = No effect. 1 = Stop PDMA channel 2 transfer.
[1]	**STOP1**	**PDMA Channel 1 Transfer Stop Control Register (Write Only)** User can set this bit to stop the PDMA channel 1 transfer. When user sets STOP1 bit, the PDMA controller will stop the on-going transfer, then clear the channel enable bit CHEN1 (PDMA_CHCTL [1]) and clear request active flag. If re-enable the stopped channel again, the remaining transfers will be processed. 0 = No effect. 1 = Stop PDMA channel 1 transfer.
[0]	**STOP0**	**PDMA Channel 0 Transfer Stop Control Register (Write Only)** User can set this bit to stop the PDMA channel 0 transfer. When user sets STOP0 bit, the PDMA controller will stop the on-going transfer, then clear the channel enable bit CHEN0 (PDMA_CHCTL [0]) and clear request active flag. If re-enable the stopped channel again, the remaining transfers will be processed. 0 = No effect. 1 = Stop PDMA channel 0 transfer.

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
SWREQ15	SWREQ14	SWREQ13	SWREQ12	SWREQ11	SWREQ10	SWREQ9	SWREQ8
7	6	5	4	3	2	1	0
SWREQ7	SWREQ6	SWREQ5	SWREQ4	SWREQ3	SWREQ2	SWREQ1	SWREQ0

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15]	**SWREQ15**	**PDMA Channel 15 Software Request Register (Write Only)** Set this bit to 1 to generate a software request to PDMA Channel 15. 0 = PDMA Channel 15 no effect. 1 = PDMA Channel 15 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.
[14]	**SWREQ14**	**PDMA Channel 14 Software Request Register (Write Only)** Set this bit to 1 to generate a software request to PDMA Channel 14. 0 = PDMA Channel 14 no effect. 1 = PDMA Channel 14 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.
[13]	**SWREQ13**	**PDMA Channel 13 Software Request Register (Write Only)** Set this bit to 1 to generate a software request to PDMA Channel 13. 0 = PDMA Channel 13 no effect. 1 = PDMA Channel 13 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.
[12]	**SWREQ12**	**PDMA Channel 12 Software Request Register (Write Only)** Set this bit to 1 to generate a software request to PDMA Channel 12.

Bits	Description	
		0 = PDMA Channel 12 no effect. 1 = PDMA Channel 12 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.
[11]	**SWREQ11**	**PDMA Channel 11 Software Request Register (Write Only)** Set this bit to 1 to generate a software request to PDMA Channel 11. 0 = PDMA Channel 11 no effect. 1 = PDMA Channel 11 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.
[10]	**SWREQ10**	**PDMA Channel 10 Software Request Register (Write Only)** Set this bit to 1 to generate a software request to PDMA Channel 10. 0 = PDMA Channel 10 no effect. 1 = PDMA Channel 10 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.
[9]	**SWREQ9**	**PDMA Channel 9 Software Request Register (Write Only)** Set this bit to 1 to generate a software request to PDMA Channel 9. 0 = PDMA Channel 9 no effect. 1 = PDMA Channel 9 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.
[8]	**SWREQ8**	**PDMA Channel 8 Software Request Register (Write Only)** Set this bit to 1 to generate a software request to PDMA Channel 8. 0 = PDMA Channel 8 no effect. 1 = PDMA Channel 8 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.
[7]	**SWREQ7**	**PDMA Channel 7 Software Request Register (Write Only)** Set this bit to 1 to generate a software request to PDMA Channel 7. 0 = PDMA Channel 7 no effect. 1 = PDMA Channel 7 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.
[6]	**SWREQ6**	**PDMA Channel 6 Software Request Register (Write Only)** Set this bit to 1 to generate a software request to PDMA Channel 6. 0 = PDMA Channel 6 no effect.

Bits	Description	
		1 = PDMA Channel 6 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.
[5]	**SWREQ5**	**PDMA Channel 5 Software Request Register (Write Only)** Set this bit to 1 to generate a software request to PDMA Channel 5. 0 = PDMA Channel 5 no effect. 1 = PDMA Channel 5 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.
[4]	**SWREQ4**	**PDMA Channel 4 Software Request Register (Write Only)** Set this bit to 1 to generate a software request to PDMA Channel 4. 0 = PDMA Channel 4 no effect. 1 = PDMA Channel 4 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.
[3]	**SWREQ3**	**PDMA Channel 3 Software Request Register (Write Only)** Set this bit to 1 to generate a software request to PDMA Channel 3. 0 = PDMA Channel 3 no effect. 1 = PDMA Channel 3 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.
[2]	**SWREQ2**	**PDMA Channel 2 Software Request Register (Write Only)** Set this bit to 1 to generate a software request to PDMA Channel 2. 0 = PDMA Channel 2 no effect. 1 = PDMA Channel 2 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.
[1]	**SWREQ1**	**PDMA Channel 1 Software Request Register (Write Only)** Set this bit to 1 to generate a software request to PDMA Channel 1. 0 = PDMA Channel 1 no effect. 1 = PDMA Channel 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.
[0]	**SWREQ0**	**PDMA Channel 0 Software Request Register (Write Only)** Set this bit to 1 to generate a software request to PDMA Channel 0. 0 = PDMA Channel 0 no effect. 1 = PDMA Channel 0 generate a software request.

Bits	Description	
		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
REQSTS15	REQSTS14	REQSTS13	REQSTS12	REQSTS11	REQSTS10	REQSTS9	REQSTS8
7	6	5	4	3	2	1	0
REQSTS7	REQSTS6	REQSTS5	REQSTS4	REQSTS3	REQSTS2	REQSTS1	REQSTS0

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15]	**REQSTS15**	**PDMA Channel 15 Request Status (Read Only)** This flag indicates whether channel 15 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 15 has no request. 1 = PDMA Channel 15 has a request. **Note:** If user stops or resets each PDMA transfer by setting PDMA_STOP or PDMA_CHRST register respectively, this bit will be cleared automatically after finishing current transfer.
[14]	**REQSTS14**	**PDMA Channel 14 Request Status (Read Only)** This flag indicates whether channel 14 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 14 has no request. 1 = PDMA Channel 14 has a request. **Note:** If user stops or resets each PDMA transfer by setting PDMA_STOP or PDMA_CHRST register respectively, this bit will be cleared automatically after finishing current transfer.
[13]	**REQSTS13**	**PDMA Channel 13 Request Status (Read Only)** This flag indicates whether channel 13 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 13 has no request. 1 = PDMA Channel 13 has a request. **Note:** If user stops or resets each PDMA transfer by setting PDMA_STOP or PDMA_CHRST register respectively, this bit will be cleared automatically after finishing current transfer.
[12]	**REQSTS12**	**PDMA Channel 12 Request Status (Read Only)**

Bits	Description	
		This flag indicates whether channel 12 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 12 has no request. 1 = PDMA Channel 12 has a request. **Note:** If user stops or resets each PDMA transfer by setting PDMA_STOP or PDMA_CHRST register respectively, this bit will be cleared automatically after finishing current transfer.
[11]	**REQSTS11**	**PDMA Channel 11 Request Status (Read Only)** This flag indicates whether channel 11 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 11 has no request. 1 = PDMA Channel 11 has a request. **Note:** If user stops or resets each PDMA transfer by setting PDMA_STOP or PDMA_CHRST register respectively, this bit will be cleared automatically after finishing current transfer.
[10]	**REQSTS10**	**PDMA Channel 10 Request Status (Read Only)** This flag indicates whether channel 10 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 10 has no request. 1 = PDMA Channel 10 has a request. **Note:** If user stops or resets each PDMA transfer by setting PDMA_STOP or PDMA_CHRST register respectively, this bit will be cleared automatically after finishing current transfer.
[9]	**REQSTS9**	**PDMA Channel 9 Request Status (Read Only)** This flag indicates whether channel 9 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 9 has no request. 1 = PDMA Channel 9 has a request. **Note1:** If user stops or resets each PDMA transfer by setting PDMA_STOP or PDMA_CHRST register respectively, this bit will be cleared automatically after finishing current transfer.
[8]	**REQSTS8**	**PDMA Channel 8 Request Status (Read Only)** This flag indicates whether channel 8 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 8 has no request. 1 = PDMA Channel 8 has a request. **Note:** If user stops or resets each PDMA transfer by setting PDMA_STOP or PDMA_CHRST register respectively, this bit will be cleared automatically after finishing current transfer.
[7]	**REQSTS7**	**PDMA Channel 7 Request Status (Read Only)** This flag indicates whether channel 7 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 7 has no request. 1 = PDMA Channel 7 has a request. **Note:** If user stops or resets each PDMA transfer by setting PDMA_STOP or PDMA_CHRST register respectively, this bit will be cleared automatically after finishing current transfer.

Bits	Description	
[6]	**REQSTS6**	**PDMA Channel 6 Request Status (Read Only)** This flag indicates whether channel 6 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 6 has no request. 1 = PDMA Channel 6 has a request. **Note:** If user stops or resets each PDMA transfer by setting PDMA_STOP or PDMA_CHRST register respectively, this bit will be cleared automatically after finishing current transfer.
[5]	**REQSTS5**	**PDMA Channel 5 Request Status (Read Only)** This flag indicates whether channel 5 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 5 has no request. 1 = PDMA Channel 5 has a request. **Note:** If user stops or resets each PDMA transfer by setting PDMA_STOP or PDMA_CHRST register respectively, this bit will be cleared automatically after finishing current transfer.
[4]	**REQSTS4**	**PDMA Channel 4 Request Status (Read Only)** This flag indicates whether channel 4 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 4 has no request. 1 = PDMA Channel 4 has a request. **Note:** If user stops or resets each PDMA transfer by setting PDMA_STOP or PDMA_CHRST register respectively, this bit will be cleared automatically after finishing current transfer.
[3]	**REQSTS3**	**PDMA Channel 3 Request Status (Read Only)** This flag indicates whether channel 3 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 3 has no request. 1 = PDMA Channel 3 has a request. **Note:** If user stops or resets each PDMA transfer by setting PDMA_STOP or PDMA_CHRST register respectively, this bit will be cleared automatically after finishing current transfer.
[2]	**REQSTS2**	**PDMA Channel 2 Request Status (Read Only)** This flag indicates whether channel 2 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 2 has no request. 1 = PDMA Channel 2 has a request. **Note:** If user stops or resets each PDMA transfer by setting PDMA_STOP or PDMA_CHRST register respectively, this bit will be cleared automatically after finishing current transfer.
[1]	**REQSTS1**	**PDMA Channel 1 Request Status (Read Only)** This flag indicates whether channel 1 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 1 has no request. 1 = PDMA Channel 1 has a request. **Note:** If user stops or resets each PDMA transfer by setting PDMA_STOP or

Bits	Description	
		PDMA_CHRST register respectively, this bit will be cleared automatically after finishing current transfer.
[0]	**REQSTS0**	**PDMA Channel 0 Request Status (Read Only)** This flag indicates whether channel 0 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 0 has no request. 1 = PDMA Channel 0 has a request. **Note:** If user stops or resets each PDMA transfer by setting PDMA_STOP or PDMA_CHRST 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
FPRISET15	FPRISET14	FPRISET13	FPRISET12	FPRISET11	FPRISET10	FPRISET9	FPRISET8
7	6	5	4	3	2	1	0
FPRISET7	FPRISET6	FPRISET5	FPRISET4	FPRISET3	FPRISET2	FPRISET1	FPRISET0

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15]	**FPRISET15**	**PDMA Channel 15 Fixed Priority Setting Register** Set this bit to 1 to enable fixed priority level. **Write Operation:** 0 = No effect. 1 = Set PDMA channel 15 to fixed priority channel. **Read Operation:** 0 = Corresponding PDMA channel 15 is round-robin priority. 1 = Corresponding PDMA channel 15 is fixed priority. **Note:** This field only set to fixed priority, clear fixed priority use PDMA_PRICLR register.
[14]	**FPRISET14**	**PDMA Channel 14 Fixed Priority Setting Register** Set this bit to 1 to enable fixed priority level. **Write Operation:** 0 = No effect. 1 = Set PDMA channel 14 to fixed priority channel. **Read Operation:** 0 = Corresponding PDMA channel 14 is round-robin priority. 1 = Corresponding PDMA channel 14 is fixed priority. **Note:** This field only set to fixed priority, clear fixed priority use PDMA_PRICLR register.
[13]	**FPRISET13**	**PDMA Channel 13 Fixed Priority Setting Register** Set this bit to 1 to enable fixed priority level. **Write Operation:** 0 = No effect. 1 = Set PDMA channel 13 to fixed priority channel. **Read Operation:** 0 = Corresponding PDMA channel 13 is round-robin priority.

Bits	Description	
		1 = Corresponding PDMA channel 13 is fixed priority. **Note:** This field only set to fixed priority, clear fixed priority use PDMA_PRICLR register.
[12]	**FPRISET12**	**PDMA Channel 12 Fixed Priority Setting Register** Set this bit to 1 to enable fixed priority level. **Write Operation:** 0 = No effect. 1 = Set PDMA channel 12 to fixed priority channel. **Read Operation:** 0 = Corresponding PDMA channel 12 is round-robin priority. 1 = Corresponding PDMA channel 12 is fixed priority. **Note:** This field only set to fixed priority, clear fixed priority use PDMA_PRICLR register.
[11]	**FPRISET11**	**PDMA Channel 10 Fixed Priority Setting Register** Set this bit to 1 to enable fixed priority level. **Write Operation:** 0 = No effect. 1 = Set PDMA channel 11 to fixed priority channel. **Read Operation:** 0 = Corresponding PDMA channel 11 is round-robin priority. 1 = Corresponding PDMA channel 11 is fixed priority. **Note:** This field only set to fixed priority, clear fixed priority use PDMA_PRICLR register.
[10]	**FPRISET10**	**PDMA Channel 10 Fixed Priority Setting Register** Set this bit to 1 to enable fixed priority level. **Write Operation:** 0 = No effect. 1 = Set PDMA channel 10 to fixed priority channel. **Read Operation:** 0 = Corresponding PDMA channel 10 is round-robin priority. 1 = Corresponding PDMA channel 10 is fixed priority. **Note:** This field only set to fixed priority, clear fixed priority use PDMA_PRICLR register.
[9]	**FPRISET9**	**PDMA Channel 9 Fixed Priority Setting Register** Set this bit to 1 to enable fixed priority level. **Write Operation:** 0 = No effect. 1 = Set PDMA channel 9 to fixed priority channel. **Read Operation:** 0 = Corresponding PDMA channel 9 is round-robin priority. 1 = Corresponding PDMA channel 9 is fixed priority. **Note:** This field only set to fixed priority, clear fixed priority use PDMA_PRICLR register.
[8]	**FPRISET8**	**PDMA Channel 8 Fixed Priority Setting Register** Set this bit to 1 to enable fixed priority level. **Write Operation:** 0 = No effect. 1 = Set PDMA channel 8 to fixed priority channel. **Read Operation:** 0 = Corresponding PDMA channel 8 is round-robin priority. 1 = Corresponding PDMA channel 8 is fixed priority.

Bits	Description	
		Note: This field only set to fixed priority, clear fixed priority use PDMA_PRICLR register.
[7]	**FPRISET7**	**PDMA Channel 7 Fixed Priority Setting Register** Set this bit to 1 to enable fixed priority level. **Write Operation:** 0 = No effect. 1 = Set PDMA channel 7 to fixed priority channel. **Read Operation:** 0 = Corresponding PDMA channel 7 is round-robin priority. 1 = Corresponding PDMA channel 7 is fixed priority. **Note:** This field only set to fixed priority, clear fixed priority use PDMA_PRICLR register.
[6]	**FPRISET6**	**PDMA Channel 6 Fixed Priority Setting Register** Set this bit to 1 to enable fixed priority level. **Write Operation:** 0 = No effect. 1 = Set PDMA channel 6 to fixed priority channel. **Read Operation:** 0 = Corresponding PDMA channel 6 is round-robin priority. 1 = Corresponding PDMA channel 6 is fixed priority. **Note:** This field only set to fixed priority, clear fixed priority use PDMA_PRICLR register.
[5]	**FPRISET5**	**PDMA Channel 5 Fixed Priority Setting Register** Set this bit to 1 to enable fixed priority level. **Write Operation:** 0 = No effect. 1 = Set PDMA channel 5 to fixed priority channel. **Read Operation:** 0 = Corresponding PDMA channel 5 is round-robin priority. 1 = Corresponding PDMA channel 5 is fixed priority. **Note:** This field only set to fixed priority, clear fixed priority use PDMA_PRICLR register.
[4]	**FPRISET4**	**PDMA Channel 4 Fixed Priority Setting Register** Set this bit to 1 to enable fixed priority level. **Write Operation:** 0 = No effect. 1 = Set PDMA channel 4 to fixed priority channel. **Read Operation:** 0 = Corresponding PDMA channel 4 is round-robin priority. 1 = Corresponding PDMA channel 4 is fixed priority. **Note:** This field only set to fixed priority, clear fixed priority use PDMA_PRICLR register.
[3]	**FPRISET3**	**PDMA Channel 3 Fixed Priority Setting Register** Set this bit to 1 to enable fixed priority level. **Write Operation:** 0 = No effect. 1 = Set PDMA channel 3 to fixed priority channel. **Read Operation:** 0 = Corresponding PDMA channel 3 is round-robin priority. 1 = Corresponding PDMA channel 3 is fixed priority. **Note:** This field only set to fixed priority, clear fixed priority use PDMA_PRICLR register.

Bits	Description	
[2]	**FPRISET2**	**PDMA Channel 2 Fixed Priority Setting Register** Set this bit to 1 to enable fixed priority level. **Write Operation:** 0 = No effect. 1 = Set PDMA channel 2 to fixed priority channel. **Read Operation:** 0 = Corresponding PDMA channel 2 is round-robin priority. 1 = Corresponding PDMA channel 2 is fixed priority. **Note:** This field only set to fixed priority, clear fixed priority use PDMA_PRICLR register.
[1]	**FPRISET1**	**PDMA Channel 1 Fixed Priority Setting Register** Set this bit to 1 to enable fixed priority level. **Write Operation:** 0 = No effect. 1 = Set PDMA channel 1 to fixed priority channel. **Read Operation:** 0 = Corresponding PDMA channel 1 is round-robin priority. 1 = Corresponding PDMA channel 1 is fixed priority. **Note:** This field only set to fixed priority, clear fixed priority use PDMA_PRICLR register.
[0]	**FPRISET0**	**PDMA Channel 0 Fixed Priority Setting Register** Set this bit to 1 to enable fixed priority level. **Write Operation:** 0 = No effect. 1 = Set PDMA channel 0 to fixed priority channel. **Read Operation:** 0 = Corresponding PDMA channel 0 is round-robin priority. 1 = Corresponding PDMA channel 0 is fixed priority. **Note:** This field only set to fixed priority, clear fixed priority use PDMA_PRICLR register.

PDMA Fix 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
FPRICLR15	FPRICLR14	FPRICLR13	FPRICLR12	FPRICLR11	FPRICLR10	FPRICLR9	FPRICLR8
7	6	5	4	3	2	1	0
FPRICLR7	FPRICLR6	FPRICLR5	FPRICLR4	FPRICLR3	FPRICLR2	FPRICLR1	FPRICLR0

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15]	**FPRICLR15**	**PDMA Channel 15 Fixed Priority Clear Register (Write Only)** Set this bit to 1 to clear fixed priority level. 0 = No effect. 1 = Clear PDMA channel 15 fixed priority setting. **Note:** User can read PDMA_PRISET register to know the channel priority.
[14]	**FPRICLR14**	**PDMA Channel 14 Fixed Priority Clear Register (Write Only)** Set this bit to 1 to clear fixed priority level. 0 = No effect. 1 = Clear PDMA channel 14 fixed priority setting. **Note:** User can read PDMA_PRISET register to know the channel priority.
[13]	**FPRICLR13**	**PDMA Channel 13 Fixed Priority Clear Register (Write Only)** Set this bit to 1 to clear fixed priority level. 0 = No effect. 1 = Clear PDMA channel 13 fixed priority setting. **Note:** User can read PDMA_PRISET register to know the channel priority.
[12]	**FPRICLR12**	**PDMA Channel 12 Fixed Priority Clear Register (Write Only)** Set this bit to 1 to clear fixed priority level. 0 = No effect. 1 = Clear PDMA channel 12 fixed priority setting. **Note:** User can read PDMA_PRISET register to know the channel priority.
[11]	**FPRICLR11**	**PDMA Channel 11 Fixed Priority Clear Register (Write Only)** Set this bit to 1 to clear fixed priority level. 0 = No effect. 1 = Clear PDMA channel 11 fixed priority setting.

Bits	Description	
		Note: User can read PDMA_PRISET register to know the channel priority.
[10]	**FPRICLR10**	**PDMA Channel 10 Fixed Priority Clear Register (Write Only)** Set this bit to 1 to clear fixed priority level. 0 = No effect. 1 = Clear PDMA channel 10 fixed priority setting. **Note:** User can read PDMA_PRISET register to know the channel priority.
[9]	**FPRICLR9**	**PDMA Channel 9 Fixed Priority Clear Register (Write Only)** Set this bit to 1 to clear fixed priority level. 0 = No effect. 1 = Clear PDMA channel 9 fixed priority setting. **Note:** User can read PDMA_PRISET register to know the channel priority.
[8]	**FPRICLR8**	**PDMA Channel 8 Fixed Priority Clear Register (Write Only)** Set this bit to 1 to clear fixed priority level. 0 = No effect. 1 = Clear PDMA channel 8 fixed priority setting. **Note:** User can read PDMA_PRISET register to know the channel priority.
[7]	**FPRICLR7**	**PDMA Channel 7 Fixed Priority Clear Register (Write Only)** Set this bit to 1 to clear fixed priority level. 0 = No effect. 1 = Clear PDMA channel 7 fixed priority setting. **Note:** User can read PDMA_PRISET register to know the channel priority.
[6]	**FPRICLR6**	**PDMA Channel 6 Fixed Priority Clear Register (Write Only)** Set this bit to 1 to clear fixed priority level. 0 = No effect. 1 = Clear PDMA channel 6 fixed priority setting. **Note:** User can read PDMA_PRISET register to know the channel priority.
[5]	**FPRICLR5**	**PDMA Channel 5 Fixed Priority Clear Register (Write Only)** Set this bit to 1 to clear fixed priority level. 0 = No effect. 1 = Clear PDMA channel 5 fixed priority setting. **Note:** User can read PDMA_PRISET register to know the channel priority.
[4]	**FPRICLR4**	**PDMA Channel 4 Fixed Priority Clear Register (Write Only)** Set this bit to 1 to clear fixed priority level. 0 = No effect. 1 = Clear PDMA channel 4 fixed priority setting. **Note:** User can read PDMA_PRISET register to know the channel priority.
[3]	**FPRICLR3**	**PDMA Channel 3 Fixed Priority Clear Register (Write Only)** Set this bit to 1 to clear fixed priority level. 0 = No effect. 1 = Clear PDMA channel 3 fixed priority setting. **Note:** User can read PDMA_PRISET register to know the channel priority.
[2]	**FPRICLR2**	**PDMA Channel 2 Fixed Priority Clear Register (Write Only)** Set this bit to 1 to clear fixed priority level. 0 = No effect.

Bits	Description	
		1 = Clear PDMA channel 2 fixed priority setting. **Note:** User can read PDMA_PRISET register to know the channel priority.
[1]	**FPRICLR1**	**PDMA Channel 1 Fixed Priority Clear Register (Write Only)** Set this bit to 1 to clear fixed priority level. 0 = No effect. 1 = Clear PDMA channel 1 fixed priority setting. **Note:** User can read PDMA_PRISET register to know the channel priority.
[0]	**FPRICLR0**	**PDMA Channel 0 Fixed Priority Clear Register (Write Only)** Set this bit to 1 to clear fixed priority level. 0 = No effect. 1 = Clear PDMA channel 0 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
INTEN15	INTEN14	INTEN13	INTEN12	INTEN11	INTEN10	INTEN9	INTEN8
7	6	5	4	3	2	1	0
INTEN7	INTEN6	INTEN5	INTEN4	INTEN3	INTEN2	INTEN1	INTEN0

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15]	**INTEN15**	**PDMA Channel 15 Interrupt Enable Register** This field is used for enabling PDMA channel 15 interrupt. 0 = PDMA channel 15 interrupt Disabled. 1 = PDMA channel 15 interrupt Enabled.
[14]	**INTEN14**	**PDMA Channel 14 Interrupt Enable Register** This field is used for enabling PDMA channel 14 interrupt. 0 = PDMA channel 14 interrupt Disabled. 1 = PDMA channel 14 interrupt Enabled.
[13]	**INTEN13**	**PDMA Channel 13 Interrupt Enable Register** This field is used for enabling PDMA channel 13 interrupt. 0 = PDMA channel 13 interrupt Disabled. 1 = PDMA channel 13 interrupt Enabled.
[12]	**INTEN12**	**PDMA Channel 0 Interrupt Enable Register** This field is used for enabling PDMA channel 12 interrupt. 0 = PDMA channel 12 interrupt Disabled. 1 = PDMA channel 12 interrupt Enabled.
[11]	**INTEN11**	**PDMA Channel 11 Interrupt Enable Register** This field is used for enabling PDMA channel 11 interrupt. 0 = PDMA channel 11 interrupt Disabled. 1 = PDMA channel 11 interrupt Enabled.
[10]	**INTEN10**	**PDMA Channel 10 Interrupt Enable Register** This field is used for enabling PDMA channel 10 interrupt. 0 = PDMA channel 10 interrupt Disabled. 1 = PDMA channel 10 interrupt Enabled.

Bits	Description	
[9]	**INTEN9**	**PDMA Channel 9 Interrupt Enable Register** This field is used for enabling PDMA channel 9 interrupt. 0 = PDMA channel 9 interrupt Disabled. 1 = PDMA channel 9 interrupt Enabled.
[8]	**INTEN8**	**PDMA Channel 8 Interrupt Enable Register** This field is used for enabling PDMA channel 8 interrupt. 0 = PDMA channel 8 interrupt Disabled. 1 = PDMA channel 8 interrupt Enabled.
[7]	**INTEN7**	**PDMA Channel 7 Interrupt Enable Register** This field is used for enabling PDMA channel 7 interrupt. 0 = PDMA channel 7 interrupt Disabled. 1 = PDMA channel 7 interrupt Enabled.
[6]	**INTEN6**	**PDMA Channel 6 Interrupt Enable Register** This field is used for enabling PDMA channel 6 interrupt. 0 = PDMA channel 6 interrupt Disabled. 1 = PDMA channel 6 interrupt Enabled.
[5]	**INTEN5**	**PDMA Channel 5 Interrupt Enable Register** This field is used for enabling PDMA channel 5 interrupt. 0 = PDMA channel 5 interrupt Disabled. 1 = PDMA channel 5 interrupt Enabled.
[4]	**INTEN4**	**PDMA Channel 4 Interrupt Enable Register** This field is used for enabling PDMA channel 4 interrupt. 0 = PDMA channel 4 interrupt Disabled. 1 = PDMA channel 4 interrupt Enabled.
[3]	**INTEN3**	**PDMA Channel 3 Interrupt Enable Register** This field is used for enabling PDMA channel 3 interrupt. 0 = PDMA channel 3 interrupt Disabled. 1 = PDMA channel 3 interrupt Enabled.
[2]	**INTEN2**	**PDMA Channel 2 Interrupt Enable Register** This field is used for enabling PDMA channel 2 interrupt. 0 = PDMA channel 2 interrupt Disabled. 1 = PDMA channel 2 interrupt Enabled.
[1]	**INTEN1**	**PDMA Channel 1 Interrupt Enable Register** This field is used for enabling PDMA channel 1 interrupt. 0 = PDMA channel 1 interrupt Disabled. 1 = PDMA channel 1 interrupt Enabled.
[0]	**INTEN0**	**PDMA Channel 0 Interrupt Enable Register** This field is used for enabling PDMA channel 0 interrupt. 0 = PDMA channel 0 interrupt Disabled. 1 = PDMA channel 0 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					ALIGNF	TDIF	ABTIF

Bits	Description	
[31:10]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[9]	**REQTOF1**	**Request Time-out Flag for Channel 1** This flag indicates that PDMA controller has waited peripheral request for a period defined by PDMA_TOC1, user can write 1 to clear these bits. 0 = No request time-out. 1 = Peripheral request time-out.
[8]	**REQTOF0**	**Request Time-out Flag for Channel 0** This flag indicates that PDMA controller has waited peripheral request for a period defined by PDMA_TOC0, user can write 1 to clear these bits. 0 = No request time-out. 1 = Peripheral request time-out.
[7:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**ALIGNF**	**Transfer Alignment Interrupt Flag (Read Only)** 0 = PDMA channel source address and destination address both follow transfer width setting. 1 = PDMA channel source address or destination address is not follow transfer width setting.
[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.

Bits	Description	
		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
ABTIF15	ABTIF14	ABTIF13	ABTIF12	ABTIF11	ABTIF10	ABTIF9	ABTIF8
7	6	5	4	3	2	1	0
ABTIF7	ABTIF6	ABTIF5	ABTIF4	ABTIF3	ABTIF2	ABTIF1	ABTIF0

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15]	**ABTIF15**	**PDMA Channel 15 Read/Write Target Abort Interrupt Status Flag** This bit indicates PDMA channel 15 has target abort error; User can write 1 to clear these bits. 0 = No AHB bus ERROR response received when channel 15 transfer. 1 = AHB bus ERROR response received when channel 15 transfer.
[14]	**ABTIF14**	**PDMA Channel 14 Read/Write Target Abort Interrupt Status Flag** This bit indicates PDMA channel 14 has target abort error; User can write 1 to clear these bits. 0 = No AHB bus ERROR response received when channel 14 transfer. 1 = AHB bus ERROR response received when channel 14 transfer.
[13]	**ABTIF13**	**PDMA Channel 13 Read/Write Target Abort Interrupt Status Flag** This bit indicates PDMA channel 13 has target abort error; User can write 1 to clear these bits. 0 = No AHB bus ERROR response received when channel 13 transfer. 1 = AHB bus ERROR response received when channel 13 transfer.
[12]	**ABTIF12**	**PDMA Channel 12 Read/Write Target Abort Interrupt Status Flag** This bit indicates PDMA channel 12 has target abort error; User can write 1 to clear these bits. 0 = No AHB bus ERROR response received when channel 12 transfer. 1 = AHB bus ERROR response received when channel 12 transfer.
[11]	**ABTIF11**	**PDMA Channel 11 Read/Write Target Abort Interrupt Status Flag** This bit indicates PDMA channel 11 has target abort error; User can write 1 to clear these bits. 0 = No AHB bus ERROR response received when channel 11 transfer. 1 = AHB bus ERROR response received when channel 11 transfer.

Bits	Description	
[10]	**ABTIF10**	**PDMA Channel 10 Read/Write Target Abort Interrupt Status Flag** This bit indicates PDMA channel 10 has target abort error; User can write 1 to clear these bits. 0 = No AHB bus ERROR response received when channel 10 transfer. 1 = AHB bus ERROR response received when channel 10 transfer.
[9]	**ABTIF9**	**PDMA Channel 9 Read/Write Target Abort Interrupt Status Flag** This bit indicates PDMA channel 9 has target abort error; User can write 1 to clear these bits. 0 = No AHB bus ERROR response received when channel 9 transfer. 1 = AHB bus ERROR response received when channel 9 transfer.
[8]	**ABTIF8**	**PDMA Channel 8 Read/Write Target Abort Interrupt Status Flag** This bit indicates PDMA channel 8 has target abort error; User can write 1 to clear these bits. 0 = No AHB bus ERROR response received when channel 8 transfer. 1 = AHB bus ERROR response received when channel 8 transfer.
[7]	**ABTIF7**	**PDMA Channel 7 Read/Write Target Abort Interrupt Status Flag** This bit indicates PDMA channel 7 has target abort error; User can write 1 to clear these bits. 0 = No AHB bus ERROR response received when channel 7 transfer. 1 = AHB bus ERROR response received when channel 7 transfer.
[6]	**ABTIF6**	**PDMA Channel 6 Read/Write Target Abort Interrupt Status Flag** This bit indicates PDMA channel 6 has target abort error; User can write 1 to clear these bits. 0 = No AHB bus ERROR response received when channel 6 transfer. 1 = AHB bus ERROR response received when channel 6 transfer.
[5]	**ABTIF5**	**PDMA Channel 5 Read/Write Target Abort Interrupt Status Flag** This bit indicates PDMA channel 5 has target abort error; User can write 1 to clear these bits. 0 = No AHB bus ERROR response received when channel 5 transfer. 1 = AHB bus ERROR response received when channel 5 transfer.
[4]	**ABTIF4**	**PDMA Channel 4 Read/Write Target Abort Interrupt Status Flag** This bit indicates PDMA channel 4 has target abort error; User can write 1 to clear these bits. 0 = No AHB bus ERROR response received when channel 4 transfer. 1 = AHB bus ERROR response received when channel 4 transfer.
[3]	**ABTIF3**	**PDMA Channel 3 Read/Write Target Abort Interrupt Status Flag** This bit indicates PDMA channel 3 has target abort error; User can write 1 to clear these bits. 0 = No AHB bus ERROR response received when channel 3 transfer. 1 = AHB bus ERROR response received when channel 3 transfer.
[2]	**ABTIF2**	**PDMA Channel 2 Read/Write Target Abort Interrupt Status Flag** This bit indicates PDMA channel 2 has target abort error; User can write 1 to clear these bits. 0 = No AHB bus ERROR response received when channel 2 transfer. 1 = AHB bus ERROR response received when channel 2 transfer.
[1]	**ABTIF1**	**PDMA Channel 1 Read/Write Target Abort Interrupt Status Flag** This bit indicates PDMA channel 1 has target abort error; User can write 1 to clear these

Bits	Description	
		bits. 0 = No AHB bus ERROR response received when channel 1 transfer. 1 = AHB bus ERROR response received when channel 1 transfer.
[0]	**ABTIF0**	**PDMA Channel 0 Read/Write Target Abort Interrupt Status Flag** This bit indicates PDMA channel 0 has target abort error; User can write 1 to clear these bits. 0 = No AHB bus ERROR response received when channel 0 transfer. 1 = AHB bus ERROR response received when channel 0 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
TDIF15	TDIF14	TDIF13	TDIF12	TDIF11	TDIF10	TDIF9	TDIF8
7	6	5	4	3	2	1	0
TDIF7	TDIF6	TDIF5	TDIF4	TDIF3	TDIF2	TDIF1	TDIF0

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15]	**TDIF15**	**PDMA Channel 15 Transfer Done Flag Register** This bit indicates PDMA channel 15 transfer has been finished or not, user can write 1 to clear this bits. 0 = PDMA channel 15 transfer has not finished. 1 = PDMA channel 15 has finished transmission.
[14]	**TDIF14**	**PDMA Channel 14 Transfer Done Flag Register** This bit indicates PDMA channel 14 transfer has been finished or not, user can write 1 to clear this bits. 0 = PDMA channel 14 transfer has not finished. 1 = PDMA channel 14 has finished transmission.
[13]	**TDIF13**	**PDMA Channel 13 Transfer Done Flag Register** This bit indicates PDMA channel 13 transfer has been finished or not, user can write 1 to clear this bits. 0 = PDMA channel 13 transfer has not finished. 1 = PDMA channel 13 has finished transmission.
[12]	**TDIF12**	**PDMA Channel 12 Transfer Done Flag Register** This bit indicates PDMA channel 12 transfer has been finished or not, user can write 1 to clear this bits. 0 = PDMA channel 12 transfer has not finished. 1 = PDMA channel 12 has finished transmission.
[11]	**TDIF11**	**PDMA Channel 11 Transfer Done Flag Register** This bit indicates PDMA channel 11 transfer has been finished or not, user can write 1 to clear this bits. 0 = PDMA channel 11 transfer has not finished. 1 = PDMA channel 11 has finished transmission.

Bits	Description	
[10]	**TDIF10**	**PDMA Channel 10 Transfer Done Flag Register** This bit indicates PDMA channel 10 transfer has been finished or not, user can write 1 to clear this bits. 0 = PDMA channel 10 transfer has not finished. 1 = PDMA channel 10 has finished transmission.
[9]	**TDIF9**	**PDMA Channel 9 Transfer Done Flag Register** This bit indicates PDMA channel 9 transfer has been finished or not, user can write 1 to clear this bits. 0 = PDMA channel 9 transfer has not finished. 1 = PDMA channel 9 has finished transmission.
[8]	**TDIF8**	**PDMA Channel 8 Transfer Done Flag Register** This bit indicates PDMA channel 8 transfer has been finished or not, user can write 1 to clear this bits. 0 = PDMA channel 8 transfer has not finished. 1 = PDMA channel 8 has finished transmission.
[7]	**TDIF7**	**PDMA Channel 7 Transfer Done Flag Register** This bit indicates PDMA channel 7 transfer has been finished or not, user can write 1 to clear this bits. 0 = PDMA channel 7 transfer has not finished. 1 = PDMA channel 7 has finished transmission.
[6]	**TDIF6**	**PDMA Channel 6 Transfer Done Flag Register** This bit indicates PDMA channel 6 transfer has been finished or not, user can write 1 to clear this bits. 0 = PDMA channel 6 transfer has not finished. 1 = PDMA channel 6 has finished transmission.
[5]	**TDIF5**	**PDMA Channel 5 Transfer Done Flag Register** This bit indicates PDMA channel 5 transfer has been finished or not, user can write 1 to clear this bits. 0 = PDMA channel 5 transfer has not finished. 1 = PDMA channel 5 has finished transmission.
[4]	**TDIF4**	**PDMA Channel 4 Transfer Done Flag Register** This bit indicates PDMA channel 4 transfer has been finished or not, user can write 1 to clear this bits. 0 = PDMA channel 4 transfer has not finished. 1 = PDMA channel 4 has finished transmission.
[3]	**TDIF3**	**PDMA Channel 3 Transfer Done Flag Register** This bit indicates PDMA channel 3 transfer has been finished or not, user can write 1 to clear this bits. 0 = PDMA channel 3 transfer has not finished. 1 = PDMA channel 3 has finished transmission.
[2]	**TDIF2**	**PDMA Channel 2 Transfer Done Flag Register** This bit indicates PDMA channel 2 transfer has been finished or not, user can write 1 to clear this bits. 0 = PDMA channel 2 transfer has not finished. 1 = PDMA channel 2 has finished transmission.
[1]	**TDIF1**	**PDMA Channel 1 Transfer Done Flag Register** This bit indicates PDMA channel 1 transfer has been finished or not, user can write 1 to

Bits	Description	
		clear this bits. 0 = PDMA channel 1 transfer has not finished. 1 = PDMA channel 1 has finished transmission.
[0]	**TDIF0**	**PDMA Channel 0 Transfer Done Flag Register** This bit indicates PDMA channel 0 transfer has been finished or not, user can write 1 to clear this bits. 0 = PDMA channel 0 transfer has not finished. 1 = PDMA channel 0 has finished transmission.

PDMA Transfer Alignment Status Register (PDMA_ALIGN)

Register	Offset	R/W	Description	Reset Value
PDMA_ALIGN	PDMA_BA + 0x428	R/W	PDMA Transfer Alignment 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
ALIGN15	ALIGN14	ALIGN13	ALIGN12	ALIGN11	ALIGN10	ALIGN9	ALIGN8
7	6	5	4	3	2	1	0
ALIGN7	ALIGN6	ALIGN5	ALIGN4	ALIGN3	ALIGN2	ALIGN1	ALIGN0

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15]	**ALIGN15**	**PDMA Channel 15 Transfer Alignment Flag Register** 0 = PDMA channel 15 source address and destination address both follow transfer width setting. 1 = PDMA channel 15 source address or destination address is not follow transfer width setting. **Note:** Software can write 1 to clear this bit.
[14]	**ALIGN14**	**PDMA Channel 14 Transfer Alignment Flag Register** 0 = PDMA channel 14 source address and destination address both follow transfer width setting. 1 = PDMA channel 14 source address or destination address is not follow transfer width setting. **Note:** Software can write 1 to clear this bit.
[13]	**ALIGN13**	**PDMA Channel 13 Transfer Alignment Flag Register** 0 = PDMA channel 13 source address and destination address both follow transfer width setting. 1 = PDMA channel 13 source address or destination address is not follow transfer width setting. **Note:** Software can write 1 to clear this bit.
[12]	**ALIGN12**	**PDMA Channel 12 Transfer Alignment Flag Register** 0 = PDMA channel 12 source address and destination address both follow transfer width setting. 1 = PDMA channel 12 source address or destination address is not follow transfer width setting. **Note:** Software can write 1 to clear this bit.
[11]	**ALIGN11**	**PDMA Channel 11 Transfer Alignment Flag Register** 0 = PDMA channel 11 source address and destination address both follow transfer width setting.

Bits	Description	
		1 = PDMA channel 11 source address or destination address is not follow transfer width setting. **Note:** Software can write 1 to clear this bit.
[10]	**ALIGN10**	**PDMA Channel 10 Transfer Alignment Flag Register** 0 = PDMA channel 10 source address and destination address both follow transfer width setting. 1 = PDMA channel 10 source address or destination address is not follow transfer width setting. **Note:** Software can write 1 to clear this bit.
[9]	**ALIGN9**	**PDMA Channel 9 Transfer Alignment Flag Register** 0 = PDMA channel 9 source address and destination address both follow transfer width setting. 1 = PDMA channel 9 source address or destination address is not follow transfer width setting. **Note:** Software can write 1 to clear this bit.
[8]	**ALIGN8**	**PDMA Channel 8 Transfer Alignment Flag Register** 0 = PDMA channel 8 source address and destination address both follow transfer width setting. 1 = PDMA channel 8 source address or destination address is not follow transfer width setting. **Note:** Software can write 1 to clear this bit.
[7]	**ALIGN7**	**PDMA Channel 7 Transfer Alignment Flag Register** 0 = PDMA channel 7 source address and destination address both follow transfer width setting. 1 = PDMA channel 7 source address or destination address is not follow transfer width setting. **Note:** Software can write 1 to clear this bit.
[6]	**ALIGN6**	**PDMA Channel 6 Transfer Alignment Flag Register** 0 = PDMA channel 6 source address and destination address both follow transfer width setting. 1 = PDMA channel 6 source address or destination address is not follow transfer width setting. **Note:** Software can write 1 to clear this bit.
[5]	**ALIGN5**	**PDMA Channel 5 Transfer Alignment Flag Register** 0 = PDMA channel 5 source address and destination address both follow transfer width setting. 1 = PDMA channel 5 source address or destination address is not follow transfer width setting. **Note:** Software can write 1 to clear this bit.
[4]	**ALIGN4**	**PDMA Channel 4 Transfer Alignment Flag Register** 0 = PDMA channel 4 source address and destination address both follow transfer width setting. 1 = PDMA channel 4 source address or destination address is not follow transfer width setting. **Note:** Software can write 1 to clear this bit.
[3]	**ALIGN3**	**PDMA Channel 3 Transfer Alignment Flag Register** 0 = PDMA channel 3 source address and destination address both follow transfer width setting. 1 = PDMA channel 3 source address or destination address is not follow transfer width setting.

Bits	Description	
		Note: Software can write 1 to clear this bit.
[2]	**ALIGN2**	**PDMA Channel 2 Transfer Alignment Flag Register** 0 = PDMA channel 2 source address and destination address both follow transfer width setting. 1 = PDMA channel 2 source address or destination address is not follow transfer width setting. **Note:** Software can write 1 to clear this bit.
[1]	**ALIGN1**	**PDMA Channel 1 Transfer Alignment Flag Register** 0 = PDMA channel 1 source address and destination address both follow transfer width setting. 1 = PDMA channel 1 source address or destination address is not follow transfer width setting. **Note:** Software can write 1 to clear this bit.
[0]	**ALIGN0**	**PDMA Channel 0 Transfer Alignment Flag Register** 0 = PDMA channel 0 source address and destination address both follow transfer width setting. 1 = PDMA channel 0 source address or destination address is not follow transfer width setting. **Note:** Software can write 1 to clear this bit.

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
TXACTF15	TXACTF14	TXACTF13	TXACTF12	TXACTF11	TXACTF10	TXACTF9	TXACTF8
7	6	5	4	3	2	1	0
TXACTF7	TXACTF6	TXACTF5	TXACTF4	TXACTF3	TXACTF2	TXACTF1	TXACTF0

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15]	**TXACTF15**	**PDMA Channel 15 Transfer on Active Flag Register (Read Only)** 0 = PDMA channel 15 is not finished. 1 = PDMA channel 15 is in active.
[14]	**TXACTF14**	**PDMA Channel 14 Transfer on Active Flag Register (Read Only)** 0 = PDMA channel 14 is not finished. 1 = PDMA channel 14 is in active.
[13]	**TXACTF13**	**PDMA Channel 13 Transfer on Active Flag Register (Read Only)** 0 = PDMA channel 13 is not finished. 1 = PDMA channel 13 is in active.
[12]	**TXACTF12**	**PDMA Channel 12 Transfer on Active Flag Register (Read Only)** 0 = PDMA channel 12 is not finished. 1 = PDMA channel 12 is in active.
[11]	**TXACTF11**	**PDMA Channel 11 Transfer on Active Flag Register (Read Only)** 0 = PDMA channel 11 is not finished. 1 = PDMA channel 11 is in active.
[10]	**TXACTF10**	**PDMA Channel 10 Transfer on Active Flag Register (Read Only)** 0 = PDMA channel 10 is not finished. 1 = PDMA channel 10 is in active.
[9]	**TXACTF9**	**PDMA Channel 9 Transfer on Active Flag Register (Read Only)** 0 = PDMA channel 9 is not finished. 1 = PDMA channel 9 is in active.
[8]	**TXACTF8**	**PDMA Channel 8 Transfer on Active Flag Register (Read Only)** 0 = PDMA channel 8 is not finished.

Bits	Description	
		1 = PDMA channel 8 is in active.
[7]	**TXACTF7**	**PDMA Channel 7 Transfer on Active Flag Register (Read Only)** 0 = PDMA channel 7 is not finished. 1 = PDMA channel 7 is in active.
[6]	**TXACTF6**	**PDMA Channel 6 Transfer on Active Flag Register (Read Only)** 0 = PDMA channel 6 is not finished. 1 = PDMA channel 6 is in active.
[5]	**TXACTF5**	**PDMA Channel 5 Transfer on Active Flag Register (Read Only)** 0 = PDMA channel 5 is not finished. 1 = PDMA channel 5 is in active.
[4]	**TXACTF4**	**PDMA Channel 4 Transfer on Active Flag Register (Read Only)** 0 = PDMA channel 4 is not finished. 1 = PDMA channel 4 is in active.
[3]	**TXACTF3**	**PDMA Channel 3 Transfer on Active Flag Register (Read Only)** 0 = PDMA channel 3 is not finished. 1 = PDMA channel 3 is in active.
[2]	**TXACTF2**	**PDMA Channel 2 Transfer on Active Flag Register (Read Only)** 0 = PDMA channel 2 is not finished. 1 = PDMA channel 2 is in active.
[1]	**TXACTF1**	**PDMA Channel 1 Transfer on Active Flag Register (Read Only)** 0 = PDMA channel 1 is not finished. 1 = PDMA channel 1 is in active.
[0]	**TXACTF0**	**PDMA Channel 0 Transfer on Active Flag Register (Read Only)** 0 = PDMA channel 0 is not finished. 1 = PDMA channel 0 is in 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. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.
[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

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						TOUTIEN1	TOUTIEN0

Bits	Description	
[31:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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_NEXT. **Note:** Only useful in Scatter-Gather mode.
[15:0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

PDMA Time-out Period Counter Register 0 (PDMA_TOC0_1)

Register	Offset	R/W	Description	Reset Value
PDMA_TOC0_1	PDMA_BA + 0x440	R/W	PDMA Time-out Counter Ch1 and Ch0 Register	0xFFFF_FFFF

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 the setting of TOUTPSC1.
[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 the setting of TOUTPSC0.

PDMA Channel Reset Register (PDMA_CHRST)

Register	Offset	R/W	Description	Reset Value
PDMA_CHRST	PDMA_BA + 0x460	R/W	PDMA Channel Reset Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
CH15RST	CH14RST	CH13RST	CH12RST	CH11RST	CH10RST	CH9RST	CH8RST
7	6	5	4	3	2	1	0
CH7RST	CH6RST	CH5RST	CH4RST	CH3RST	CH2RST	CH1RST	CH0RST

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15]	**CH15RST**	**Channel 15 Reset** 0 = corresponding channel 15 not reset. 1 = corresponding channel 15 is reset. **Note 1:** This bit will be cleared automatically after finishing reset. **Note 2 :** Set this bit to 1 will also clear corresponding bit of PDMA_CHCTL.
[14]	**CH14RST**	**Channel 14 Reset** 0 = corresponding channel 14 not reset. 1 = corresponding channel 14 is reset. **Note 1:** This bit will be cleared automatically after finishing reset. **Note 2 :** Set this bit to 1 will also clear corresponding bit of PDMA_CHCTL.
[13]	**CH13RST**	**Channel 13 Reset** 0 = corresponding channel 13 not reset. 1 = corresponding channel 13 is reset. **Note 1 :** This bit will be cleared automatically after finishing reset. **Note 2 :** Set this bit to 1 will also clear corresponding bit of PDMA_CHCTL.
[12]	**CH12RST**	**Channel 12 Reset** 0 = corresponding channel 12 not reset. 1 = corresponding channel 12 is reset. **Note 1:** This bit will be cleared automatically after finishing reset. **Note 2 :** Set this bit to 1 will also clear corresponding bit of PDMA_CHCTL.
[11]	**CH11RST**	**Channel 11 Reset** 0 = corresponding channel 11 not reset. 1 = corresponding channel 11 is reset. **Note 1:** This bit will be cleared automatically after finishing reset.

Bits	Description	
		Note 2 : Set this bit to 1 will also clear corresponding bit of PDMA_CHCTL.
[10]	**CH10RST**	**Channel 10 Reset** 0 = corresponding channel 10 not reset. 1 = corresponding channel 10 is reset. **Note 1:** This bit will be cleared automatically after finishing reset. **Note 2 :** Set this bit to 1 will also clear corresponding bit of PDMA_CHCTL.
[9]	**CH9RST**	**Channel 9 Reset** 0 = corresponding channel 9 not reset. 1 = corresponding channel 9 is reset. **Note 1:** This bit will be cleared automatically after finishing reset. **Note 2 :** Set this bit to 1 will also clear corresponding bit of PDMA_CHCTL.
[8]	**CH8RST**	**Channel 8 Reset** 0 = corresponding channel 8 not reset. 1 = corresponding channel 8 is reset. **Note 1:** This bit will be cleared automatically after finishing reset. **Note 2 :** Set this bit to 1 will also clear corresponding bit of PDMA_CHCTL.
[7]	**CH7RST**	**Channel 7 Reset** 0 = corresponding channel 7 not reset. 1 = corresponding channel 7 is reset. **Note 1:** This bit will be cleared automatically after finishing reset. **Note 2 :** Set this bit to 1 will also clear corresponding bit of PDMA_CHCTL.
[6]	**CH6RST**	**Channel 6 Reset** 0 = corresponding channel 6 not reset. 1 = corresponding channel 6 is reset. **Note 1:** This bit will be cleared automatically after finishing reset. **Note 2 :** Set this bit to 1 will also clear corresponding bit of PDMA_CHCTL.
[5]	**CH5RST**	**Channel 5 Reset** 0 = corresponding channel 5 not reset. 1 = corresponding channel 5 is reset. **Note 1:** This bit will be cleared automatically after finishing reset. **Note 2 :** Set this bit to 1 will also clear corresponding bit of PDMA_CHCTL.
[4]	**CH4RST**	**Channel 4 Reset** 0 = corresponding channel 4 not reset. 1 = corresponding channel 4 is reset. **Note 1:** This bit will be cleared automatically after finishing reset. **Note 2 :** Set this bit to 1 will also clear corresponding bit of PDMA_CHCTL.
[3]	**CH3RST**	**Channel 3 Reset** 0 = corresponding channel 3 not reset. 1 = corresponding channel 3 is reset. **Note 1:** This bit will be cleared automatically after finishing reset. **Note 2 :** Set this bit to 1 will also clear corresponding bit of PDMA_CHCTL.
[2]	**CH2RST**	**Channel 2 Reset** 0 = corresponding channel 2 not reset. 1 = corresponding channel 2 is reset.

Bits	Description	
		Note 1: This bit will be cleared automatically after finishing reset. **Note 2 :** Set this bit to 1 will also clear corresponding bit of PDMA_CHCTL.
[1]	**CH1RST**	**Channel 1 Reset** 0 = corresponding channel 1 not reset. 1 = corresponding channel 1 is reset. **Note 1:** This bit will be cleared automatically after finishing reset. **Note 2 :** Set this bit to 1 will also clear corresponding bit of PDMA_CHCTL.
[0]	**CH0RST**	**Channel 0 Reset** 0 = corresponding channel 0 not reset. 1 = corresponding channel 0 is reset. **Note 1:** This bit will be cleared automatically after finishing reset. **Note 2 :** Set this bit to 1 will also clear corresponding bit of PDMA_CHCTL.

PDMA Request Source Select Register 0 (PDMA_REQSEL0_3)

Register	Offset	R/W	Description	Reset Value
PDMA_REQSEL0_3	PDMA_BA + 0x480	R/W	PDMA Request Source Select Register 0	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. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.
[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. 4 = Channel connects to UART0_TX.

Bits	Description	
		5 = Channel connects to UART0_RX. 20 = Channel connects to SPI0_TX. 21 = Channel connects to SPI0_RX. 22 = Channel connects to SPI1_TX. 23 = Channel connects to SPI1_RX. 24 = Channel connects to SPI2_TX. 25 = Channel connects to SPI2_RX. 27 = Channel connects to DMIC_RX. 28 = Channel connects to DPWM_TX. 32 = Channel connects to PWM0_P1_RX. 33 = Channel connects to PWM0_P2_RX. 34 = Channel connects to PWM0_P3_RX. 44 = Channel connects to I2S0_TX. 45 = Channel connects to I2S0_RX. 46 = Channel connects to TMR0. 47 = Channel connects to TMR1. 48 = Channel connects to TMR2. 49 = Channel connects to TMR3. 50 = Channel connects to ADC_RX. Others = Reserved. Do not use. **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 Request Source Select Register 1 (PDMA_REQSEL4_7)

Register	Offset	R/W	Description	Reset Value
PDMA_REQSEL4_7	PDMA_BA + 0x484	R/W	PDMA Request Source Select Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		REQSRC7					
23	22	21	20	19	18	17	16
Reserved		REQSRC6					
15	14	13	12	11	10	9	8
Reserved		REQSRC5					
7	6	5	4	3	2	1	0
Reserved		REQSRC4					

Bits	Description	
[31:30]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[29:24]	**REQSRC7**	**Channel 7 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 7. User can configure the peripheral setting by REQSRC7. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[23:22]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[21:16]	**REQSRC6**	**Channel 6 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 6. User can configure the peripheral setting by REQSRC6. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[15:14]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[13:8]	**REQSRC5**	**Channel 5 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 5. User can configure the peripheral setting by REQSRC5. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[7:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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.

PDMA Request Source Select Register 2 (PDMA_REQSEL8_11)

Register	Offset	R/W	Description	Reset Value
PDMA_REQSEL8_11	PDMA_BA + 0x488	R/W	PDMA Request Source Select Register 2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		**REQSRC11**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		**REQSRC10**					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**REQSRC9**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**REQSRC8**					

Bits	Description	
[31:30]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[29:24]	**REQSRC11**	**Channel 11 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 11. User can configure 1the peripheral setting by REQSRC11. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[23:22]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[21:16]	**REQSRC10**	**Channel 10 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 10. User can configure the peripheral setting by REQSRC10. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[15:14]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[13:8]	**REQSRC9**	**Channel 9 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 9. User can configure the peripheral setting by REQSRC9. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[7:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5:0]	**REQSRC8**	**Channel 8 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 8. User can configure the peripheral setting by REQSRC8. **Note:** The channel configuration is the same as REQSRC.0 field. Please refer to the explanation of REQSRC0.

PDMA Request Source Select Register 3 (PDMA_REQSEL12_15)

Register	Offset	R/W	Description	Reset Value
PDMA_REQSEL12_15	PDMA_BA + 0x48C	R/W	PDMA Request Source Select Register 3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		REQSRC15					
23	22	21	20	19	18	17	16
Reserved		REQSRC14					
15	14	13	12	11	10	9	8
Reserved		REQSRC13					
7	6	5	4	3	2	1	0
Reserved		REQSRC12					

Bits	Description	
[31:30]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[29:24]	**REQSRC15**	**Channel 7 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 15. User can configure the peripheral setting by REQSRC15. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[23:22]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[21:16]	**REQSRC14**	**Channel 6 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 14. User can configure the peripheral setting by REQSRC14. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[15:14]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[13:8]	**REQSRC13**	**Channel 5 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 13. User can configure the peripheral setting by REQSRC13. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[7:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5:0]	**REQSRC12**	**Channel 4 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 12. User can configure the peripheral setting by REQSRC12. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.

PDMA Stride Transfer Count Register n (PDMA_STCRn)

Register	Offset	R/W	Description	Reset Value
PDMA_STCR0	PDMA_BA + 0x500	R/W	Stride Transfer Count Register of PDMA Channel 0	0x0000_0000
PDMA_STCR1	PDMA_BA + 0x508	R/W	Stride Transfer Count Register of PDMA Channel 1	0x0000_0000
PDMA_STCR2	PDMA_BA + 0x510	R/W	Stride Transfer Count Register of PDMA Channel 2	0x0000_0000
PDMA_STCR3	PDMA_BA + 0x518	R/W	Stride Transfer Count Register of PDMA Channel 3	0x0000_0000
PDMA_STCR4	PDMA_BA + 0x520	R/W	Stride Transfer Count Register of PDMA Channel 4	0x0000_0000
PDMA_STCR5	PDMA_BA + 0x528	R/W	Stride Transfer Count Register of PDMA Channel 5	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**STC**	**PDMA Stride Transfer Count** The 16-bit register defines the stride transfer count of each row. The stride transfer count = STC+1.

PDMA Address Stride Offset Control Register n (PDMA_ASOCRn)

Register	Offset	R/W	Description	Reset Value
PDMA_ASOCR0	PDMA_BA + 0x504	R/W	Address Stride Offset Register of PDMA Channel 0	0x0000_0000
PDMA_ASOCR1	PDMA_BA + 0x50C	R/W	Address Stride Offset Register of PDMA Channel 1	0x0000_0000
PDMA_ASOCR2	PDMA_BA + 0x514	R/W	Address Stride Offset Register of PDMA Channel 2	0x0000_0000
PDMA_ASOCR3	PDMA_BA + 0x51C	R/W	Address Stride Offset Register of PDMA Channel 3	0x0000_0000
PDMA_ASOCR4	PDMA_BA + 0x524	R/W	Address Stride Offset Register of PDMA Channel 4	0x0000_0000
PDMA_ASOCR5	PDMA_BA + 0x52C	R/W	Address Stride Offset Register of PDMA Channel 5	0x0000_0000

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

Bits	Description	
[31:16]	**DASOL**	**PDMA Destination Address Stride Offset Length** The 16-bit register defines the destination address stride transfer offset count of each row.
[15:0]	**SASOL**	**PDMA Source Address Stride Offset Length** The 16-bit register defines the source address stride transfer offset count of each row.

6.7 Timer Controller (TMR)

6.7.1 Overview

The Timer controller contains four 32-bit multi-functional timers, Timer0, 1, 2 and 3. These timers can be used to count, or time external events that drive the Timer input pins.

Or the four timers can be configured as four PWM generators; each can support two PWM output channels in independent or complementary mode. The output state of PWM output pin can be control by pin mask, polarity, and dead-time generator.

6.7.2 Features

6.7.2.1 Timer Function Features

- Four sets of 32-bit timers, each timer equips one 24-bit up counter and one 8-bit prescaler 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, EADC and PDMA function
- Supports Inter-Timer trigger mode

6.7.2.2 PWM Function Features

- Supports maximum clock frequency up to maximum PCLK
- Supports independent mode for PWM generator with two output channels
- Supports complementary mode for PWM generator with paired PWM output channel
  - 12-bit dead-time insertion with 12-bit prescaler
- Supports 12-bit prescaler from 1 to 4096
- Supports 16-bit PWM counter
  - Up, down and up-down count operation type
  - One-shot or auto-reload counter operation mode
- Supports mask function and tri-state enable for each PWM output pin
- Supports interrupt on the following events:
  - PWM zero point, period point, up-count compared or down-count compared point events
- Supports trigger ADC on the following events:

- PWM zero point, period, zero or period point, up-count compared or down-count compared point events

6.7.3 Block Diagram

Timer Controller block diagram is shown below.

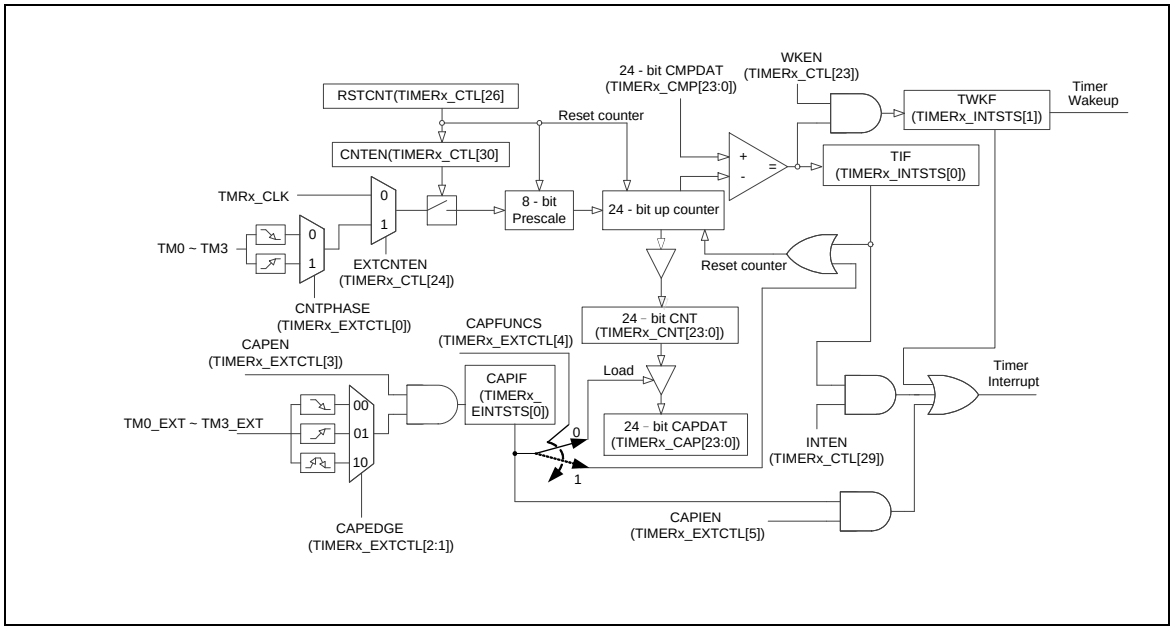


Figure 6.7-1 Timer Controller Block Diagram

Timer clock source can be from:

- HXT
- LXT
- PCLK1
- External clock on TMx pin, x=0, 1, 2 or 3.
- LIRC
- HIRC

Note: in PWM mode clock source selection is different, as explanation follows.

.

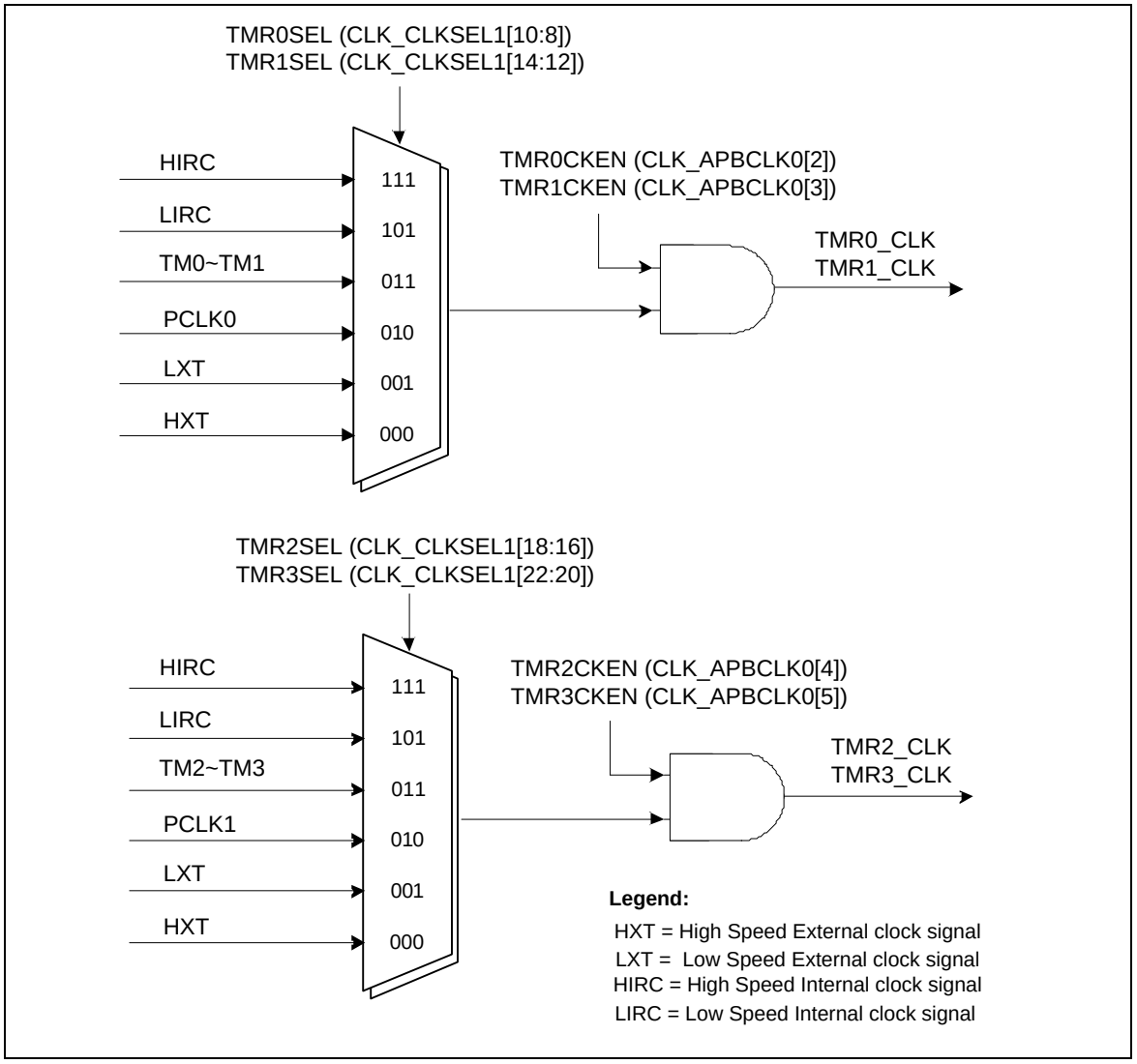


Figure 6.7-2 Clock Source of Timer Controller

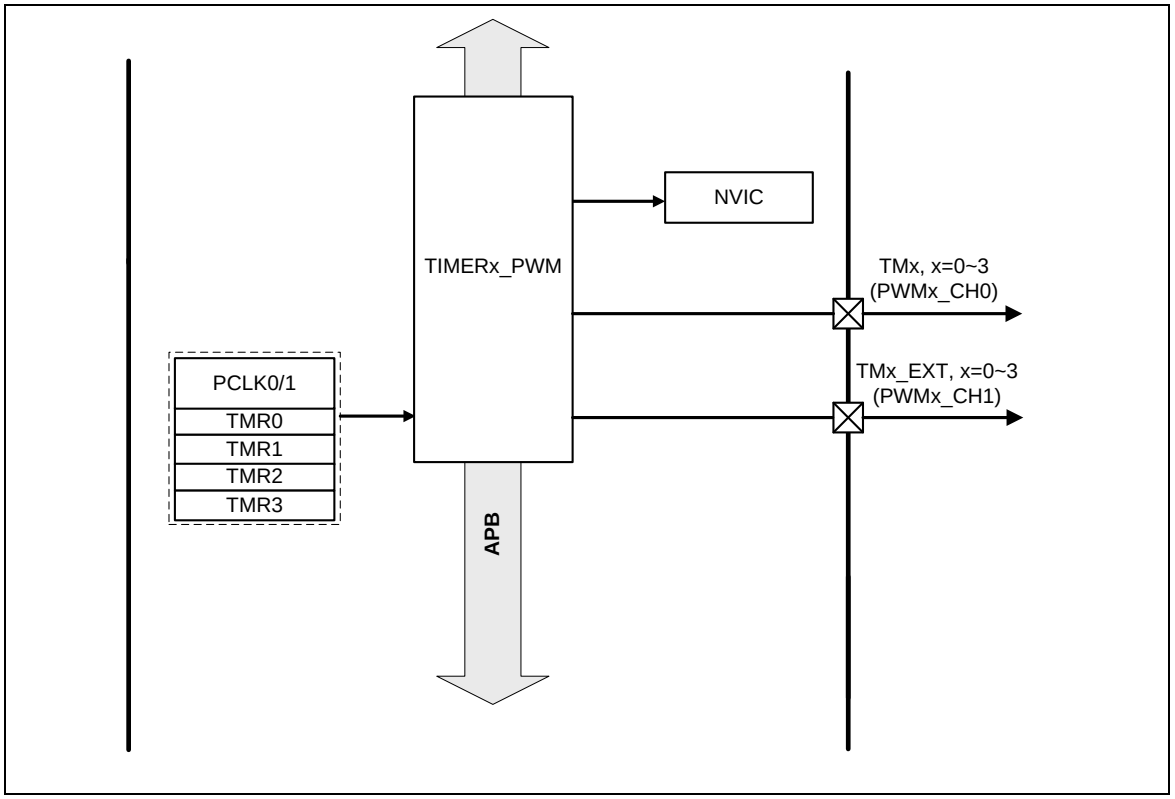


Figure 6.7-3 PWM Generator Overview Block Diagram

In PWM mode, the timer clock source, i.e. now the PWM system clock, TMR0_CLK and TMR1_CLK clock sources are fixed to be from PCLK0; TMR2_CLK and TMR3_CLK clock sources are fixed to be from PCLK1, shown in Figure 6.7-4 .

Further, the PWM counter (TIMERx_PWMCLK) clock source can be selected from TMRx_CLK (PWM system clock) or Timer interrupt events (TMRx_INT) as shown in Figure 6.7-5.

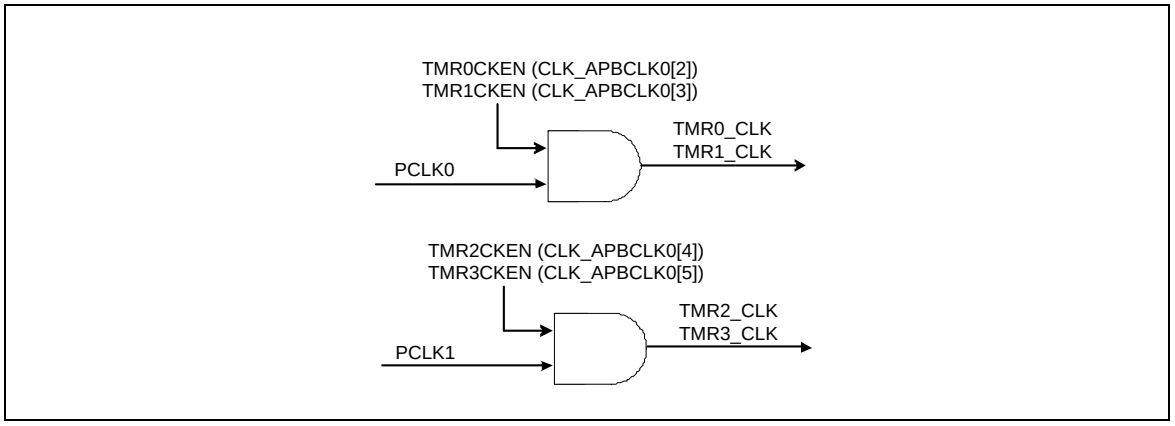


Figure 6.7-4 PWM System Clock Source Control

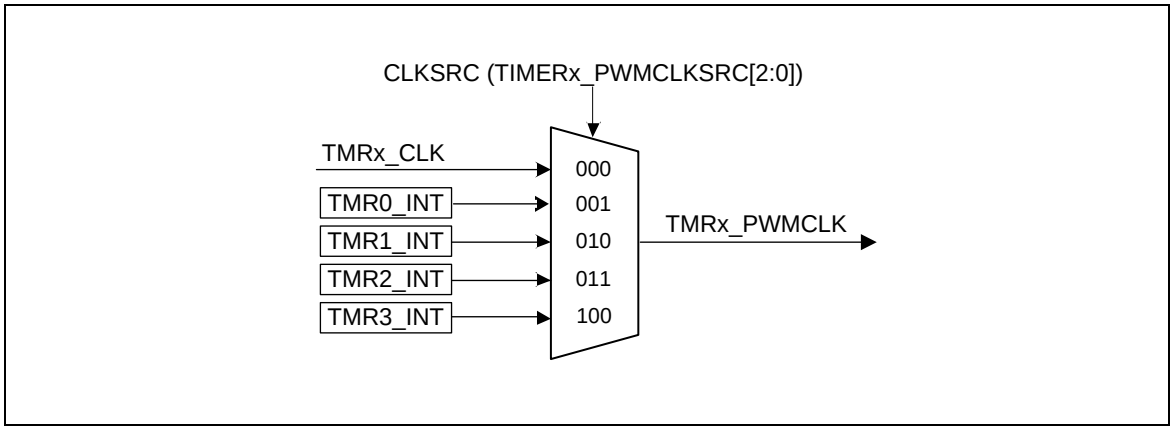


Figure 6.7-5 PWM Counter Clock Source Control

Figure 6.7-6 and Figure 6.7-7 illustrate the architecture of PWM independent mode and complementary mode. Both independent mode and complementary mode support PWMx_CH0 and PWMx_CH1 output channels in each PWM generator.

When PWM counter counts to value 0, or PERIOD (TIMERx_PWMPERIOD[15:0]), or CMP (TIMERx_PWMCMPDAT[15:0]), an event will be generated and trigger the operations following, such as PWM pulse (Pulse Generator), interrupt signal (Interrupt Generator) and triggering signal for ADC to start conversion.

Output Control block manages PWM pulse output as well as interrupt events. Dead-Time Control is available only in PWM complementary mode.

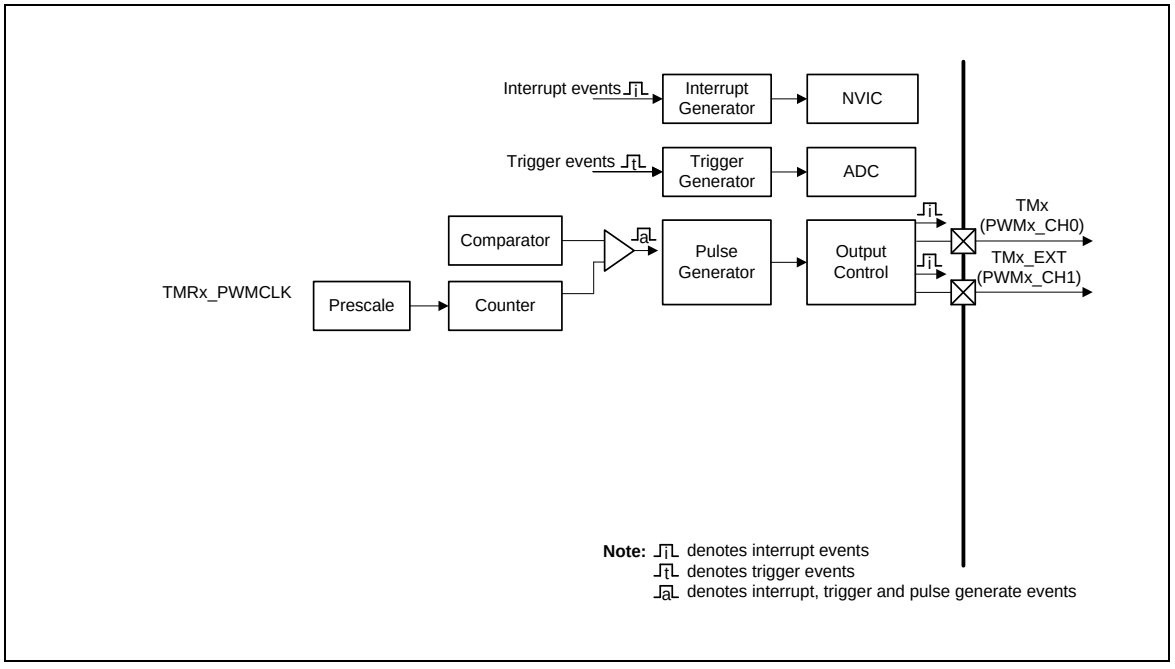


Figure 6.7-6 PWM Independent Mode Architecture Diagram

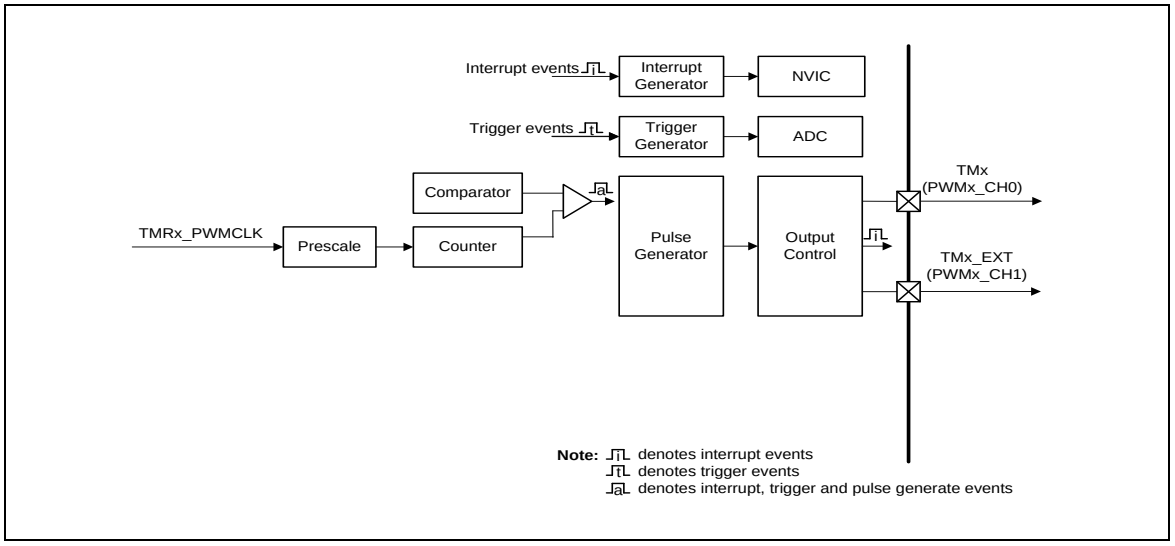


Figure 6.7-7 PWM Complementary Mode Architecture Diagram

6.7.4 Basic Configuration

Typically software needs to configure the registers below to configure a timer:

- FUNCSEL bit (TIMERx_ALTCTL[0]) chooses timer or PWM function for a timer,
- TMRxCKEN is the timer enable bit,
- TMRxSEL bits select the timer clock source,
  x=0, 1, 2 or 3.

For example,

- FUNCSEL (TIMER1_ALTCTL[0]) = 0 chooses timer fucntion for Timer0,
- FUNCSEL (TIMER1_ALTCTL[0]) = 1 chooses PWM function for Timer0,
- TMR0CKEN (CLK_APBCLK0[2]) is the Timer0 enable/disable bit
- and TMR0SEL (CLK_CLKSEL1[10:8]) chooses the Timer0 clock source.

6.7.4.1 TIMER01 basic configurations

- Clock source configuration
  - Enable TIMER0 peripheral clock in TMR0CKEN (CLK_APBCLK0[2]).
  - Enable TIMER1 peripheral clock in TMR1CKEN (CLK_APBCLK0[3]).
- Reset configuration
  - Reset TIMER0 controller in TMR0RST (SYS_IPRST1[2]).
  - Reset TIMER1 controller in TMR1RST (SYS_IPRST1[3]).
- Pin configuration

Group	Pin Name	GPIO	MFP
TM0	TM0	PA.11	MFP2
		PD.8	MFP2

		PD.2	MFP5
	TM0_EXT	PA.12	MFP2
		PD.9	MFP2
TM1	TM1	PA.14	MFP2
		PD.4	MFP5
	TM1_EXT	PA.15	MFP2

6.7.4.2 TIMER23 basic configurations

- Clock source configuration
  - Enable TIMER2 peripheral clock in TMR2CKEN (CLK_APBCLK0[4]).
  - Enable TIMER3 peripheral clock in TMR2CKEN (CLK_APBCLK0[5]).
- Reset configuration
  - Reset TIMER2 controller in TMR2RST (SYS_IPRST1[4]).
  - Reset TIMER3 controller in TMR3RST (SYS_IPRST1[5]).
- Pin configuration

Group	Pin Name	GPIO	MFP
TM2	TM2	PB.2	MFP2
		PD.3	MFP5
	TM2_EXT	PB.3	MFP2
TM3	TM3	PC.2	MFP2
	TM3_EXT	PC.3	MFP2

6.7.5 Timer Functional Description

6.7.5.1 Timer Interrupt Flag

In timer mode, Timer controller can generate the two interrupts:

- Timer Interrupt: TIF (TIMERx_INTSTS[0]) bit will be set when timer counter value CNT (TIMERx_CNT[23:0]) matches the timer compared value CMPDAT (TIMERx_CMP[23:0]);
- Timer External Capture Interrupt: CAPIF (TIMERx_EINTSTS[0]) bit will be set when a transition that matches CAPEDGE (TIMERx_EXTCTL[14:12]) setting is detected on the TMx_EXT pin.

TWKF (TIMERx_INTSTS[1]) bit will be set if a timer mode interrupt wakes up the system. Writing 1 to this bit clears the flag.

6.7.5.2 Timer Counting Mode

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

Timer input clock or event source is divided by (PSC+1) before it is fed to the 24 bit up counter.

By default the 8-bit prescaler PSC (TIMERx_CTL[7:0]) value is 0.

6.7.5.3 One–shot Mode

Writing 0b00 into TIMERx_CTL[28:27] selects one-shot mode for that timer.

CNTEN (TIMERx_CTL[30]) is the timer enable bit. Once enabled, the timer counter starts up counting. Once the CNT (TIMERx_CNT[23:0]) value reaches CMPDAT (TIMERx_CMP[23:0]), TIF (TIMERx_INTSTS[0]) will be set to 1, CNT value and CNTEN bit is automatically cleared by hardware and timer counting operation stops. A timer interrupt will be triggered if INTEN (TIMERx_CTL[29]) is enabled.

Software can monitor bit RSTACT (TIMERx_CNT[31]) to judge if the counter reset operation is still in progress. active by. Software can also set ICEDEBUG (TIMERx_CTL[31]) to 1 that disable ICE debug mode acknowledgement if found affecting TIMER counting.

6.7.5.4 Periodic Mode

Writing 0b01 into TIMERx_CTL[28:27] selects periodic mode for that timer.

CNTEN (TIMERx_CTL[30]) is the timer enable bit. Once enabled, the timer counter starts to count. Once the CNT (TIMERx_CNT[23:0]) value reaches CMPDAT (TIMERx_CMP[23:0]), TIF (TIMERx_INTSTS[0]) will be set to 1, CNT value will be automatically cleared by hardware and timer counter start to count from 0 again. In the meantime, if INTEN (TIMERx_CTL[29]) is enabled a timer interrupt will be generated.

In this mode, timer controller keeps counting and comparing with CMPDAT value continuously, raises TIF periodically until the CNTEN bit is cleared.

6.7.5.5 Toggle-Output Mode

Writing 0b10 into TIMERx_CTL[28:27] selects toggle-output mode.

Under toggle-output mode, timer starts up counting once CNTEN (TIMERx_CTL[30]) is enabled. when the CNT (TIMERx_CNT[23:0]) value reaches CMPDAT (TIMERx_CMP[23:0]), device toggles the output signal on associated TM0/1/2/3 pin; for example, Timer0 associated toggle-output pin is TM0. Also device raises TIF flag.

In this mode, the timer continuously counts, periodically toggles the output on the TMx pin and raises TIF (hence interrupt will be generated if interrupt enabled), until CNTEN bit is cleared. The duty cycle on TMx pin is 50%.

6.7.5.6 Continuous Counting Mode

Writing 0b11 into TIMERx_CTL[28:27] selects continuous counting mode.

Under continuous counting mode, timer starts up counting once CNTEN (TIMERx_CTL[30]) is enabled.

When CNT (TIMERx_CNT[23:0]) value reaches CMPDAT (TIMERx_CMP[23:0]) value, the TIF (TIMERx_INTSTS[0]) will be set to 1, hence a timer interrupt generated if enabled, and counting continues. Software can change CMPDAT to a different value immediately without stopping/resetting timer.

For example, if CMPDAT value is initially set as 80, when CNT reaches 80, timer raises TIF and generates interrupt (if interrupt enabled), and counting continues. Now assume software clears TIF and re-set CMPDAT value to 200, then when CNT reaches 200 timer will raise TIF and generate interrupt again.

In this mode, timer counter counts endlessly from 0 to 2^{24} -1, then from 0 again, until CNTEN is disabled.

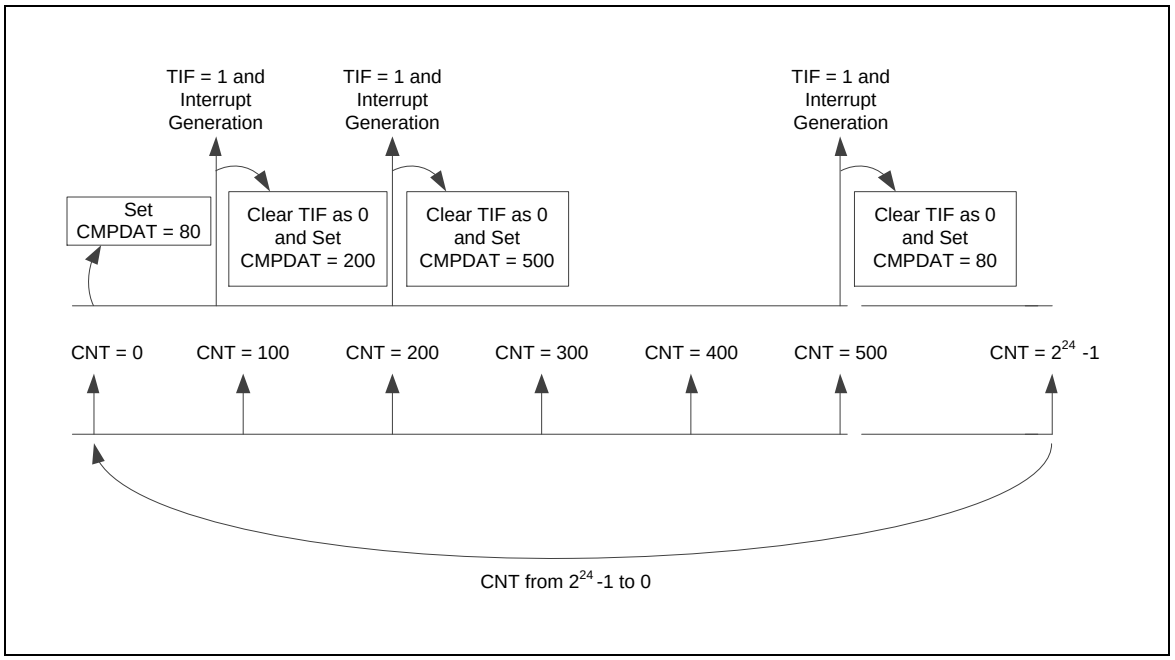


Figure 6.7-8 Continuous Counting Mode

6.7.5.7 Event Counting Mode

In Event Counting mode, a timer counts the external input event from its TMx pin. The timer clock source should be from PCLK.

- EXTCNTEN (TIMERx_CTL[24]): Event counting mode enable bit.
- CNT (TIMERx_CNT[23:0]): number of events occurred on the TMx pin.
- ECNTSSEL (TIMERx_EXTCTL[16]): equals 0 to choose event source from TMx pin. '1' is a reserved value for this bit.
- CNTDBEN (TIMERx_EXTCTL[7]): configures the TMx pin de-bounce value.
- CNTPHASE (TIMERx_EXTCTL[0]): determine to count falling edge or rising edge

In this mode, the clock source should choose PCK0 for Time0/1, and PCK1 for Timer2/3. Timer counting mode can be selected as one-shot, periodic or continuous counting mode.

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 TMx_EXT (x= 0~3) pin. In this mode, CAPFUNCS (TIMERx_EXTCTL[4]) should be as 0 to trigger event capture function and the timer peripheral clock source should be set as PCLK.

User can enable or disable TMx_EXT pin de-bounce circuit by setting CAPDBEN (TIMERx_EXTCTL[6]). The transition frequency of TMx_EXT pin should be less than 1/3 PCLK if TMx_EXT pin de-bounce disabled or less than 1/8 PCLK if TMx_EXT pin de-bounce enabled to assure the capture function can be work normally, and user can also select edge transition detection of TMx_EXT pin by setting CAPEDGE (TIMERx_EXTCTL[14:12]).

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

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

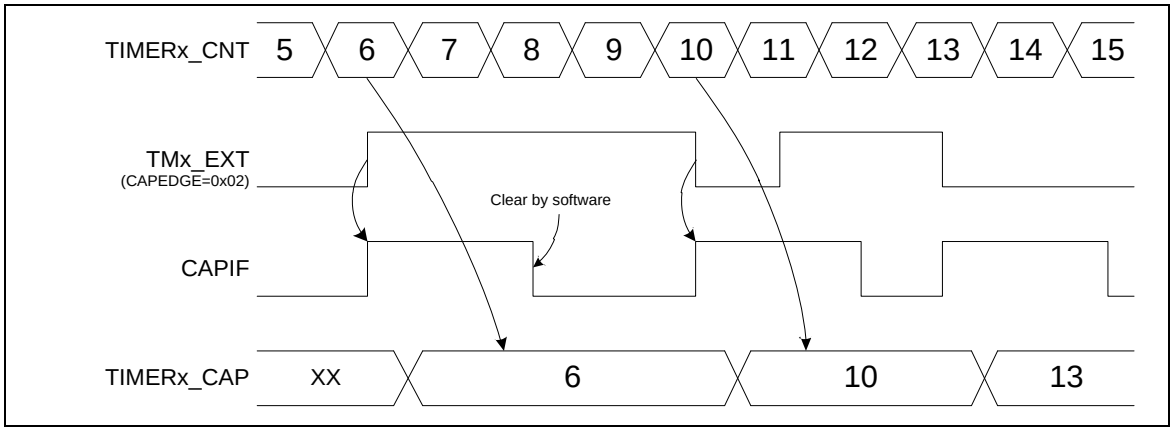


Figure 6.7-9 External Capture Mode

6.7.5.9 External Reset Counter Mode

Timer controller also provides reset counter function to reset CNT (TIMERx_CNT[23:0]) value while capture event is generated. In this mode, CAPFUNCS (TIMERx_EXTCTL[4]) should be as 1 for select TMx_EXT transition to trigger reset counter value.

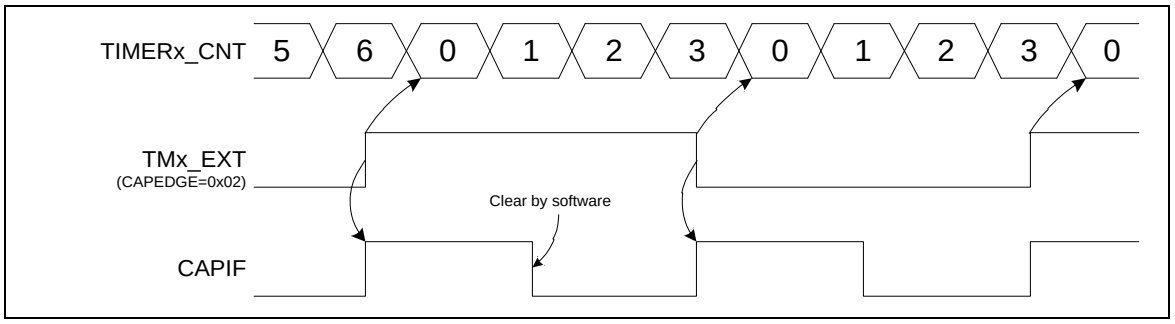


Figure 6.7-10 External Reset Counter Mode

6.7.5.10 Timer Trigger Function

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

When the TRGPWM (TIMERx_TRGCTL[1]) is set, if the timer interrupt signal is generated, the timer controller will generate a trigger pulse as PWM external clock source.

When the TRGADC (TIMERx_TRGCTL[2]) is set, if the timer interrupt signal is generated, the timer controller will trigger ADC to start converter.

When the TRGPDMA (TIMERx_TRGCTL[4]) is set, if the timer interrupt signal is generated, the timer controller will trigger PDMA.

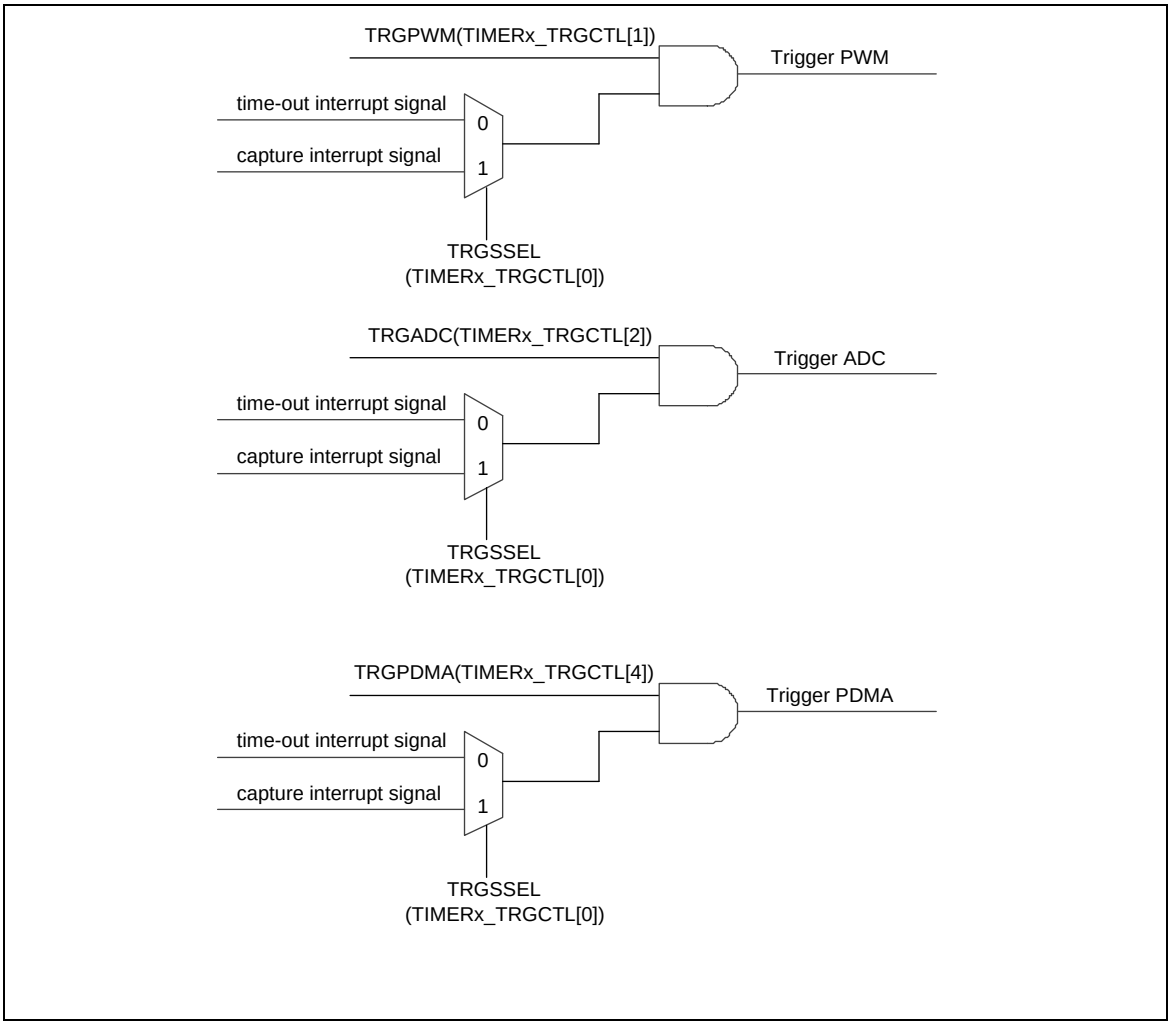


Figure 6.7-11 Internal Timer Trigger

6.7.5.11 Inter-Timer Trigger Capture Mode

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

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

- Start Trigger

While INTRGEN (TIMERx_CTL[19]) in Timer0/2 is set, the Timer0/2 will make a rising-edge transition of INTR_TMR_TRG while Timer0/2 24-bit counter value (CNT) is counting from 0x0 to 0x1 and Timer1/3 counter will start counting immediately and automatically.

- Stop Trigger

When Timer0/2 CNT reaches the Timer0/2 CMPDAT value, the Timer0/2 will make a falling-edge transition of INTR_TMR_TRG. Then Timer0/2 counter mode function will be disabled and INTRGEN (TIMERx_CTL[19]) will be cleared by hardware then Timer1/3 will stop counting also. At the same time, the Timer1/3 CNT value will be saved into Timer1/3 CAPDAT

(TIMERx_CAP[23:0]).

User can use inter-timer trigger mode to measure the period of external event (TMx) more precisely. Figure 6.7-12 shows the sample flow of Inter-Timer Trigger Capture Mode for Timer0 as event counting mode and Timer1 as trigger-counting capture mode.

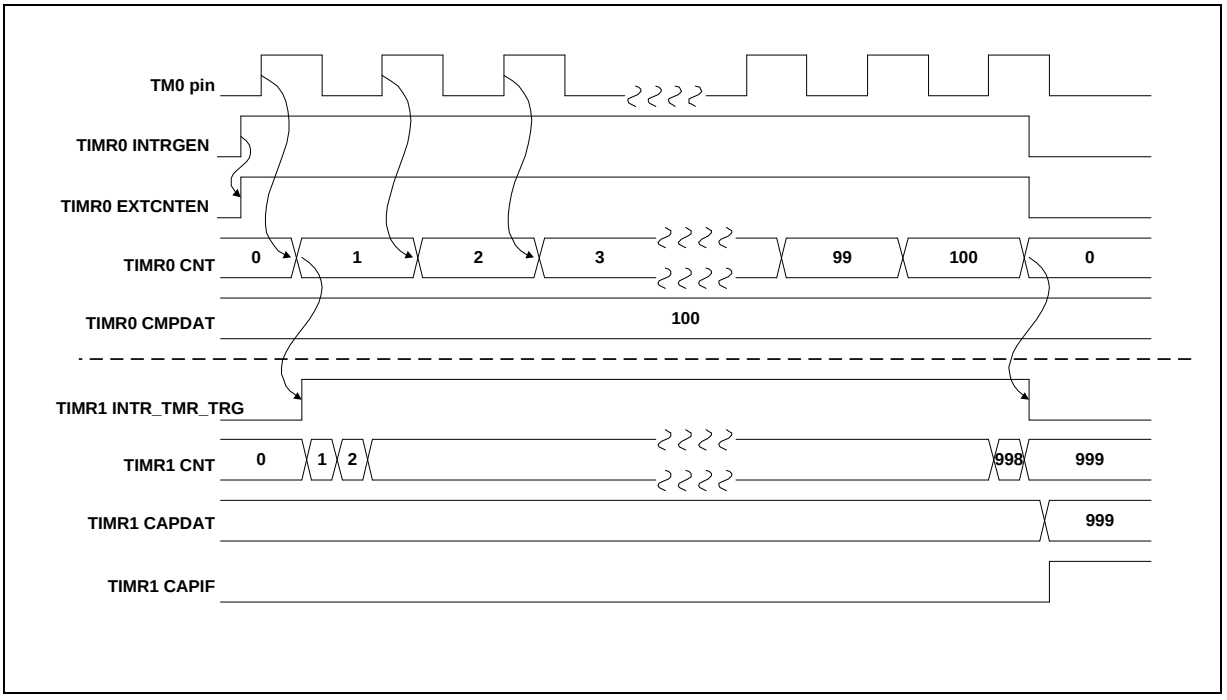


Figure 6.7-12 Inter-Timer Trigger Capture Timing

6.7.6 PWM Functional Description

6.7.6.1 PWM Prescale

The PWM prescale is used to divide clock source, and the clock of PWM counter is divided by (CLKPSC+ 1). The prescale is set by CLKPSC (TIMERx_PWMCLKPSC[11:0]). Figure 6.7-13 shows an example of PWM prescale waveform in up count type.

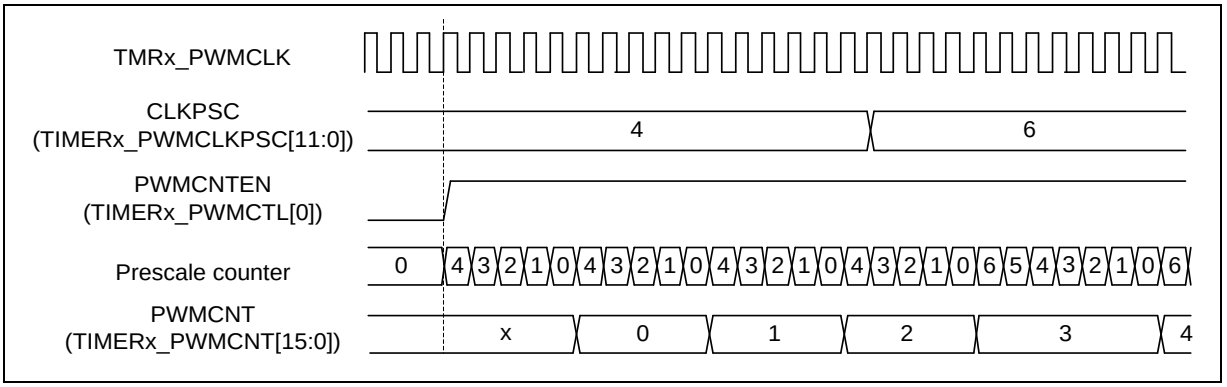


Figure 6.7-13 PWM Prescale Waveform in Up Count Type

6.7.6.2 PWM Counter

PWM supports three counter types operation: up count, down count and up-down count types.

6.7.6.3 Up Count Type

When PWM counter is set to up count type, CNTTYPE (TIMERx_PWMCTL[2:1]) is 0x0, it starts up-counting from zero to PERIOD (TIMERx_PWMPERIOD[15:0]). The current counter value can be read from the CNT (TIMERx_PWMCNT[15:0]). PWM generates a zero point event when both counter and prescale counts to 0. PWM generates a period point event when the counter counts to PERIOD and prescale counts to 0. Figure 6.7-14 shows an example of PWM up count type, where PWM period time is (PERIOD+1) * (CLKPSC+1) * TMRx_PWMCLK.

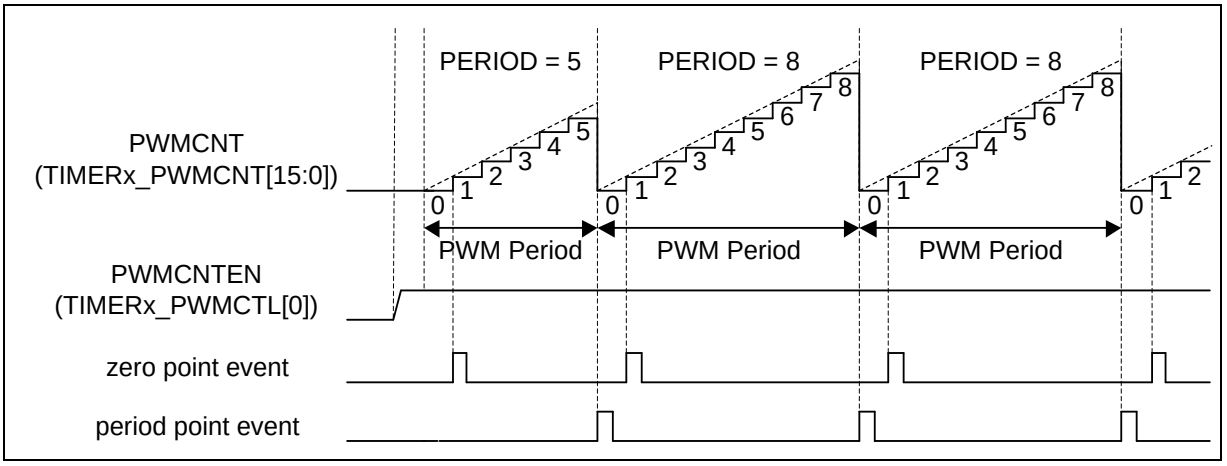


Figure 6.7-14 PWM Up Count Type

6.7.6.4 Down Count Type

When PWM counter is set to down count type, CNTTYPE (TIMERx_PWMCTL[2:1]) is 0x1, it starts down-counting from PERIOD to zero, current counter value can be read from CNT (TIMERx_PWMCNT[15:0]). PWM generates a zero point event when both counter and prescale counts to 0. PWM generates a period point event when the counter counts to PERIOD and prescale counts to 0. Figure 6.7-15 is an example of PWM down count type, where PWM period time is (PERIOD+1) * (CLKPSC+1) * TMRx_PWMCLK.

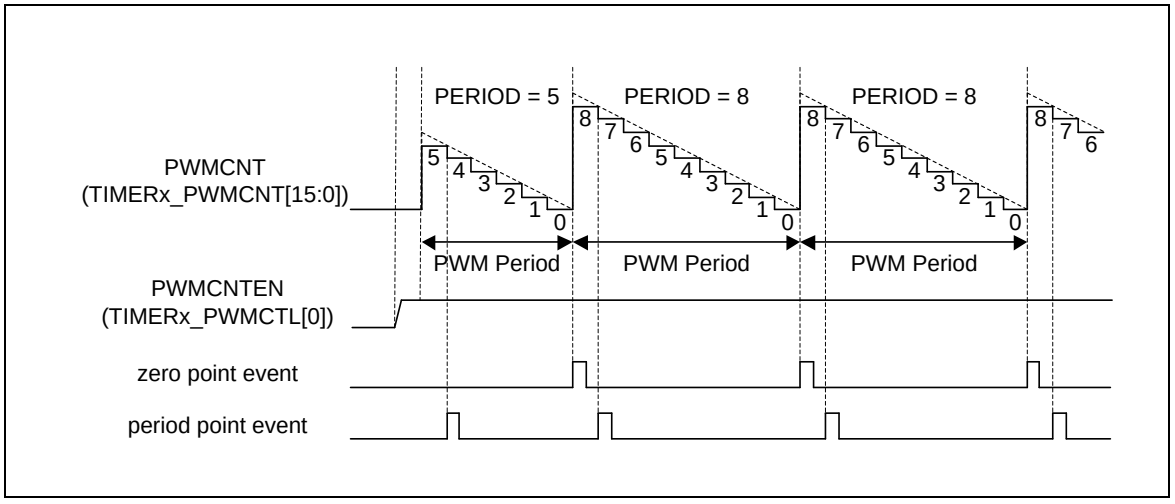
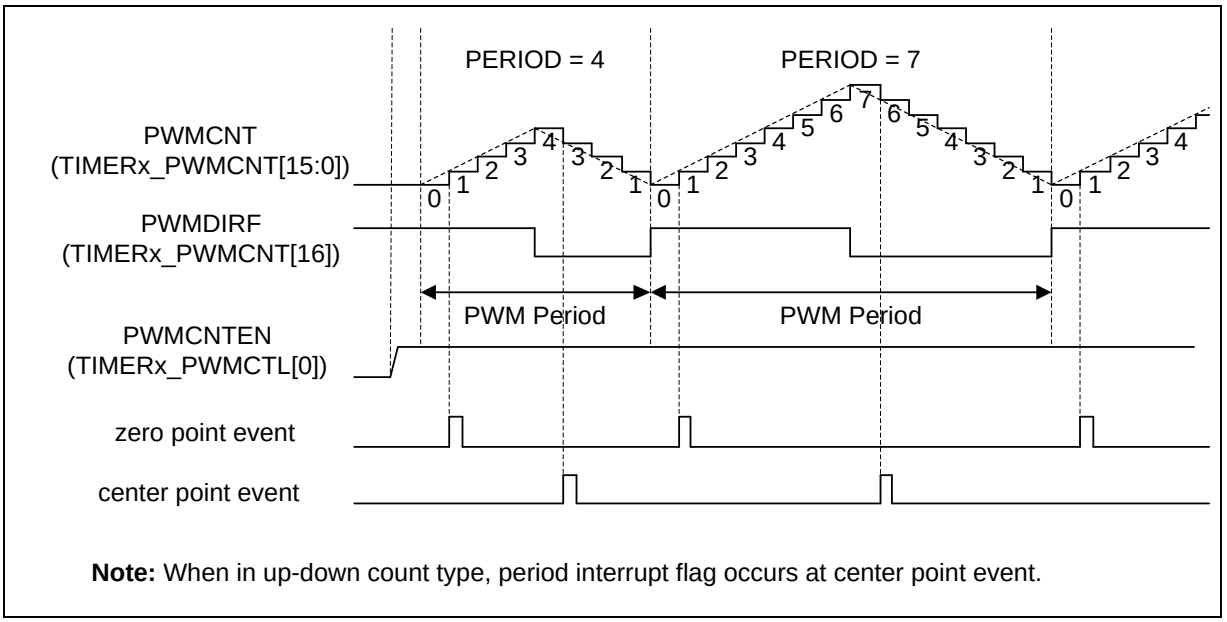


Figure 6.7-15 PWM Down Count Type

6.7.6.5 Up-Down Count Type

When PWM counter is set to up-down count type, CNTTYPE (TIMERx_PWMCTL[2:1]) is 0x2, it starts counting up from zero to PERIOD and then starts counting down to zero. The current counter value can be read from CNT (TIMERx_PWMCNT[15:0]). PWM generates a zero point event when both counter and prescale counts to 0. PWM generates a center point event when the counter counts to PERIOD and prescale counts to 0. Figure 6.7-16 shows an example of PWM up-down count type, where PWM period time is (2 * PERIOD) * (CLKPSC+1) * TMRx_PWMCLK. The DIRF (TIMERx_PWMCNT[16]) is counter direction indicator flag, where 1 is up counting, and 0 is down counting.



Note: When in up-down count type, period interrupt flag occurs at center point event.

Figure 6.7-16 PWM Up-Down Count Type

6.7.6.6 PWM Counter Operation mode

The PWM counter supports two operation modes: one-shot mode and auto-reload mode. PWM counter will operate in one-shot mode if CNTMODE (TIMERx_PWMCTL[3]) bit is set to 1, and operate in auto-reload mode if CNTMODE bit is set to 0.

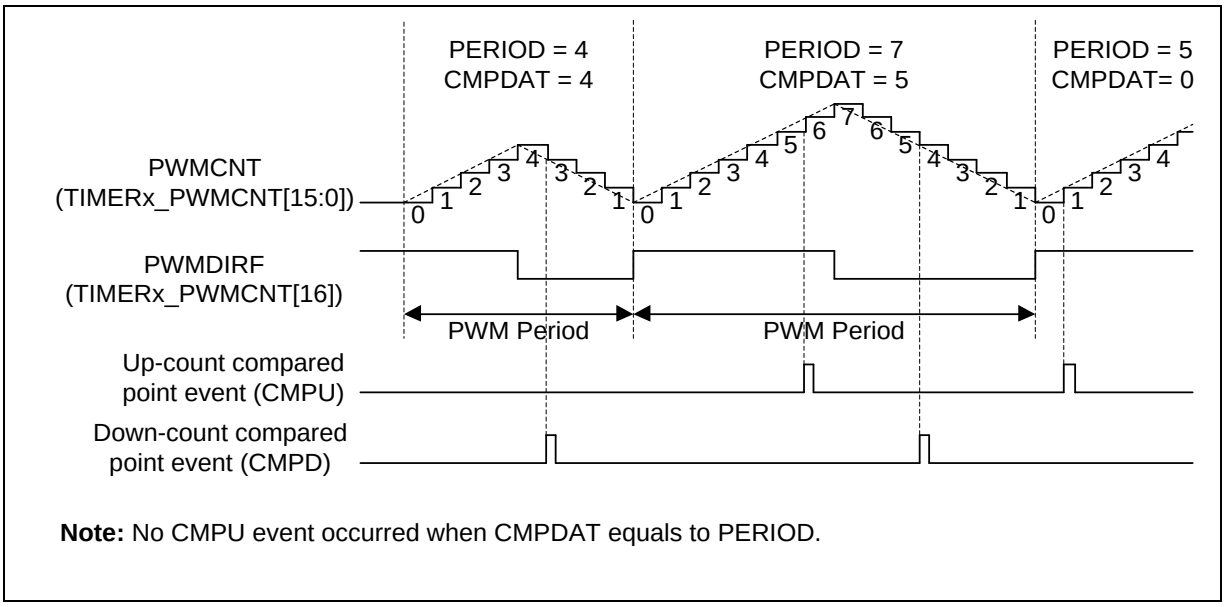
In both modes, CMP (TIMERx_PWMCMPDAT[15:0]) and PERIOD (TIMERx_PWMPERIOD[15:0]) should be written first and then set CNTEN (TIMERx_PWMCTL[0]) bit to 1 to start counter running.

In one-shot mode, PWM counter value will reload to default value according count type after one PWM period is completed. User can write CMP to continuous one-shot operation to generate next one-shot pulse once no matter current one-shot counter is running or completed.

In auto-reload mode, PWM counter is continuous running with current active PERIOD and CMP. If user set PERIOD to zero in auto-reload mode, PWM counter value will reload to default value according count type after one PWM period is completed.

6.7.6.7 PWM Comparator

The CMP (TIMERx_PWMCMPDAT[15:0]) is comparator register of PWM. The CMP value is continuously compared to the corresponding counter value. When the counter is equal to CMP, PWM generates a compared point event. This event will generate PWM output pulse, interrupt signal or trigger ADC start convert. In up-down count type, two events will be generated in a PWM period as shown in Figure 6.7-17. The CMPU is up count compared point event and CMPD is down count compared point event.



Note: No CMPU event occurred when CMPDAT equals to PERIOD.

Figure 6.7-17 PWM Comparator Events in Up-Down Count Type

6.7.6.8 Period Loading Mode

When the IMMLDEN (TIMERx_PWMCTL[9]) bit set to 0, PWM operates at period loading mode. The PWM provides PBUF (TIMERx_PWMPBUF[15:0]) is the active PERIOD buffer register and CMPBUF (TIMERx_PWMCMPBUF[15:0]) is the active CMP buffer register. In period loading mode, both PERIOD (TIMERx_PWMPERIOD[15:0]) and CMP (TIMERx_PWMCMPDAT[15:0]) will load to their active PBUF and CMPBUF register while each PWM period is completed. Figure

6.7-18 shows period loading timing of up count type, where PERIOD DATA0 denotes the initial data of PERIOD, PERIOD DATA1 denotes the first updated PERIOD data by user and so on, CMP also follows this rule. The following steps are the sequence of Figure 6.7-18.

1. User writes CMP DATA1 to CMP at point 1.
2. Period loading CMP DATA1 to CMPBUF at the end of PWM period at point 2.
3. User 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. User writes PERIOD DATA2 to PERIOD at point 5.
6. Period loading PERIOD DATA2 to PBUF at the end of PWM period at point 6.

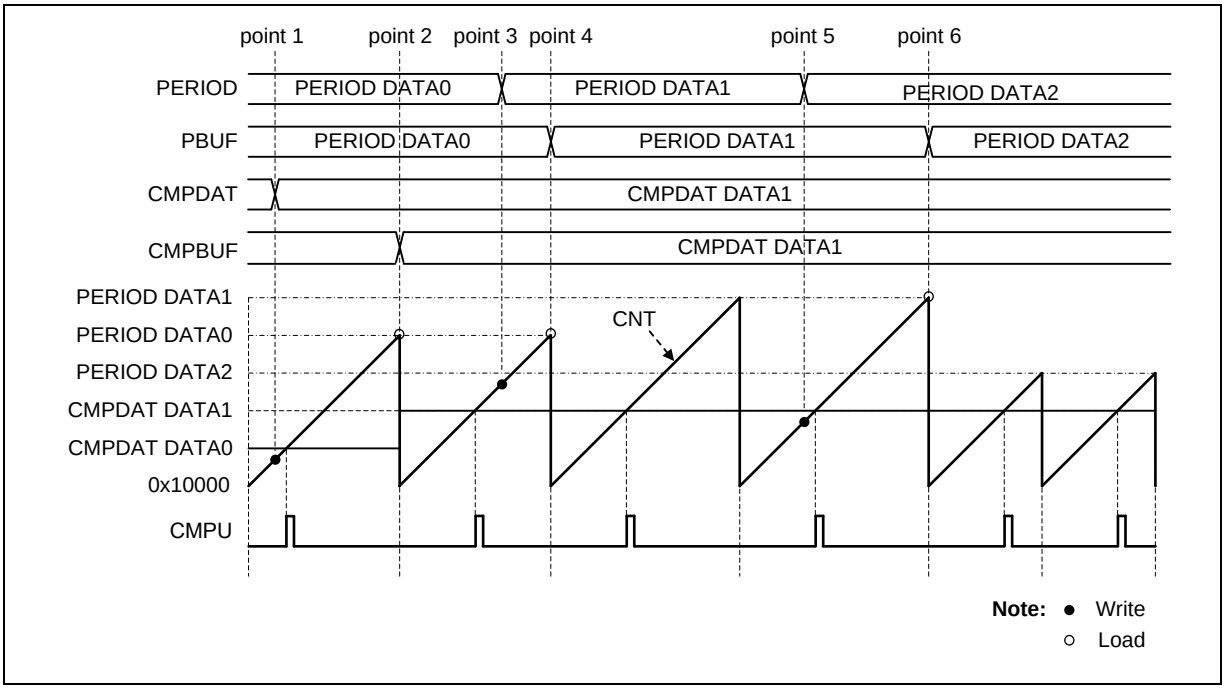


Figure 6.7-18 Period Loading Mode with Up Count Type

6.7.6.9 Immediately Loading Mode

When the IMMLDEN (TIMERx_PWMCTL[9]) bit set to 1, PWM operates at immediately loading mode. In immediately loading mode, when user update PERIOD (TIMERx_PWMPERIOD[15:0]) or CMP (TIMERx_PWMCMPDAT[15:0]), PERIOD or CMP will be load to active PBUF (TIMERx_PWMPBUF[15:0]) or CMPBUF (TIMERx_PWMCMPBUF[15:0]) after current counter count is completed. If the update PERIOD value is less than current counter value, counter will count wraparound. The following steps are the sequence of Figure 6.7-19.

1. User writes CMP DATA1 at point 1 and hardware will load CMP DATA1 to CMPBUF after current counter count is completed.
2. User writes PERIOD DATA1 at point 2 and PERIOD DATA1 is greater than current counter value, PWM counter will continuously count until equal to PERIOD DATA1 to complete one PWM period.
3. User writes PERIOD DATA2 at point 3 and PERIOD DATA2 is less than the current counter value, PWM counter will continuously count to maximum counter value 0x1FFFF and wraparound from 0x10000 to PERIOD DATA2 to complete one PWM period.

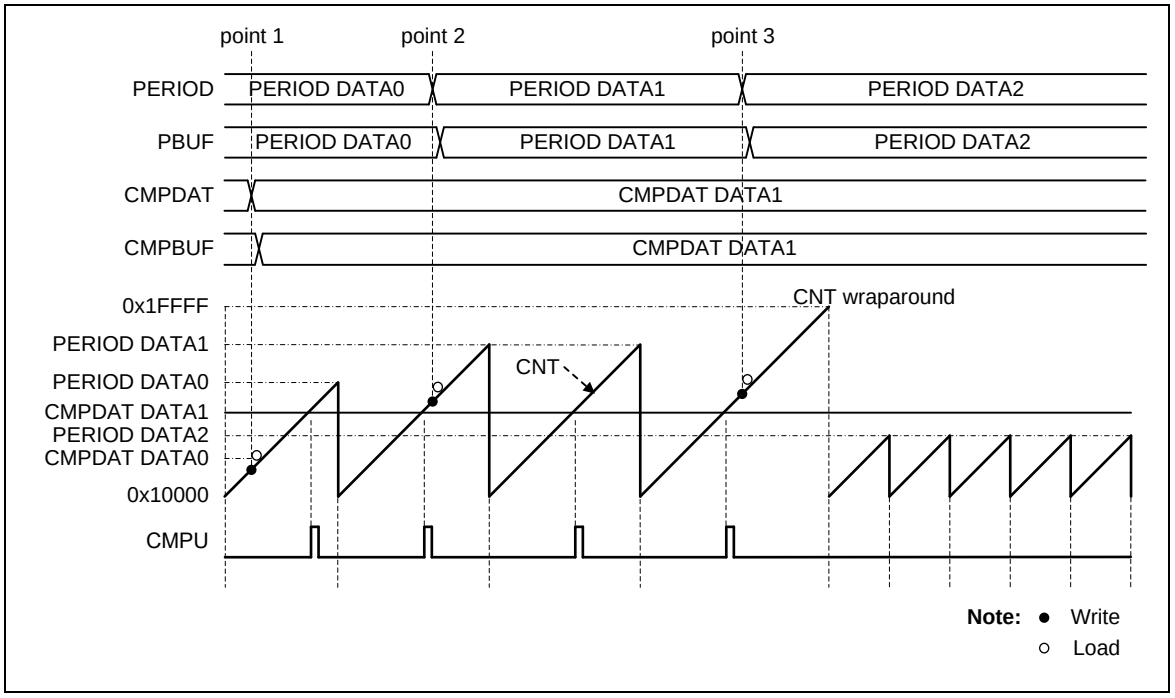


Figure 6.7-19 Immediately Loading Mode with Up Count Type

6.7.6.10 PWM Pulse Generator

PWM pulse generator uses counter and comparator events to generate PWM output pulse. The events are zero point and period point in up count type and down count type, center point in up-down count type and counter equal to comparator point in three count types.

Each event point can generate PWM output waveform in different count type as shown in Figure 6.7-20.

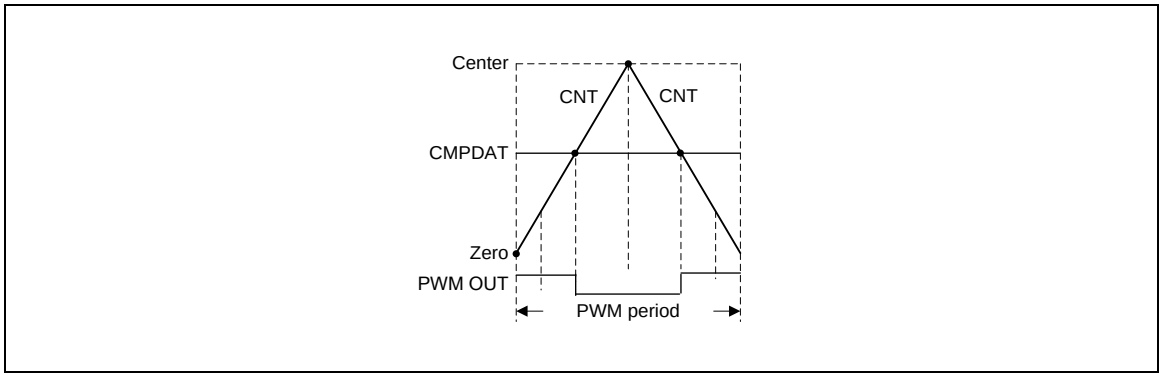


Figure 6.7-20 PWM Pulse Generation in Up-Down Count Type

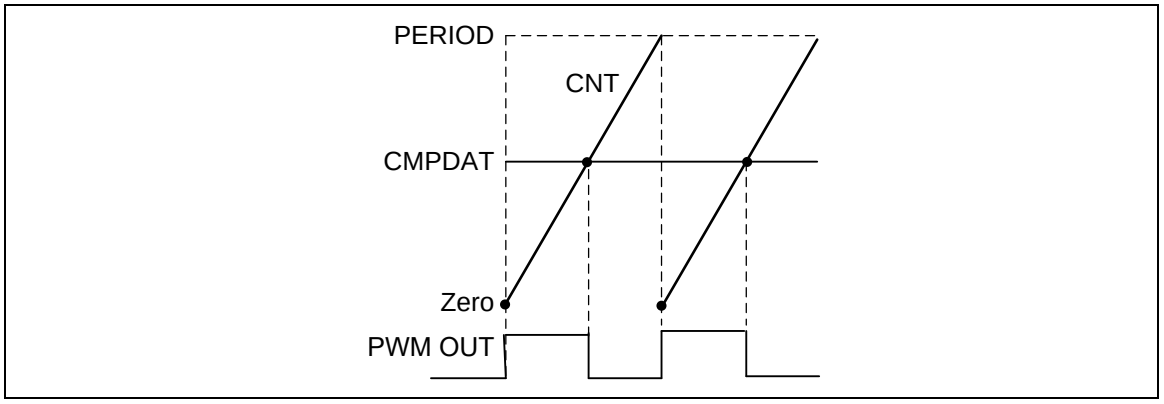


Figure 6.7-21 PWM Pulse Generation in Up Count Type

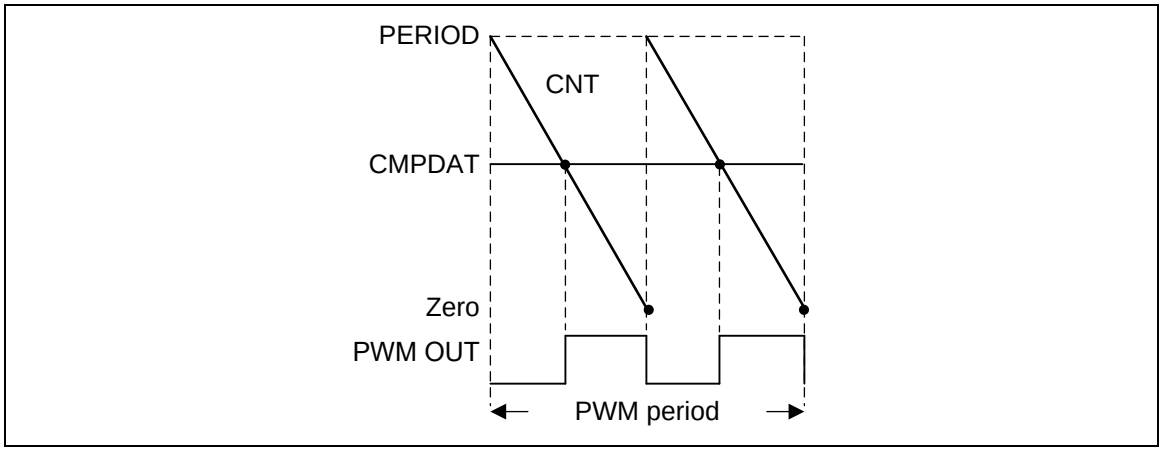


Figure 6.7-22 PWM Pulse Generation in Down Count Type

The PWM generation events may sometimes generated at the same time, as the reason, events priority between different counter types should be take care are list in Table 6.7.6-1 , Table 6.7.6-2 and Table 6.7.6-3, event priority in up count type, event priority in down count type and event priority in up-down count type.

Priority	Zero And CMPU Point Event (CMP = 0)	PWM Output
1 (High)	Compare up event	Low
2 (Low)	Zero event	High

Table 6.7.6-1 PWM Pulse Generation Event Priority in Up Count Type

Priority	Zero And CMPD Point Event (CMP = 0)	PWM Output
1 (High)	Zero event	Low
2 (Low)	Compare down event	High

Priority	Period and CMPD point event (CMP = PERIOD)	PWM output
1 (High)	Compare down event	High
2 (Low)	Period event	Low

Table 6.7.6-2 PWM Pulse Generation Event Priority in Down Count Type

Priority	CMPU And CMPU Point Event (CMP = PERIOD)	PWM Output
1 (High)	Compare down event	High
2 (Low)	Compare up event	Low

Table 6.7.6-3 PWM Pulse Generation Event Priority in Up-Down Count Type

According to event priority limitation, PWM generator can support 0% and 100% duty cycle PWM output waveform only in up count and up-down count type. Figure 6.7-23 is an example about PWM duty cycle from 0% to 100% in up count type and up-down count type where PERIOD is 4 with different CMP value.

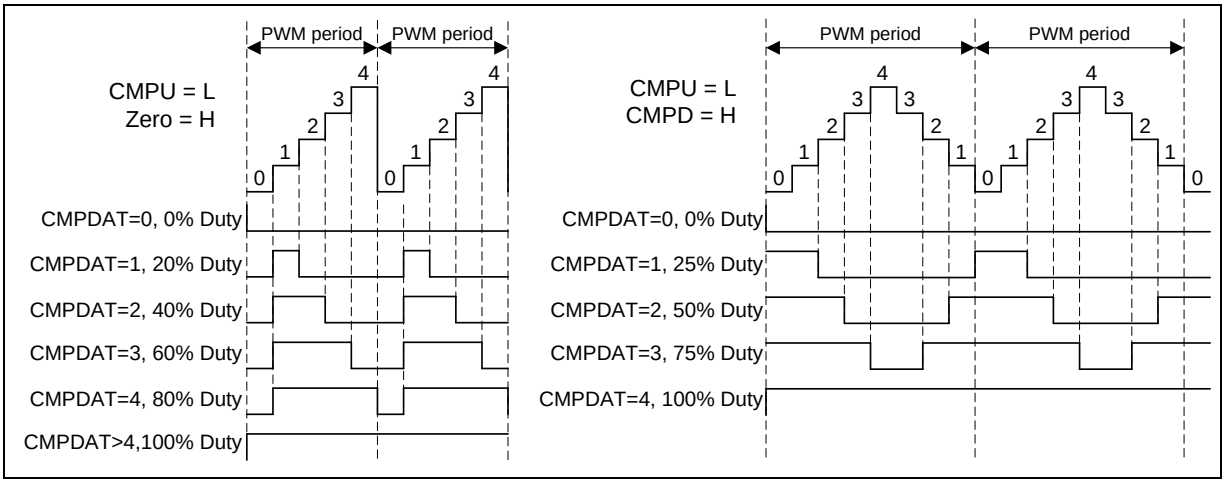


Figure 6.7-23 PWM 0% to 100% Duty Cycle in Up Count Type and Up-Down Count Type

6.7.6.11 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.

6.7.6.12 Independent mode

When OUTMODE (TIMERx_PWMCTL[16]) bit is set to 0, PWM output operates in independent mode. In this mode, both PWMx_CH0 and PWMx_CH1 can output the same waveform as shown in Figure 6.7-24.

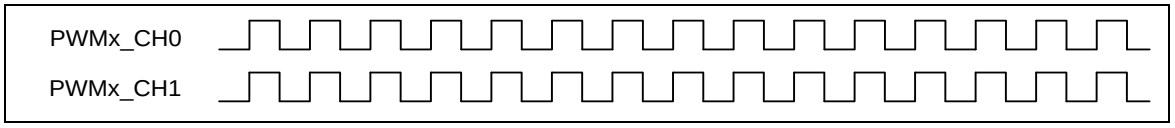


Figure 6.7-24 PWM Independent Mode Output Waveform

6.7.6.13 Complementary mode

When OUTMODE (TIMERx_PWMCTL[16]) bit is set to 1, PWM output operates in complementary mode. In this mode, both PWMx_CH0 and PWMx_CH1 can output waveform and PWMx_CH1 must always be the complement of PWMx_CH0 as shown in Figure 6.7-25.

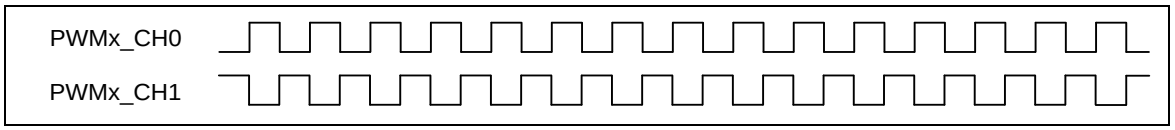


Figure 6.7-25 PWM Complementary Mode Output Waveform

6.7.6.14 PWM Output Control

After PWM pulse generator, there are four steps to control output waveform in independent output mode and five control steps in complementary output mode. User can set POEN0 (TIMERx_PWMPOEN[0]) and POEN1 (TIMERx_PWMPOEN[1]) 1 to enable PWMx_CH0 and PWMx_CH1 output waveform.

In Independent mode, there are mask control, polarity control and output enable control to control output waveform as shown in Figure 6.7-26.

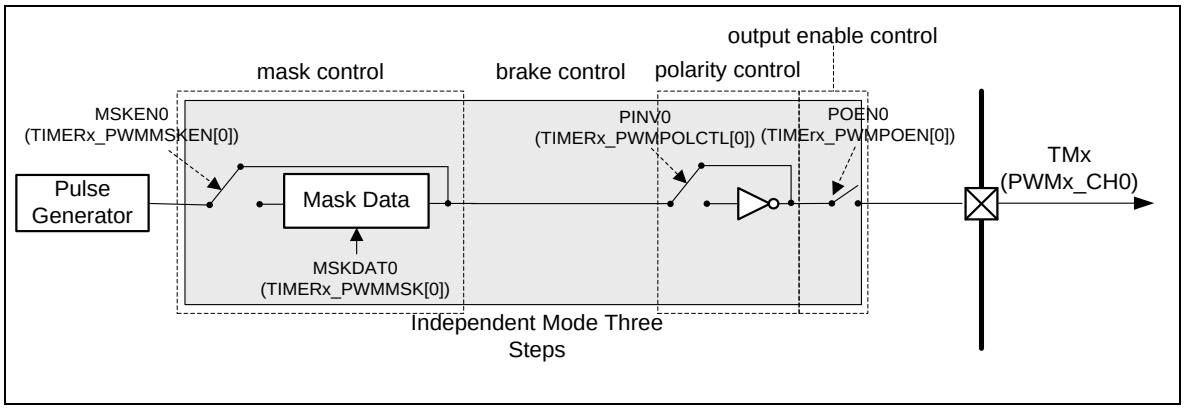


Figure 6.7-26 PWMx_CH0 Output Control in Independent Mode

In complementary mode, there are dead-time insertion control and three control steps the same as independent mode to control PWMx_CH0 and PWMx_CH1 outputs as shown in Figure 6.7-27.

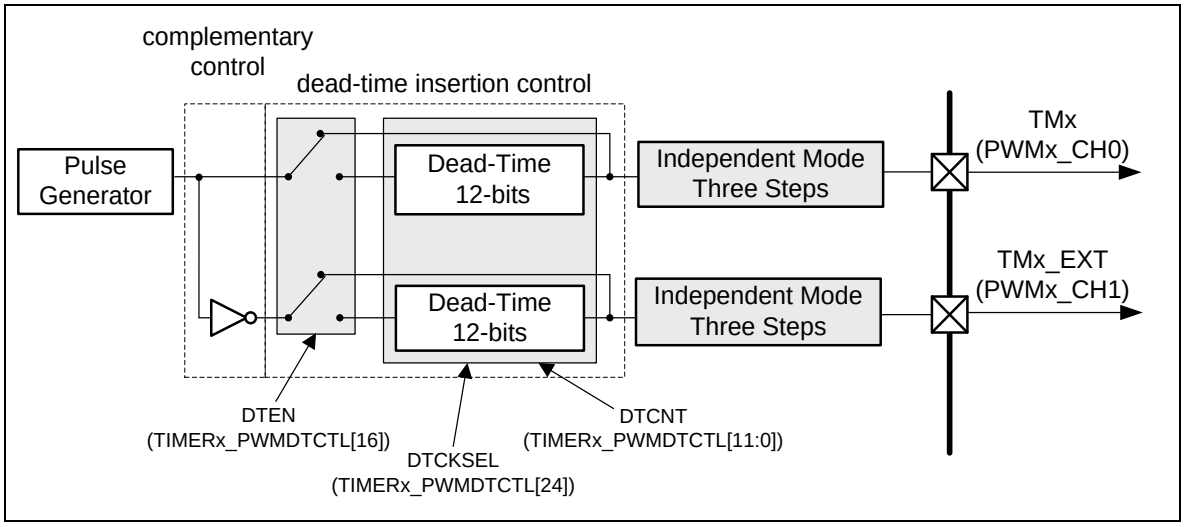


Figure 6.7-27 PWMx_CH0 and PWMx_CH1 Output Control in Complementary Mode

6.7.6.15 Dead-Time Insertion Control

In the complementary application, the complement channels may drive the external devices like power switches. The dead-time generator inserts a low level interval between complementary outputs PWMx_CH0 and PWMx_CH1 as shown in Figure 6.7-28. User sets DTEN (TIMERx_PWMDTCTL[16]) bit to enable dead-time control function, DTCNT (TIMERx_PWMDTCTL[11:0]) and DTCKSEL (TIMERx_PWMDTCTL[24]) to control dead-time interval. The dead-time interval can be calculated from the following formula:

Dead-time interval = (DTCNT + 1) * TMRx_PWMCLK period, if DTCKSEL is 0

Dead-time interval = (DTCNT + 1) * TMRx_PWMCLK * (CLKPSC + 1) period, if DTCKSEL is 1

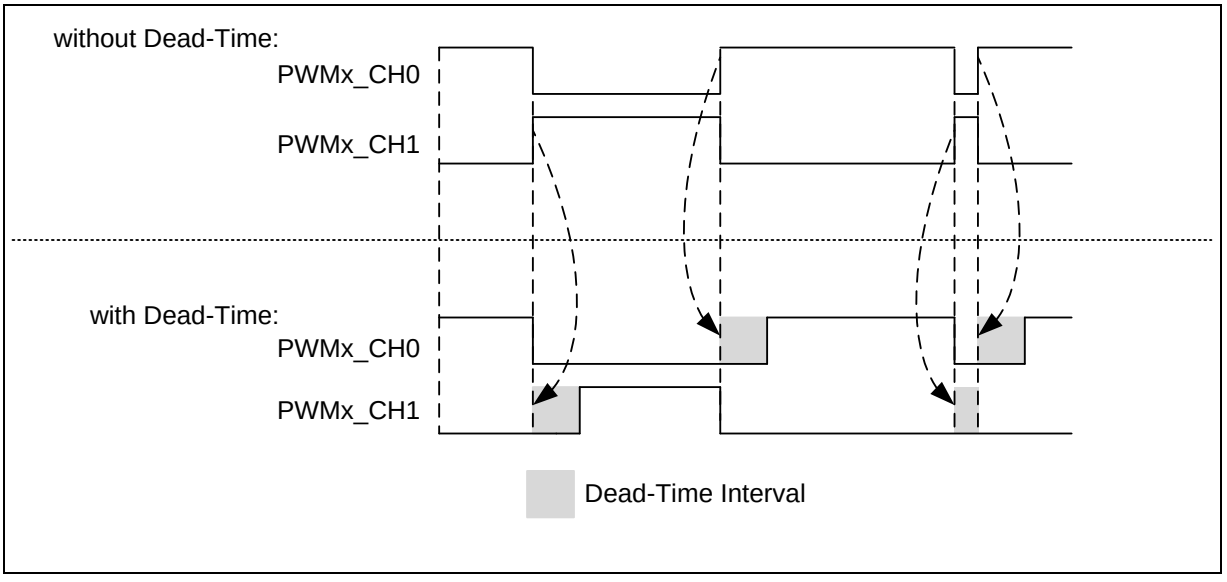


Figure 6.7-28 Dead-Time Insertion

6.7.6.16 PWM Mask Output Control

PWMx_CH0/CH1 output value can be masked to specified logic states by setting MSKEN0/1 (TIMERx_PWMMSKEN[1:0]) and MSKDAT0/1 (TIMERx_PWMMSK[1:0]). The PWM output mask function is useful when controlling various types of Electrically Commutated Motor (ECM) like a BLDC motor. Figure 6.7-29 shows an example of PWM output mask control in PWMx_CH0 and PWMx_CH1.

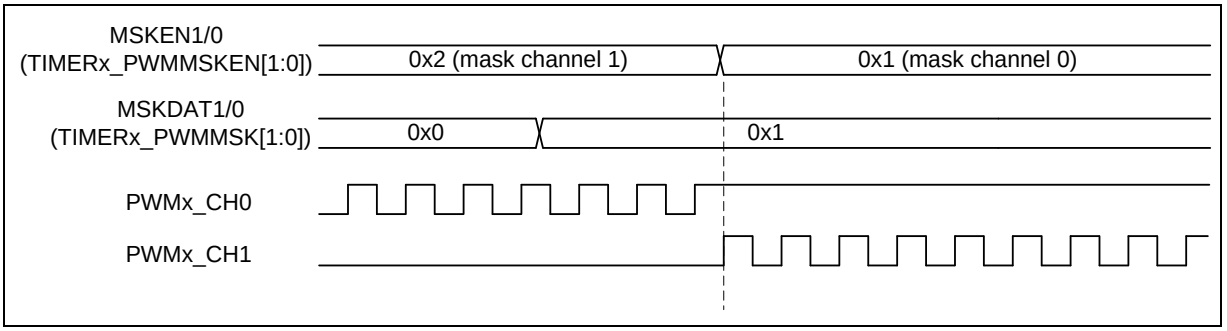


Figure 6.7-29 PWM Output Mask Control Waveform

6.7.6.17 Polarity Control

Each PWMx_CH0 and PWMx_CH1 has an independent polarity control to configure the polarity of the active state of PWM output. User can control polarity state of PWMx_CH0 on PINV0 (TIMERx_PWMPOLCTL[0]) and PWMx_CH1 on PINV1 (TIMERx_PWMPOLCTL[1]). Figure 6.7-30 shows the PWMx_CH0 and PWMx_CH1 output with polarity control and dead-time insertion.

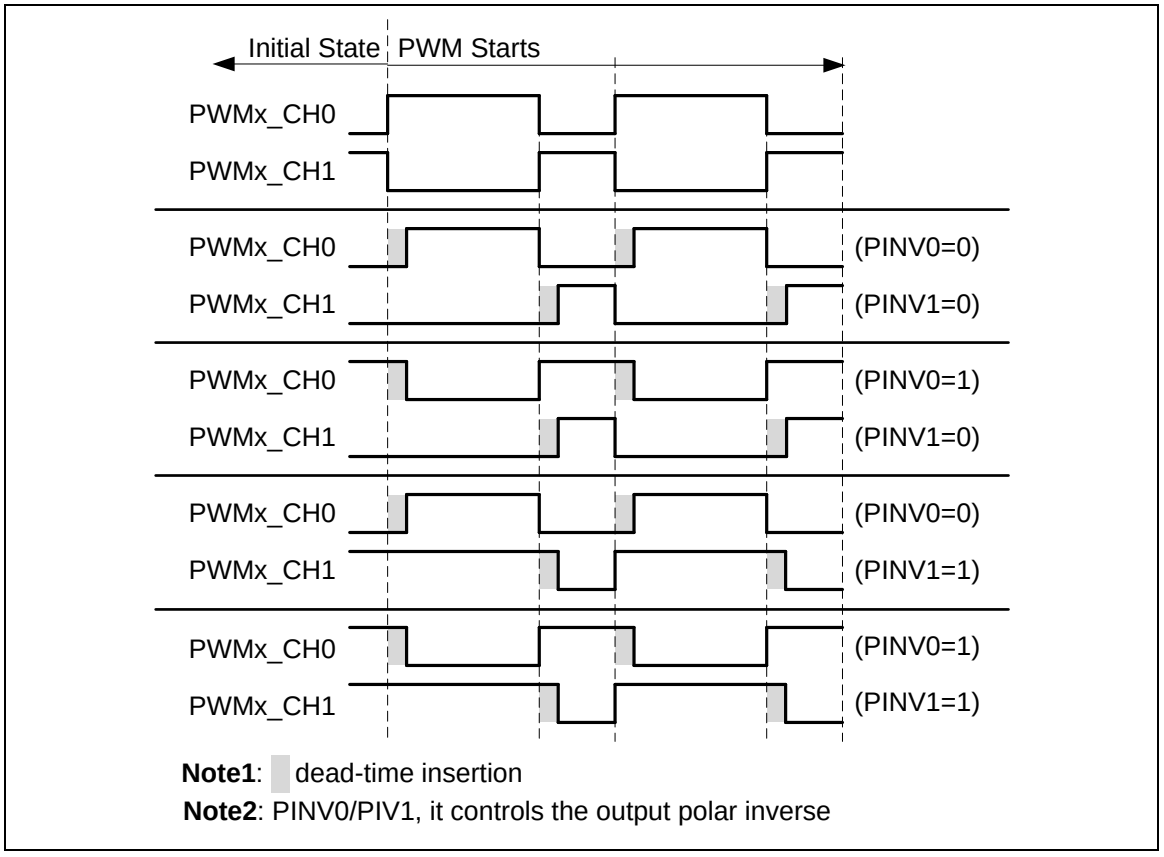


Figure 6.7-30 PWMx_CH0 and PWMx_CH1 Polarity Control with Dead-Time Insertion

6.7.6.18 PWM Interrupt Generator

There are independent interrupts for each PWM as shown in Figure 6.7-31.

The PWM interrupt (PWMx_INT) comes from PWM complementary pair events. The counter can generate the zero point interrupt flag ZIF (TIMERx_PWMINTSTS0[0]) and the period point interrupt flag PIF (TIMERx_PWMINTSTS0[1]). When counter equals to the comparator value stored in CMP (TIMERx_PWMCMPDAT[15:0]), the different interrupt flags will be triggered depending on the counting direction. If counter and CMP matched occurs at up-count direction, the comparator up interrupt flag CMPUIF (TIMERx_PWMINTSTS0[2]) is set and if matched at down-count direction, the comparator down interrupt flag CMPDIF (TIMERx_PWMINTSTS0[3]) is set. If the corresponding interrupt enable bits are set, the interrupt trigger events will also generates interrupt signals.

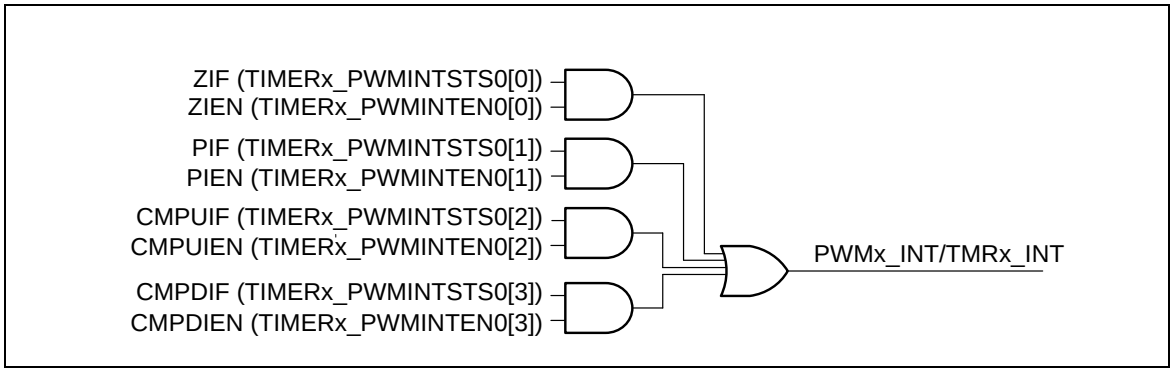


Figure 6.7-31 PWM Interrupt Architecture Diagram

6.7.6.19 PWM Trigger ADC Generator

PWM counter event can be one of the ADC conversion trigger source. User sets TRGSEL (TIMERx_PWMADCTS[3:0]) to select which PWM counter event can trigger ADC conversion after TRGEN (TIMERx_PWMADCTS [7]) is enabled.

There are five PWM counter events can be selected as the trigger source to start ADC conversion which shown in Figure 6.7-32.

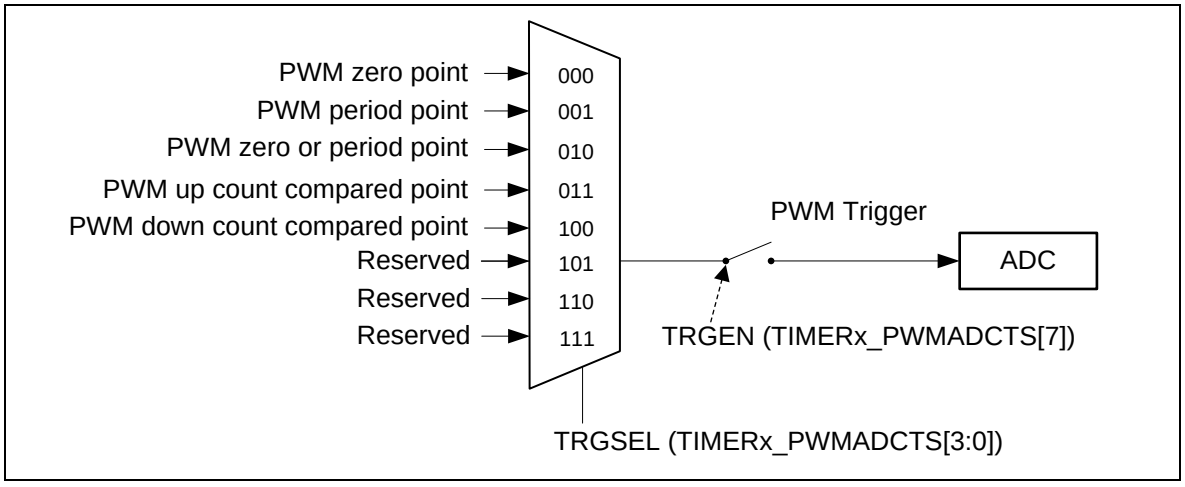


Figure 6.7-32 PWM Trigger ADC Block Diagram

6.7.7 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 = 0x4005_0000** **TMR23_BA = 0x4005_1000**				
TIMER0_CTL	TMR01_BA+0x00	R/W	Timer0 Control Register	0x0000_0005
TIMER0_CMP	TMR01_BA+0x04	R/W	Timer0 Comparator Register	0x0000_0000
TIMER0_INTSTS	TMR01_BA+0x08	R/W	Timer0 Interrupt Status Register	0x0000_0000
TIMER0_CNT	TMR01_BA+0x0C	R/W	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
TIMER0_TRGCTL	TMR01_BA+0x1C	R/W	Timer0 Trigger Control Register	0x0000_0000
TIMER0_ALTCTL	TMR01_BA+0x20	R/W	Timer0 Alternative Control Register	0x0000_0000
TIMER0_PWMCTL	TMR01_BA+0x40	R/W	Timer0 PWM Control Register	0x0000_0000
TIMER0_PWMCLKSRC	TMR01_BA+0x44	R/W	Timer0 PWM Counter Clock Source Register	0x0000_0000
TIMER0_PWMCLKPSC	TMR01_BA+0x48	R/W	Timer0 PWM Counter Clock Pre-scale Register	0x0000_0000
TIMER0_PWMCNTCLR	TMR01_BA+0x4C	R/W	Timer0 PWM Clear Counter Register	0x0000_0000
TIMER0_PWMPERIOD	TMR01_BA+0x50	R/W	Timer0 PWM Period Register	0x0000_0000
TIMER0_PWMCMPDAT	TMR01_BA+0x54	R/W	Timer0 PWM Comparator Register	0x0000_0000
TIMER0_PWMDTCTL	TMR01_BA+0x58	R/W	Timer0 PWM Dead-Time Control Register	0x0000_0000
TIMER0_PWMCNT	TMR01_BA+0x5C	R	Timer0 PWM Counter Register	0x0000_0000
TIMER0_PWMMSKEN	TMR01_BA+0x60	R/W	Timer0 PWM Output Mask Enable Register	0x0000_0000
TIMER0_PWMMSK	TMR01_BA+0x64	R/W	Timer0 PWM Output Mask Data Control Register	0x0000_0000
TIMER0_PWMPOLCTL	TMR01_BA+0x74	R/W	Timer0 PWM Pin Output Polar Control Register	0x0000_0000
TIMER0_PWMPOEN	TMR01_BA+0x78	R/W	Timer0 PWM Pin Output Enable Register	0x0000_0000
TIMER0_PWMINTEN0	TMR01_BA+0x80	R/W	Timer0 PWM Interrupt Enable Register 0	0x0000_0000
TIMER0_PWMINTSTS0	TMR01_BA+0x88	R/W	Timer0 PWM Interrupt Status Register 0	0x0000_0000
TIMER0_PWMADCTS	TMR01_BA+0x90	R/W	Timer0 PWM ADC Trigger Source Select Register	0x0000_0000
TIMER0_PWMSCTL	TMR01_BA+0x94	R/W	Timer0 PWM Synchronous Control Register	0x0000_0000
TIMER0_PWMSTRG	TMR01_BA+0x98	W	Timer0 PWM Synchronous Trigger Register	0x0000_0000

Register	Offset	R/W	Description	Reset Value
TIMER Base Address: **TMR01_BA = 0x4005_0000** **TMR23_BA = 0x4005_1000**				
TIMER0_PWMSTATUS	TMR01_BA+0x9C	R/W	Timer0 PWM Status Register	0x0000_0000
TIMER0_PWMPBUF	TMR01_BA+0xA0	R	Timer0 PWM Period Buffer Register	0x0000_0000
TIMER0_PWMCMPBUF	TMR01_BA+0xA4	R	Timer0 PWM Comparator Buffer Register	0x0000_0000
TIMER1_CTL	TMR01_BA+0x100	R/W	Timer1 Control Register	0x0000_0005
TIMER1_CMP	TMR01_BA+0x104	R/W	Timer1 Comparator Register	0x0000_0000
TIMER1_INTSTS	TMR01_BA+0x108	R/W	Timer1 Interrupt Status Register	0x0000_0000
TIMER1_CNT	TMR01_BA+0x10C	R/W	Timer1 Data Register	0x0000_0000
TIMER1_CAP	TMR01_BA+0x110	R	Timer1 Capture Data Register	0x0000_0000
TIMER1_EXTCTL	TMR01_BA+0x114	R/W	Timer1 External Control Register	0x0000_0000
TIMER1_EINTSTS	TMR01_BA+0x118	R/W	Timer1 External Interrupt Status Register	0x0000_0000
TIMER1_TRGCTL	TMR01_BA+0x11C	R/W	Timer1 Trigger Control Register	0x0000_0000
TIMER1_ALTCTL	TMR01_BA+0x120	R/W	Timer1 Alternative Control Register	0x0000_0000
TIMER1_PWMCTL	TMR01_BA+0x140	R/W	Timer1 PWM Control Register	0x0000_0000
TIMER1_PWMCLKSRC	TMR01_BA+0x144	R/W	Timer1 PWM Counter Clock Source Register	0x0000_0000
TIMER1_PWMCLKPSC	TMR01_BA+0x148	R/W	Timer1 PWM Counter Clock Pre-scale Register	0x0000_0000
TIMER1_PWMCNTCLR	TMR01_BA+0x14C	R/W	Timer1 PWM Clear Counter Register	0x0000_0000
TIMER1_PWMPERIOD	TMR01_BA+0x150	R/W	Timer1 PWM Period Register	0x0000_0000
TIMER1_PWMCMPDAT	TMR01_BA+0x154	R/W	Timer1 PWM Comparator Register	0x0000_0000
TIMER1_PWMDTCTL	TMR01_BA+0x158	R/W	Timer1 PWM Dead-Time Control Register	0x0000_0000
TIMER1_PWMCNT	TMR01_BA+0x15C	R	Timer1 PWM Counter Register	0x0000_0000
TIMER1_PWMMSKEN	TMR01_BA+0x160	R/W	Timer1 PWM Output Mask Enable Register	0x0000_0000
TIMER1_PWMMSK	TMR01_BA+0x164	R/W	Timer1 PWM Output Mask Data Control Register	0x0000_0000
TIMER1_PWMPOLCTL	TMR01_BA+0x174	R/W	Timer1 PWM Pin Output Polar Control Register	0x0000_0000
TIMER1_PWMPOEN	TMR01_BA+0x178	R/W	Timer1 PWM Pin Output Enable Register	0x0000_0000
TIMER1_PWMINTEN0	TMR01_BA+0x180	R/W	Timer1 PWM Interrupt Enable Register 0	0x0000_0000
TIMER1_PWMINTSTS0	TMR01_BA+0x188	R/W	Timer1 PWM Interrupt Status Register 0	0x0000_0000
TIMER1_PWMADCTS	TMR01_BA+0x190	R/W	Timer1 PWM ADC Trigger Source Select Register	0x0000_0000
TIMER1_PWMSCTL	TMR01_BA+0x194	R/W	Timer1 PWM Synchronous Control Register	0x0000_0000

Register	Offset	R/W	Description	Reset Value
TIMER Base Address: **TMR01_BA = 0x4005_0000** **TMR23_BA = 0x4005_1000**				
TIMER1_PWMSSTRG	TMR01_BA+0x198	W	Timer1 PWM Synchronous Start Trigger Register	0x0000_0000
TIMER1_PWMSTATUS	TMR01_BA+0x19C	R/W	Timer1 PWM Status Register	0x0000_0000
TIMER1_PWMPBUF	TMR01_BA+0x1A0	R	Timer1 PWM Period Buffer Register	0x0000_0000
TIMER1_PWMCMPBUF	TMR01_BA+0x1A4	R	Timer1 PWM Comparator Buffer Register	0x0000_0000
TIMER2_CTL	TMR23_BA+0x00	R/W	Timer2 Control Register	0x0000_0005
TIMER2_CMP	TMR23_BA+0x04	R/W	Timer2 Comparator Register	0x0000_0000
TIMER2_INTSTS	TMR23_BA+0x08	R/W	Timer2 Interrupt Status Register	0x0000_0000
TIMER2_CNT	TMR23_BA+0x0C	R/W	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
TIMER2_TRGCTL	TMR23_BA+0x1C	R/W	Timer2 Trigger Control Register	0x0000_0000
TIMER2_ALTCTL	TMR23_BA+0x20	R/W	Timer2 Alternative Control Register	0x0000_0000
TIMER2_PWMCTL	TMR23_BA+0x40	R/W	Timer2 PWM Control Register	0x0000_0000
TIMER2_PWMCLKSRC	TMR23_BA+0x44	R/W	Timer2 PWM Counter Clock Source Register	0x0000_0000
TIMER2_PWMCLKPSC	TMR23_BA+0x48	R/W	Timer2 PWM Counter Clock Pre-scale Register	0x0000_0000
TIMER2_PWMCNTCLR	TMR23_BA+0x4C	R/W	Timer2 PWM Clear Counter Register	0x0000_0000
TIMER2_PWMPERIOD	TMR23_BA+0x50	R/W	Timer2 PWM Period Register	0x0000_0000
TIMER2_PWMCMPDAT	TMR23_BA+0x54	R/W	Timer2 PWM Comparator Register	0x0000_0000
TIMER2_PWMDTCTL	TMR23_BA+0x58	R/W	Timer2 PWM Dead-Time Control Register	0x0000_0000
TIMER2_PWMCNT	TMR23_BA+0x5C	R	Timer2 PWM Counter Register	0x0000_0000
TIMER2_PWMMSKEN	TMR23_BA+0x60	R/W	Timer2 PWM Output Mask Enable Register	0x0000_0000
TIMER2_PWMMSK	TMR23_BA+0x64	R/W	Timer2 PWM Output Mask Data Control Register	0x0000_0000
TIMER2_PWMPOLCTL	TMR23_BA+0x74	R/W	Timer2 PWM Pin Output Polar Control Register	0x0000_0000
TIMER2_PWMPOEN	TMR23_BA+0x78	R/W	Timer2 PWM Pin Output Enable Register	0x0000_0000
TIMER2_PWMINTEN0	TMR23_BA+0x80	R/W	Timer2 PWM Interrupt Enable Register 0	0x0000_0000
TIMER2_PWMINTSTS0	TMR23_BA+0x88	R/W	Timer2 PWM Interrupt Status Register 0	0x0000_0000
TIMER2_PWMADCTS	TMR23_BA+0x90	R/W	Timer2 PWM ADC Trigger Source Select Register	0x0000_0000

Register	Offset	R/W	Description	Reset Value
TIMER Base Address: **TMR01_BA = 0x4005_0000** **TMR23_BA = 0x4005_1000**				
TIMER2_PWMSCTL	TMR23_BA+0x94	R/W	Timer2 PWM Synchronous Control Register	0x0000_0000
TIMER2_PWMSTRG	TMR23_BA+0x98	W	Timer2 PWM Synchronous Trigger Register	0x0000_0000
TIMER2_PWMSTATUS	TMR23_BA+0x9C	R/W	Timer2 PWM Status Register	0x0000_0000
TIMER2_PWMPBUF	TMR23_BA+0xA0	R	Timer2 PWM Period Buffer Register	0x0000_0000
TIMER2_PWMCMPBUF	TMR23_BA+0xA4	R	Timer2 PWM Comparator Buffer Register	0x0000_0000
TIMER3_CTL	TMR23_BA+0x100	R/W	Timer3 Control Register	0x0000_0005
TIMER3_CMP	TMR23_BA+0x104	R/W	Timer3 Comparator Register	0x0000_0000
TIMER3_INTSTS	TMR23_BA+0x108	R/W	Timer3 Interrupt Status Register	0x0000_0000
TIMER3_CNT	TMR23_BA+0x10C	R/W	Timer3 Data Register	0x0000_0000
TIMER3_CAP	TMR23_BA+0x110	R	Timer3 Capture Data Register	0x0000_0000
TIMER3_EXTCTL	TMR23_BA+0x114	R/W	Timer3 External Control Register	0x0000_0000
TIMER3_EINTSTS	TMR23_BA+0x118	R/W	Timer3 External Interrupt Status Register	0x0000_0000
TIMER3_TRGCTL	TMR23_BA+0x11C	R/W	Timer3 Trigger Control Register	0x0000_0000
TIMER3_ALTCTL	TMR23_BA+0x120	R/W	Timer3 Alternative Control Register	0x0000_0000
TIMER3_PWMCTL	TMR23_BA+0x140	R/W	Timer3 PWM Control Register	0x0000_0000
TIMER3_PWMCLKSRC	TMR23_BA+0x144	R/W	Timer3 PWM Counter Clock Source Register	0x0000_0000
TIMER3_PWMCLKPSC	TMR23_BA+0x148	R/W	Timer3 PWM Counter Clock Pre-scale Register	0x0000_0000
TIMER3_PWMCNTCLR	TMR23_BA+0x14C	R/W	Timer3 PWM Clear Counter Register	0x0000_0000
TIMER3_PWMPERIOD	TMR23_BA+0x150	R/W	Timer3 PWM Period Register	0x0000_0000
TIMER3_PWMCMPDAT	TMR23_BA+0x154	R/W	Timer3 PWM Comparator Register	0x0000_0000
TIMER3_PWMDTCTL	TMR23_BA+0x158	R/W	Timer3 PWM Dead-Time Control Register	0x0000_0000
TIMER3_PWMCNT	TMR23_BA+0x15C	R	Timer3 PWM Counter Register	0x0000_0000
TIMER3_PWMMSKEN	TMR23_BA+0x160	R/W	Timer3 PWM Output Mask Enable Register	0x0000_0000
TIMER3_PWMMSK	TMR23_BA+0x164	R/W	Timer3 PWM Output Mask Data Control Register	0x0000_0000
TIMER3_PWMPOLCTL	TMR23_BA+0x174	R/W	Timer3 PWM Pin Output Polar Control Register	0x0000_0000
TIMER3_PWMPOEN	TMR23_BA+0x178	R/W	Timer3 PWM Pin Output Enable Register	0x0000_0000
TIMER3_PWMINTEN0	TMR23_BA+0x180	R/W	Timer3 PWM Interrupt Enable Register 0	0x0000_0000
TIMER3_PWMINTSTS0	TMR23_BA+0x188	R/W	Timer3 PWM Interrupt Status Register 0	0x0000_0000

Register	Offset	R/W	Description	Reset Value
TIMER Base Address: **TMR01_BA = 0x4005_0000** **TMR23_BA = 0x4005_1000**				
TIMER3_PWMADCTS	TMR23_BA+0x190	R/W	Timer3 PWM ADC Trigger Source Select Register	0x0000_0000
TIMER3_PWMSCTL	TMR23_BA+0x194	R/W	Timer3 PWM Synchronous Control Register	0x0000_0000
TIMER3_PWMSSTRG	TMR23_BA+0x198	W	Timer3 PWM Synchronous Start Trigger Register	0x0000_0000
TIMER3_PWMSTATUS	TMR23_BA+0x19C	R/W	Timer3 PWM Status Register	0x0000_0000
TIMER3_PWMPBUF	TMR23_BA+0x1A0	R	Timer3 PWM Period Buffer Register	0x0000_0000
TIMER3_PWMCMPBUF	TMR23_BA+0x1A4	R	Timer3 PWM Comparator Buffer Register	0x0000_0000

Note:

1. Any register not listed here is reserved and must not be written. The result of a read operation on these bits is undefined.
2. The reserved register fields that listed in register description must be written to their reset value. Writing reserved fields with other than reset values may produce undefined results.

6.7.8 Register Description

Timer Control Register (TIMERx_CTL)

Register	Offset	R/W	Description	Reset Value
TIMER0_CTL	TMR01_BA+0x00	R/W	Timer0 Control Register	0x0000_0005
TIMER1_CTL	TMR01_BA+0x100	R/W	Timer1 Control Register	0x0000_0005
TIMER2_CTL	TMR23_BA+0x00	R/W	Timer2 Control Register	0x0000_0005
TIMER3_CTL	TMR23_BA+0x100	R/W	Timer3 Control Register	0x0000_0005

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

Bits	Description	
[31]	**ICEDEBUG**	**ICE Debug Mode Acknowledge Disable Control (Write Protected)** 0 = ICE debug mode acknowledgement effects TIMER counting. TIMER counter will be held while CPU is held by ICE. 1 = ICE debug mode acknowledgement Disabled. TIMER counter will keep going no matter CPU is held by ICE or not. **Note:** This bit is write protected. Refer to the 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 OPMODE (TIMER_CTL[28:27] = 00) when the timer time-out interrupt flag TIF (TIMERx_INTSTS[0]) is generated. **Note3:** Set enable/disable this bit needs 2 * TMR_CLK period to become active, user can read ACTSTS (TIMERx_CTL[25]) to check enable/disable command is completed or not.
[29]	**INTEN**	**Timer Interrupt Enable Bit** 0 = Timer time-out interrupt Disabled. 1 = Timer time-out interrupt Enabled. **Note:** If this bit is enabled, when the timer time-out interrupt flag TIF is set to 1, the timer interrupt signal is generated and inform to CPU.
[28:27]	**OPMODE**	**Timer Counting 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]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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. **Note:** This bit may active when CNT 0 transition to CNT 1.
[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]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[21]	**TGLPINSEL**	**Toggle-output Pin Select** 0 = Toggle mode output to TMx (Timer Event Counter Pin). 1 = Toggle mode output to TMx_EXT (Timer External Capture Pin).
[20]	**PERIOSEL**	**Periodic Mode Behavior Selection Enable Bit** 0 = The behavior selection in periodic mode is Disabled. When user updates CMPDAT while timer is running in periodic mode, CNT will be reset to default value. 1 = The behavior selection in periodic mode is Enabled. When user update CMPDAT while timer is running in periodic mode, the limitations as bellows list, If updated CMPDAT value > CNT, CMPDAT will be updated and CNT keep running continually. If updated CMPDAT value = CNT, timer time-out interrupt will be asserted immediately. If updated CMPDAT value < CNT, CNT will be reset to default value.
[19]	**INTRGEN**	**Inter-timer Trigger Mode Enable Control** Setting this bit will enable the inter-timer trigger capture function. The Timer0/2 will be in event counter mode and counting with external clock source or event. Also, Timer1/3 will be in trigger-counting mode of capture function. 0 = Inter-Timer Trigger Capture mode Disabled. 1 = Inter-Timer Trigger Capture mode Enabled. **Note:** For Timer1/3, this bit is ignored and the read back value is always 0.
[18:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[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. **Note:** Overwriting prescale counter value will reset internal 8-bit prescale counter and 24-bit up counter value.

Timer Comparator Register (TIMERx_CMP)

Register	Offset	R/W	Description	Reset Value
TIMER0_CMP	TMR01_BA+0x04	R/W	Timer0 Comparator Register	0x0000_0000
TIMER1_CMP	TMR01_BA+0x104	R/W	Timer1 Comparator Register	0x0000_0000
TIMER2_CMP	TMR23_BA+0x04	R/W	Timer2 Comparator Register	0x0000_0000
TIMER3_CMP	TMR23_BA+0x104	R/W	Timer3 Comparator 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. Any values read should be ignored. When writing to this field always write with reset value.
[23:0]	**CMPDAT**	**Timer Comparator 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+0x108	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+0x108	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. Any values read should be ignored. When writing to this field always write with reset value.
[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 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/W	Timer0 Data Register	0x0000_0000
TIMER1_CNT	TMR01_BA+0x10C	R/W	Timer1 Data Register	0x0000_0000
TIMER2_CNT	TMR23_BA+0x0C	R/W	Timer2 Data Register	0x0000_0000
TIMER3_CNT	TMR23_BA+0x10C	R/W	Timer3 Data Register	0x0000_0000

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

Bits	Description	
[31]	**RSTACT**	**Timer Data Register Reset Active (Read Only)** This bit indicates if the counter reset operation active. When user writes this CNT register, timer starts to reset its internal 24-bit timer up-counter to 0 and reload 8-bit pre-scale counter. At the same time, timer set this flag to 1 to indicate the counter reset operation is in progress. Once the counter reset operation done, timer clear this bit to 0 automatically. 0 = Reset operation is done. 1 = Reset operation triggered by writing TIMERx_CNT is in progress. **Note:** This bit is read only.
[30:24]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[23:0]	**CNT**	**Timer Data Register** Read operation. Read this register to get CNT value. For example: If EXTCNTEN (TIMERx_CTL[24]) is 0, user can read CNT value for getting current 24-bit counter value. If EXTCNTEN (TIMERx_CTL[24]) is 1, user can read CNT value for getting current 24-bit event input counter value. Write operation. Writing any value to this register will reset current CNT value to 0 and reload internal 8-bit prescale counter.

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+0x110	R	Timer1 Capture Data Register	0x0000_0000
TIMER2_CAP	TMR23_BA+0x10	R	Timer2 Capture Data Register	0x0000_0000
TIMER3_CAP	TMR23_BA+0x110	R	Timer3 Capture Data Register	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[23:0]	**CAPDAT**	**Timer Capture Data Register (Read Only)** When CAPEN (TIMERx_EXTCTL[3]) bit is set, CAPFUNCS (TIMERx_EXTCTL[4]) bit is 0, and a transition on TMx_EXT pin matched the CAPEDGE (TIMERx_EXTCTL[14:12]) 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+0x114	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+0x114	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							**ECNTSSEL**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	**CAPEDGE**			**Reserved**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNTDBEN	**CAPDBEN**	**CAPIEN**	**CAPFUNCS**	**CAPEN**	**Reserved**		**CNTPHASE**

Bits	Description	
[31:17]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[16]	**ECNTSSEL**	**Event Counter Source Selection to Trigger Event Counter Function** 0 = Event Counter input source is from TMx (x= 0~3) pin. 1 = Reserved Event Counter input source is from USB internal SOF output signal.
[15]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[14:12]	**CAPEDGE**	**Timer External Capture Pin Edge Detect** When first capture event is generated, the CNT (TIMERx_CNT[23:0]) will be reset to 0 and first CAPDAT (TIMERx_CAP[23:0]) should be to 0. 000 = Capture event occurred when detect falling edge transfer on TMx_EXT (x= 0~3) pin. 001 = Capture event occurred when detect rising edge transfer on TMx_EXT (x= 0~3) pin. 010 = Capture event occurred when detect both falling and rising edge transfer on TMx_EXT (x= 0~3) pin, and first capture event occurred at falling edge transfer. 011 = Capture event occurred when detect both rising and falling edge transfer on TMx_EXT (x= 0~3) pin, and first capture event occurred at rising edge transfer.. 110 = First capture event occurred at falling edge, follows capture events are at rising edge transfer on TMx_EXT (x= 0~3) pin. 111 = First capture event occurred at rising edge, follows capture events are at falling edge transfer on TMx_EXT (x= 0~3) pin. 100, 101 = Reserved. Do not use.
[11:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7]	**CNTDBEN**	**Timer Counter Pin De-bounce Enable Bit**

		0 = TMx (x= 0~3) pin de-bounce Disabled. 1 = TMx (x= 0~3) pin de-bounce Enabled. **Note:** If this bit is enabled, the edge detection of TMx pin is detected with de-bounce circuit.
[6]	**CAPDBEN**	**Timer External Capture Pin De-bounce Enable Bit** 0 = TMx_EXT (x= 0~3) pin de-bounce Disabled. 1 = TMx_EXT (x= 0~3) pin de-bounce Enabled. **Note:** If this bit is enabled, the edge detection of TMx_EXT pin output is detected with de-bounce circuit.
[5]	**CAPIEN**	**Timer External Capture Interrupt Enable Bit** 0 = TMx_EXT (x= 0~3) pin detection Interrupt Disabled. 1 = TMx_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 TMx_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 TMx_EXT (x= 0~3) pin is using to save current 24-bit timer counter value (CNT value) to CAPDAT field. **Note2:** When CAPFUNCS is 1, transition on TMx_EXT (x= 0~3) pin is using to save current 24-bit timer counter value (CNT value) to CAPDAT field then CNT value will be reset immediately.
[3]	**CAPEN**	**Timer External Capture Pin Enable Bit** This bit enables the TMx_EXT capture pin input function. 0 =TMx_EXT (x= 0~3) pin Disabled. 1 =TMx_EXT (x= 0~3) pin Enabled.
[2:1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**CNTPHASE**	**Timer External Count Phase** This bit indicates the detection phase of external counting pin TMx (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+0x118	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+0x118	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. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**CAPIF**	**Timer External Capture Interrupt Flag** This bit indicates the timer external capture interrupt flag status. 0 = TMx_EXT (x= 0~3) pin interrupt did not occur. 1 = TMx_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 TMx_EXT (x= 0~3) pin matched 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.

Timer Trigger Control Register (TIMERx_TRGCTL)

Register	Offset	R/W	Description	Reset Value
TIMER0_TRGCTL	TMR01_BA+0x1C	R/W	Timer0 Trigger Control Register	0x0000_0000
TIMER1_TRGCTL	TMR01_BA+0x11C	R/W	Timer1 Trigger Control Register	0x0000_0000
TIMER2_TRGCTL	TMR23_BA+0x1C	R/W	Timer2 Trigger Control Register	0x0000_0000
TIMER3_TRGCTL	TMR23_BA+0x11C	R/W	Timer3 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**	**3**	**4**	**3**	**2**	**1**	**0**
Reserved			**TRGPDMA**	**Reserved**	**TRGADC**	**TRGPWM**	**TRGSSEL**

Bits	Description	
[31:5]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[4]	**TRGPDMA**	**Trigger PDMA Enable Bit** If this bit is set to 1, each timer time-out event or capture event can be triggered PDMA transfer. 0 = Timer interrupt trigger PDMA Disabled. 1 = Timer interrupt trigger PDMA Enabled. **Note:** If TRGSSEL (TIMERx_TRGCTL[0]) = 0, time-out interrupt signal will trigger PDMA transfer. If TRGSSEL (TIMERx_TRGCTL[0]) = 1, capture interrupt signal will trigger PDMA transfer.
[3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**TRGADC**	**Trigger ADC Enable Bit** If this bit is set to 1, each timer time-out event or capture event can be triggered ADC conversion. 0 = Timer interrupt trigger ADC Disabled. 1 = Timer interrupt trigger ADC Enabled. **Note:** If TRGSSEL (TIMERx_TRGCTL[0]) = 0, time-out interrupt signal will trigger ADC conversion. If TRGSSEL (TIMERx_TRGCTL[0]) = 1, capture interrupt signal will trigger ADC conversion.
[1]	**TRGPWM**	**Trigger PWM Enable Bit** If this bit is set to 1, each timer time-out event or capture event can be as PWM counter clock source.

		0 = Timer interrupt trigger PWM Disabled. 1 = Timer interrupt trigger PWM Enabled. **Note:** If TRGSSEL (TIMERx_TRGCTL[0]) = 0, time-out interrupt signal as PWM counter clock source. If TRGSSEL (TIMERx_TRGCTL[0]) = 1, capture interrupt signal as PWM counter clock source.
[0]	**TRGSSEL**	**Trigger Source Select Bit** This bit is used to select internal trigger source is form timer time-out interrupt signal or capture interrupt signal. 0 = Time-out interrupt signal is used to internal trigger PWM, PDMA, and ADC. 1 = Capture interrupt signal is used to internal trigger PWM, PDMA, and ADC.

Timer Alternative Control Register (TIMERx_ALTCTL)

Register	Offset	R/W	Description	Reset Value
TIMER0_ALTCTL	TMR01_BA+0x20	R/W	Timer0 Alternative Control Register	0x0000_0000
TIMER1_ALTCTL	TMR01_BA+0x120	R/W	Timer1 Alternative Control Register	0x0000_0000
TIMER2_ALTCTL	TMR23_BA+0x20	R/W	Timer2 Alternative Control Register	0x0000_0000
TIMER3_ALTCTL	TMR23_BA+0x120	R/W	Timer3 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							**FUNCSEL**

Bits	Description	
[31:1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**FUNCSEL**	**Function Selection** 0 = Timer controller is used as timer function. 1 = Timer controller is used as PWM function. **Note:** When timer is used as PWM, the clock source of time controller will be forced to PCLKx automatically.

Timer PWM Control Register (TIMERx_PWMCTL)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMCTL	TMR01_BA+0x40	R/W	Timer0 PWM Control Register	0x0000_0000
TIMER1_PWMCTL	TMR01_BA+0x140	R/W	Timer1 PWM Control Register	0x0000_0000
TIMER2_PWMCTL	TMR23_BA+0x40	R/W	Timer2 PWM Control Register	0x0000_0000
TIMER3_PWMCTL	TMR23_BA+0x140	R/W	Timer3 PWM Control Register	0x0000_0000

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

Bits	Description	
[31]	**DBGTRIOFF**	**ICE Debug Mode Acknowledge Disable Bit (Write Protected)** 0 = ICE debug mode acknowledgement effects PWM output. PWM output pin will be forced as tri-state while ICE debug mode acknowledged. 1 = ICE debug mode acknowledgement disabled. PWM output pin will keep output no matter ICE debug mode acknowledged or not. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[30]	**DBGHALT**	**ICE Debug Mode Counter Halt (Write Protected)** If debug mode counter halt is enabled, PWM counter 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 register is write protected. Refer to SYS_REGLCTL register.
[29:17]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[16]	**OUTMODE**	**PWM Output Mode** This bit controls the output mode of corresponding PWM channel. 0 = PWM independent mode. 1 = PWM complementary mode.
[15:10]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[9]	**IMMLDEN**	**Immediately Load Enable Bit** 0 = PERIOD will load to PBUF when current PWM period is completed no matter CTRLD is enabled/disabled. If CTRLD is disabled, CMP will load to CMPBUF when current PWM

		period is completed; if CTRLD is enabled in up-down count type, CMP will load to CMPBUF at the center point of current period. 1 = PERIOD/CMP will load to PBUF/CMPBUF immediately when user update PERIOD/CMP. **Note:** If IMMLDEN is enabled, CTRLD will be invalid.
[8]	**CTRLD**	**Center Re-load** In up-down count type, PERIOD will load to PBUF when current PWM period is completed always and CMP will load to CMPBUF at the center point of current period.
[7:4]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[3]	**CNTMODE**	**PWM Counter Mode** 0 = Auto-reload mode. 1 = One-shot mode.
[2:1]	**CNTTYPE**	**PWM Counter Behavior Type** 00 = Up count type. 01 = Down count type. 10 = Up-down count type. 11 = Reserved. Do not use.
[0]	**CNTEN**	**PWM Counter Enable Bit** 0 = PWM counter and clock prescale Stop Running. 1 = PWM counter and clock prescale Start Running.

Timer PWM Counter Clock Source Register (TIMERx PWMCLKSRC)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMCLKSRC	TMR01_BA+0x44	R/W	Timer0 PWM Counter Clock Source Register	0x0000_0000
TIMER1_PWMCLKSRC	TMR01_BA+0x144	R/W	Timer1 PWM Counter Clock Source Register	0x0000_0000
TIMER2_PWMCLKSRC	TMR23_BA+0x44	R/W	Timer2 PWM Counter Clock Source Register	0x0000_0000
TIMER3_PWMCLKSRC	TMR23_BA+0x144	R/W	Timer3 PWM Counter 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					**CLKSRC**		

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**CLKSRC**	**PWM Counter Clock Source Select** The PWM counter clock source can be selected from TMRx_CLK or internal timer time-out or capture event. 000 = TMRx_CLK. 001 = Internal TIMER0 time-out or capture event. 010 = Internal TIMER1 time-out or capture event. 011 = Internal TIMER2 time-out or capture event. 100 = Internal TIMER3 time-out or capture event. Others = Reserved. Do not use. **Note:** If TIMER0 PWM function is enabled, the PWM counter clock source can be selected from TMR0_CLK, TIMER1 interrupt events, TIMER2 interrupt events, or TIMER3 interrupt events.

Timer PWM Counter Clock Pre-scale Register (TIMERx PWMCLKPSC)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMCLKPSC	TMR01_BA+0x48	R/W	Timer0 PWM Counter Clock Pre-scale Register	0x0000_0000
TIMER1_PWMCLKPSC	TMR01_BA+0x148	R/W	Timer1 PWM Counter Clock Pre-scale Register	0x0000_0000
TIMER2_PWMCLKPSC	TMR23_BA+0x48	R/W	Timer2 PWM Counter Clock Pre-scale Register	0x0000_0000
TIMER3_PWMCLKPSC	TMR23_BA+0x148	R/W	Timer3 PWM Counter 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. Any values read should be ignored. When writing to this field always write with reset value.
[11:0]	**CLKPSC**	**PWM Counter Clock Pre-scale** The active clock of PWM counter is decided by counter clock prescale and divided by (CLKPSC + 1). If CLKPSC is 0, then there is no scaling in PWM counter clock source.

Timer PWM Clear Counter Register (TIMERx PWMCNTCLR)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMCNTCLR	TMR01_BA+0x4C	R/W	Timer0 PWM Clear Counter Register	0x0000_0000
TIMER1_PWMCNTCLR	TMR01_BA+0x14C	R/W	Timer1 PWM Clear Counter Register	0x0000_0000
TIMER2_PWMCNTCLR	TMR23_BA+0x4C	R/W	Timer2 PWM Clear Counter Register	0x0000_0000
TIMER3_PWMCNTCLR	TMR23_BA+0x14C	R/W	Timer3 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							**CNTCLR**

Bits	Description	
[31:1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**CNTCLR**	**Clear PWM Counter Control Bit** It is automatically cleared by hardware. 0 = No effect. 1 = Clear 16-bit PWM counter to 0x10000 in up and up-down count type and reset counter value to PERIOD in down count type.

Timer PWM Period Register (TIMERx PWMPERIOD)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMPERIOD	TMR01_BA+0x50	R/W	Timer0 PWM Period Register	0x0000_0000
TIMER1_PWMPERIOD	TMR01_BA+0x150	R/W	Timer1 PWM Period Register	0x0000_0000
TIMER2_PWMPERIOD	TMR23_BA+0x50	R/W	Timer2 PWM Period Register	0x0000_0000
TIMER3_PWMPERIOD	TMR23_BA+0x150	R/W	Timer3 PWM 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. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**PERIOD**	**PWM Period Register** In up count type: PWM counter counts from 0 to PERIOD, and restarts from 0. In down count type: PWM counter counts from PERIOD to 0, and restarts from PERIOD. In up-down count type: PWM counter counts from 0 to PERIOD, then decrements to 0 and repeats again. In up and down count type: PWM period time = (PERIOD + 1) * (CLKPSC + 1) * TMRx_PWMCLK. In up-down count type: PWM period time = 2 * PERIOD * (CLKPSC+ 1) * TMRx_PWMCLK. **Note:** User should take care DIRF (TIMERx_PWMCNT[16]) bit in up/down/up-down count type to monitor current counter direction in each count type.

Timer PWM Comparator Register (TIMERx PWMCMPDAT)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMCMPDAT	TMR01_BA+0x54	R/W	Timer0 PWM Comparator Register	0x0000_0000
TIMER1_PWMCMPDAT	TMR01_BA+0x154	R/W	Timer1 PWM Comparator Register	0x0000_0000
TIMER2_PWMCMPDAT	TMR23_BA+0x54	R/W	Timer2 PWM Comparator Register	0x0000_0000
TIMER3_PWMCMPDAT	TMR23_BA+0x154	R/W	Timer3 PWM Comparator Register	0x0000_0000

31	30	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. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**CMP**	**PWM Comparator Register** PWM CMP is used to compare with PWM CNT to generate PWM output waveform, interrupt events and trigger ADC to start convert.

Timer PWM Dead-Time Control Register (TIMERx PWMDTCTL)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMDTCTL	TMR01_BA+0x58	R/W	Timer0 PWM Dead-Time Control Register	0x0000_0000
TIMER1_PWMDTCTL	TMR01_BA+0x158	R/W	Timer1 PWM Dead-Time Control Register	0x0000_0000
TIMER2_PWMDTCTL	TMR23_BA+0x58	R/W	Timer2 PWM Dead-Time Control Register	0x0000_0000
TIMER3_PWMDTCTL	TMR23_BA+0x158	R/W	Timer3 PWM Dead-Time Control Register	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. Any values read should be ignored. When writing to this field always write with reset value.
[24]	**DTCKSEL**	**Dead-time Clock Select (Write Protected)** 0 = Dead-time clock source from TMRx_PWMCLK without counter clock prescale. 1 = Dead-time clock source from TMRx_PWMCLK with counter clock prescale. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[23:17]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[16]	**DTEN**	**Enable Dead-time Insertion for PWMx_CH0 and PWMx_CH1 (Write Protected)** Dead-time insertion function is only active when PWM complementary mode is enabled. If dead- time insertion is inactive, the outputs of PWMx_CH0 and PWMx_CH1 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 register is write protected. Refer to SYS_REGLCTL register.
[15:12]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[11:0]	**DTCNT**	**Dead-time Counter (Write Protected)** The dead-time can be calculated from the following two formulas: Dead-time = (DTCNT[11:0] + 1) * TMRx_PWMCLK, if DTCKSEL is 0. Dead-time = (DTCNT[11:0] + 1) * TMRx_PWMCLK * (CLKPSC + 1), if DTCKSEL is 1. **Note:** This register is write protected. Refer to SYS_REGLCTL register.

Timer PWM Counter Register (TIMERx PWMCNT)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMCNT	TMR01_BA+0x5C	R	Timer0 PWM Counter Register	0x0000_0000
TIMER1_PWMCNT	TMR01_BA+0x15C	R	Timer1 PWM Counter Register	0x0000_0000
TIMER2_PWMCNT	TMR23_BA+0x5C	R	Timer2 PWM Counter Register	0x0000_0000
TIMER3_PWMCNT	TMR23_BA+0x15C	R	Timer3 PWM Counter Register	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. Any values read should be ignored. When writing to this field always write with reset value.
[16]	**DIRF**	**PWM Counter Direction Indicator Flag (Read Only)** 0 = Counter is active in down count. 1 = Counter is active up count.
[15:0]	**CNT**	**PWM Counter Value Register (Read Only)** User can monitor CNT to know the current counter value in 16-bit period counter.

Timer PWM Output Mask Enable Register (TIMERx PWMMSKEN)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMMSKEN	TMR01_BA+0x60	R/W	Timer0 PWM Output Mask Enable Register	0x0000_0000
TIMER1_PWMMSKEN	TMR01_BA+0x160	R/W	Timer1 PWM Output Mask Enable Register	0x0000_0000
TIMER2_PWMMSKEN	TMR23_BA+0x60	R/W	Timer2 PWM Output Mask Enable Register	0x0000_0000
TIMER3_PWMMSKEN	TMR23_BA+0x160	R/W	Timer3 PWM Output 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						**MSKEN1**	**MSKEN0**

Bits	Description	
[31:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1]	**MSKEN1**	**PWMx_CH1 Output Mask Enable Bit** The PWMx_CH1 output signal will be masked when this bit is enabled. The PWMx_CH1 will output MSKDAT1 (TIMER_PWMMSK[1]) data. 0 = PWMx_CH1 output signal is non-masked. 1 = PWMx_CH1 output signal is masked and output MSKDAT1 data.
[0]	**MSKEN0**	**PWMx_CH0 Output Mask Enable Bit** The PWMx_CH0 output signal will be masked when this bit is enabled. The PWMx_CH0 will output MSKDAT0 (TIMER_PWMMSK[0]) data. 0 = PWMx_CH0 output signal is non-masked. 1 = PWMx_CH0 output signal is masked and output MSKDAT0 data.

Timer PWM Output Mask Data Control Register (TIMERx PWMMSK)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMMSK	TMR01_BA+0x64	R/W	Timer0 PWM Output Mask Data Control Register	0x0000_0000
TIMER1_PWMMSK	TMR01_BA+0x164	R/W	Timer1 PWM Output Mask Data Control Register	0x0000_0000
TIMER2_PWMMSK	TMR23_BA+0x64	R/W	Timer2 PWM Output Mask Data Control Register	0x0000_0000
TIMER3_PWMMSK	TMR23_BA+0x164	R/W	Timer3 PWM Output Mask Data 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						**MSKDAT1**	**MSKDAT0**

Bits	Description	
[31:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1]	**MSKDAT1**	**PWMx_CH1 Output Mask Data Control Bit** This bit is used to control the output state of PWMx_CH1 pin when PWMx_CH1 output mask function is enabled (MSKEN1 = 1). 0 = Output logic Low to PWMx_CH1. 1 = Output logic High to PWMx_CH1.
[0]	**MSKDAT0**	**PWMx_CH0 Output Mask Data Control Bit** This bit is used to control the output state of PWMx_CH0 pin when PWMx_CH0 output mask function is enabled (MSKEN0 = 1). 0 = Output logic Low to PWMx_CH0. 1 = Output logic High to PWMx_CH0.

Timer PWM Pin Output Polar Control Register (TIMERx PWMPOLCTL)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMPOLCTL	TMR01_BA+0x74	R/W	Timer0 PWM Pin Output Polar Control Register	0x0000_0000
TIMER1_PWMPOLCTL	TMR01_BA+0x174	R/W	Timer1 PWM Pin Output Polar Control Register	0x0000_0000
TIMER2_PWMPOLCTL	TMR23_BA+0x74	R/W	Timer2 PWM Pin Output Polar Control Register	0x0000_0000
TIMER3_PWMPOLCTL	TMR23_BA+0x174	R/W	Timer3 PWM Pin Output Polar 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						**PINV1**	**PINV0**

Bits	Description	
[31:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1]	**PINV1**	**PWMx_CH1 Output Pin Polar Control Bit** The bit is used to control polarity state of PWMx_CH1 output pin. 0 = PWMx_CH1 output pin polar inverse Disabled. 1 = PWMx_CH1 output pin polar inverse Enabled.
[0]	**PINV0**	**PWMx_CH0 Output Pin Polar Control Bit** The bit is used to control polarity state of PWMx_CH0 output pin. 0 = PWMx_CH0 output pin polar inverse Disabled. 1 = PWMx_CH0 output pin polar inverse Enabled.

Timer PWM Pin Output Enable Register (TIMERx PWMPOEN)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMPOEN	TMR01_BA+0x78	R/W	Timer0 PWM Pin Output Enable Register	0x0000_0000
TIMER1_PWMPOEN	TMR01_BA+0x178	R/W	Timer1 PWM Pin Output Enable Register	0x0000_0000
TIMER2_PWMPOEN	TMR23_BA+0x78	R/W	Timer2 PWM Pin Output Enable Register	0x0000_0000
TIMER3_PWMPOEN	TMR23_BA+0x178	R/W	Timer3 PWM Pin 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						**POEN1**	**POEN0**

Bits	Description	
[31:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1]	**POEN1**	**PWMx_CH1 Output Pin Enable Bit** 0 = PWMx_CH1 pin at tri-state mode. 1 = PWMx_CH1 pin in output mode.
[0]	**POEN0**	**PWMx_CH0 Output Pin Enable Bit** 0 = PWMx_CH0 pin at tri-state mode. 1 = PWMx_CH0 pin in output mode.

Timer PWM Interrupt Enable Register 0 (TIMERx PWMINTEN0)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMINTEN0	TMR01_BA+0x80	R/W	Timer0 PWM Interrupt Enable Register 0	0x0000_0000
TIMER1_PWMINTEN0	TMR01_BA+0x180	R/W	Timer1 PWM Interrupt Enable Register 0	0x0000_0000
TIMER2_PWMINTEN0	TMR23_BA+0x80	R/W	Timer2 PWM Interrupt Enable Register 0	0x0000_0000
TIMER3_PWMINTEN0	TMR23_BA+0x180	R/W	Timer3 PWM Interrupt Enable 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				**CMPDIEN**	**CMPUIEN**	**PIEN**	**ZIEN**

Bits	Description	
[31:4]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[3]	**CMPDIEN**	**PWM Compare Down Count Interrupt Enable Bit** 0 = Compare down count interrupt Disabled. 1 = Compare down count interrupt Enabled.
[2]	**CMPUIEN**	**PWM Compare Up Count Interrupt Enable Bit** 0 = Compare up count interrupt Disabled. 1 = Compare up count interrupt Enabled.
[1]	**PIEN**	**PWM Period Point Interrupt Enable Bit** 0 = Period point interrupt Disabled. 1 = Period point interrupt Enabled. **Note:** When in up-down count type, period point means the center point of current PWM period.
[0]	**ZIEN**	**PWM Zero Point Interrupt Enable Bit** 0 = Zero point interrupt Disabled. 1 = Zero point interrupt Enabled.

Timer PWM Interrupt Status Register 0 (TIMERx PWMINTSTS0)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMINTSTS0	TMR01_BA+0x88	R/W	Timer0 PWM Interrupt Status Register 0	0x0000_0000
TIMER1_PWMINTSTS0	TMR01_BA+0x188	R/W	Timer1 PWM Interrupt Status Register 0	0x0000_0000
TIMER2_PWMINTSTS0	TMR23_BA+0x88	R/W	Timer2 PWM Interrupt Status Register 0	0x0000_0000
TIMER3_PWMINTSTS0	TMR23_BA+0x188	R/W	Timer3 PWM Interrupt Status 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				CMPDIF	CMPUIF	PIF	ZIF

Bits	Description	
[31:4]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[3]	**CMPDIF**	**PWM Compare Down Count Interrupt Flag** This bit is set by hardware when TIMERx_PWM counter in down count direction and reaches CMP. **Note1:** If CMP equal to PERIOD, there is no CMPDIF flag in down count type. **Note2:** This bit is cleared by writing 1 to it.
[2]	**CMPUIF**	**PWM Compare Up Count Interrupt Flag** This bit is set by hardware when TIMERx_PWM counter in up count direction and reaches CMP. **Note1:** If CMP equal to PERIOD, there is no CMPUIF flag in up count type and up-down count type.. **Note2:** This bit is cleared by writing 1 to it.
[1]	**PIF**	**PWM Period Point Interrupt Flag** This bit is set by hardware when TIMERx_PWM counter reaches PERIOD. **Note1:** When in up-down count type, PIF flag means the center point flag of current PWM period. **Note2:** This bit is cleared by writing 1 to it.
[0]	**ZIF**	**PWM Zero Point Interrupt Flag** This bit is set by hardware when TIMERx_PWM counter reaches zero. **Note:** This bit is cleared by writing 1 to it.

Timer PWM ADC Trigger Control Register (TIMERx PWMADCTS)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMADCTS	TMR01_BA+0x90	R/W	Timer0 PWM ADC Trigger Source Select Register	0x0000_0000
TIMER1_PWMADCTS	TMR01_BA+0x190	R/W	Timer1 PWM ADC Trigger Source Select Register	0x0000_0000
TIMER2_PWMADCTS	TMR23_BA+0x90	R/W	Timer2 PWM ADC Trigger Source Select Register	0x0000_0000
TIMER3_PWMADCTS	TMR23_BA+0x190	R/W	Timer3 PWM ADC Trigger 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**
TRGEN	**Reserved**				**TRGSEL**		

Bits	Description	
[31:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7]	**TRGEN**	**PWM Counter Event Trigger ADC Conversion Enable Bit** 0 = PWM counter event trigger ADC conversion Disabled. 1 = PWM counter event trigger ADC conversion Enabled.
[6:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**TRGSEL**	**PWM Counter Event Source Select to Trigger ADC Conversion** 000 = Trigger ADC conversion at zero point (ZIF). 001 = Trigger ADC conversion at period point (PIF). 010 = Trigger ADC conversion at zero or period point (ZIF or PIF). 011 = Trigger ADC conversion at compare up count point (CMPUIF). 100 = Trigger ADC conversion at compare down count point (CMPDIF). Others = Reserved. Do not use.

Timer PWM Synchronous Control Register (TIMERx PWMSCTL)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMSCTL	TMR01_BA+0x94	R/W	Timer0 PWM Synchronous Control Register	0x0000_0000
TIMER1_PWMSCTL	TMR01_BA+0x194	R/W	Timer1 PWM Synchronous Control Register	0x0000_0000
TIMER2_PWMSCTL	TMR23_BA+0x94	R/W	Timer2 PWM Synchronous Control Register	0x0000_0000
TIMER3_PWMSCTL	TMR23_BA+0x194	R/W	Timer3 PWM Synchronous 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							**SYNCSRC**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**SYNCMODE**	

Bits	Description	
[31:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8]	**SYNCSRC**	**PWM Synchronous Counter Start/Clear Source Select** 0 = Counter synchronous start/clear by trigger STRGEN (TIMER0_PWMSTRG[0]). 1 = Counter synchronous start/clear by trigger STRGEN (TIMER2_PWMSTRG[0]). **Note1:** If TIMER0/1/2/3 PWM counter synchronous source are from TIMER0, TIMER0_PWMSCTL[8], TIMER1_PWMSCTL[8], TIMER2_PWMSCTL[8] and TIMER3_PWMSCTL[8] should be 0. **Note2:** If TIMER0/1/ PWM counter synchronous source are from TIMER0, TIMER0_PWMSCTL[8] and TIMER1_PWMSCTL[8] should be set 0, and TIMER2/3/ PWM counter synchronous source are from TIMER2, TIMER2_PWMSCTL[8] and TIMER3_PWMSCTL[8] should be set 1.
[7:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1:0]	**SYNCMODE**	**PWM Synchronous Mode Enable Select** 00 = PWM synchronous function Disabled. 01 = PWM synchronous counter start function Enabled. 10 = Reserved. Do not use. 11 = PWM synchronous counter clear function Enabled.

Timer PWM Synchronous Trigger Register (TIMERx PWMSTRG)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMSTRG	TMR01_BA+0x98	W	Timer0 PWM Synchronous Trigger Register	0x0000_0000
TIMER2_PWMSTRG	TMR23_BA+0x98	W	Timer2 PWM Synchronous 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							**STRGEN**

Bits	Description	
[31:1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**STRGEN**	**PWM Counter Synchronous Trigger Enable Bit (Write Only)** PMW counter synchronous function is used to make selected PWM channels (include TIMER0/1/2/3 PWM, TIMER0/1 PWM and TIMER2/3 PWM) start counting or clear counter at the same time according to TIMERx_PWMSCTL setting. **Note:** This bit is only available in TIMER0 and TIMER2.

Timer PWM Status Register (TIMERx PWMSTATUS)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMSTATUS	TMR01_BA+0x9C	R/W	Timer0 PWM Status Register	0x0000_0000
TIMER1_PWMSTATUS	TMR01_BA+0x19C	R/W	Timer1 PWM Status Register	0x0000_0000
TIMER2_PWMSTATUS	TMR23_BA+0x9C	R/W	Timer2 PWM Status Register	0x0000_0000
TIMER3_PWMSTATUS	TMR23_BA+0x19C	R/W	Timer3 PWM Status Register	0x0000_0000

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

Bits	Description	
[31:17]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[16]	**ADCTRGF**	**Trigger ADC Start Conversion Flag** 0 = PWM counter event trigger ADC start conversion is not occurred. 1 = PWM counter event trigger ADC start conversion has occurred. **Note:** This bit is cleared by writing 1 to it.
[15:1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**CNTMAXF**	**PWM Counter Equal to 0xFFFF Flag** 0 = Indicates the PWM counter value never reached its maximum value 0xFFFF. 1 = Indicates the PWM counter value has reached its maximum value. **Note:** This bit is cleared by writing 1 to it.

Timer PWM Period Buffer Register (TIMERx PWMPBUF)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMPBUF	TMR01_BA+0xA0	R	Timer0 PWM Period Buffer Register	0x0000_0000
TIMER1_PWMPBUF	TMR01_BA+0x1A0	R	Timer1 PWM Period Buffer Register	0x0000_0000
TIMER2_PWMPBUF	TMR23_BA+0xA0	R	Timer2 PWM Period Buffer Register	0x0000_0000
TIMER3_PWMPBUF	TMR23_BA+0x1A0	R	Timer3 PWM Period Buffer Register	0x0000_0000

31	30	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. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**PBUF**	**PWM Period Buffer Register (Read Only)** Used as PERIOD active register.

Timer PWM Comparator Buffer Register (TIMERx PWMCMPBUF)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMCMPBUF	TMR01_BA+0xA4	R	Timer0 PWM Comparator Buffer Register	0x0000_0000
TIMER1_PWMCMPBUF	TMR01_BA+0x1A4	R	Timer1 PWM Comparator Buffer Register	0x0000_0000
TIMER2_PWMCMPBUF	TMR23_BA+0xA4	R	Timer2 PWM Comparator Buffer Register	0x0000_0000
TIMER3_PWMCMPBUF	TMR23_BA+0x1A4	R	Timer3 PWM Comparator Buffer Register	0x0000_0000

31	30	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. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**CMPBUF**	**PWM Comparator Buffer Register (Read Only)** Used as CMP active register.

6.8 PWM Generator and Capture Timer (PWM)

6.8.1 Overview

The NPCA121 series provides one PWM generators – PWM0. It 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 using comparator compared with counter to generate events. These events use to generate PWM pulse, interrupt and trigger signal for EADC to start conversion.

The PWM generator supports two standard PWM output modes: Independent mode and Complementary mode, they have difference architecture. There are two output functions based on standard output modes: Group function and Synchronous function. Group function can be enabled under Independent mode or complementary mode. Synchronous function only enabled under complementary mode. Complementary mode has two comparators to generate various PWM pulse with 12-bit dead-time generator and another free trigger comparator to generate trigger signal for EADC. For PWM output control unit, it supports polarity output, independent pin mask and brake functions.

The PWM generator also supports input capture function. It supports latch PWM counter value to corresponding register when input channel has a rising transition, falling transition or both transition is happened. Capture function also support PDMA to transfer captured data to memory.

6.8.2 Features

6.8.2.1 PWM function features

- Clock source supports maximum clock frequency up to 200 MHz
- Supports up to 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
  - Synchronous function for phase control
  - 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 one-shot or auto-reload counter operation mode
- Supports group function
- Supports synchronous function
- Supports mask function and tri-state enable for each PWM pin
- Supports brake function
  - Brake source from pin, system safety events (clock failed, SRAM parity error, 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 EADC on the following events:
  - PWM counter match zero, period value or compared value
  - PWM counter match free trigger comparator compared value (only for EADC)

6.8.2.2 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
- Supports PDMA transfer function for PWM all channels

6.8.3 Block Diagram

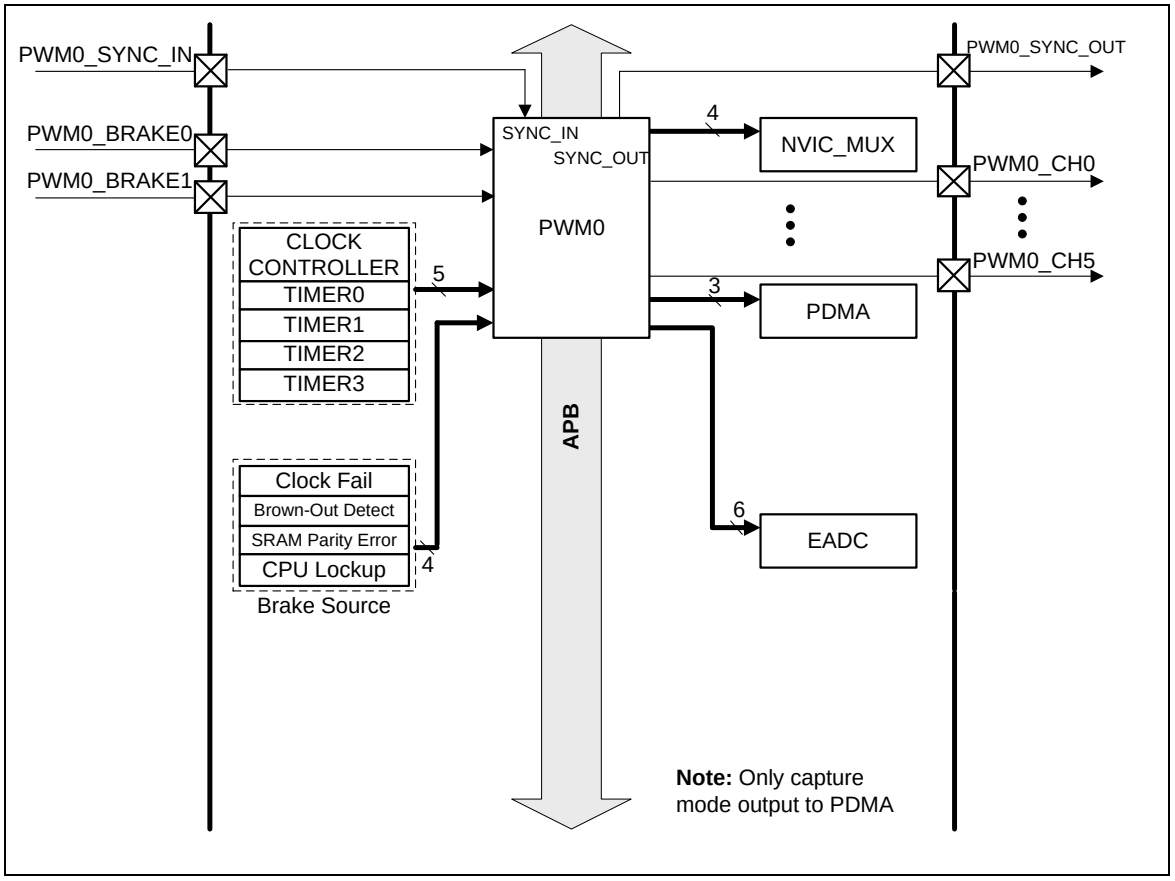


Figure 6.8-1 PWM Generator Overview Block Diagram

PWM system clock frequency can be set equal or double to HCLK frequency as the Figure 6.8-2, the detail register setting, please refer to Table 6.8.3-1.

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 6.8-3 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.

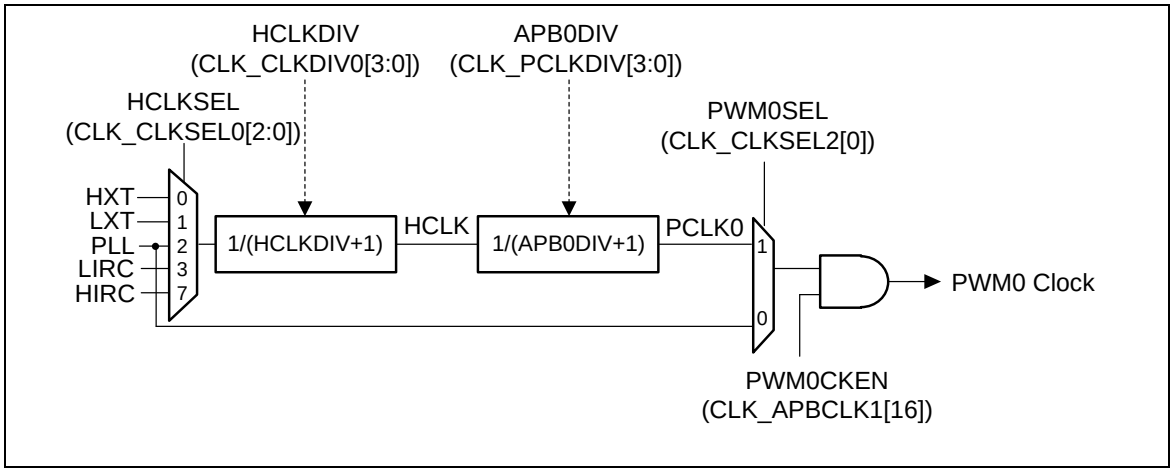


Figure 6.8-2 PWM System Clock Source Control

Frequency Ratio PCLK:PWM Clock	HCLK	PCLK	PWM Clock	HCLKSEL CLK_CLKSEL0[2:0]	HCLKDIV CLK_CLKDIV0[3:0]	APBnDIV (CLK_CLKDIVn[2+4n:4n]), N Denotes 0 Or 1	PWMnSEL (CLK_CLKSEL2[N]), N Denotes 0 Or 1
1:1	HCLK	PCLK	PCLK	Don't care	Don't care	Don't care	1
1:2	PLL	PLL/ 2	PLL	2	0	1	0
1:2	PLL/ 2	PLL/ 2	PLL	2	1	0	0

Table 6.8.3-1 PWM System Clock Source Control Registers Setting Table

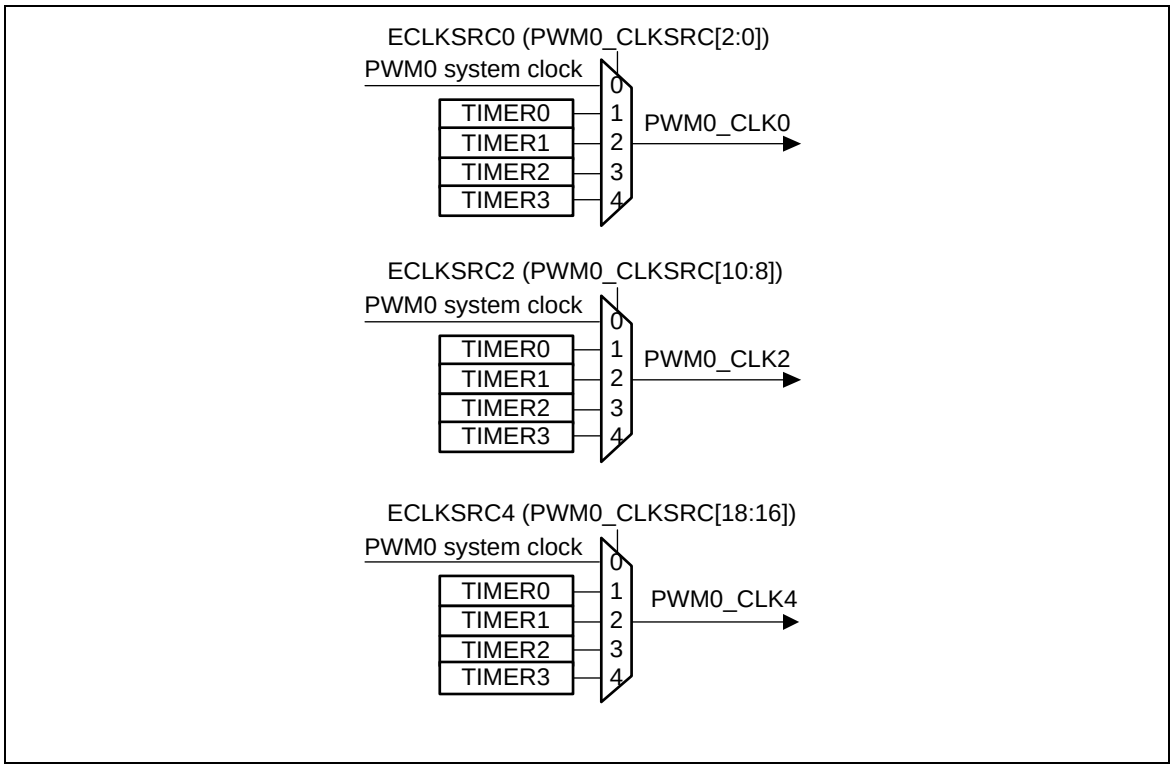


Figure 6.8-3 PWM Clock Source Control

Figure 6.8-4 and Figure 6.8-5 illustrate the architecture of PWM independent mode and complementary mode. No matter independent mode or complementary mode, paired channels' (PWM_CH0 and PWM_CH1, PWM_CH2 and PWM_CH3, PWM_CH4 and PWM_CH5) counters both come from the same clock source and prescaler. When counter count to 0, PERIOD (PWM_PERIODn[15:0]) or equal to comparator, events will be generated. These events are passed to corresponding generators to generate PWM pulse, interrupt signal and trigger signal for EADC 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, synchronize function is available and even channel use odd channel comparator to generate events, free trigger comparator events only use to generate trigger EADC signals.

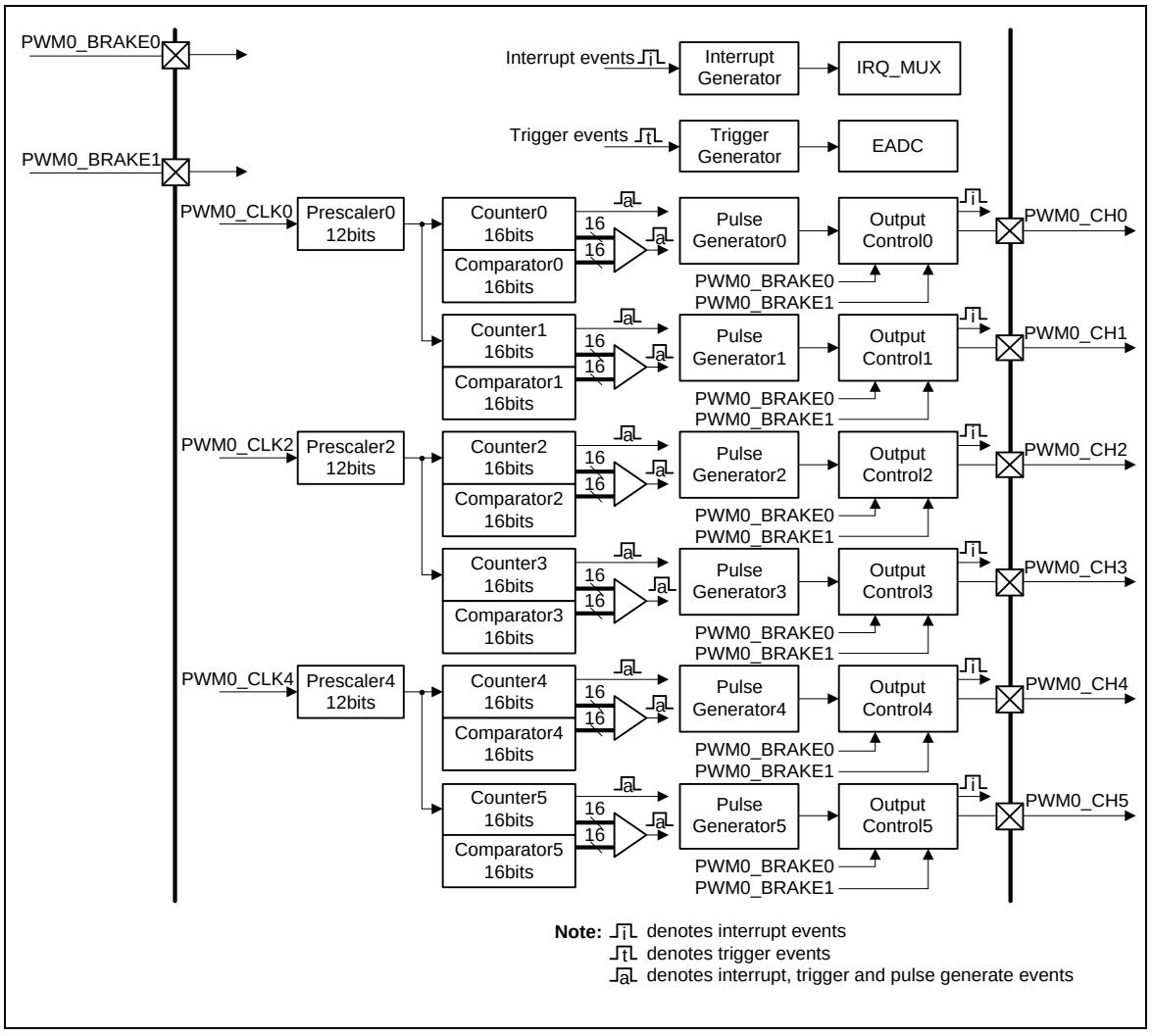


Figure 6.8-4 PWM Independent Mode Architecture Diagram

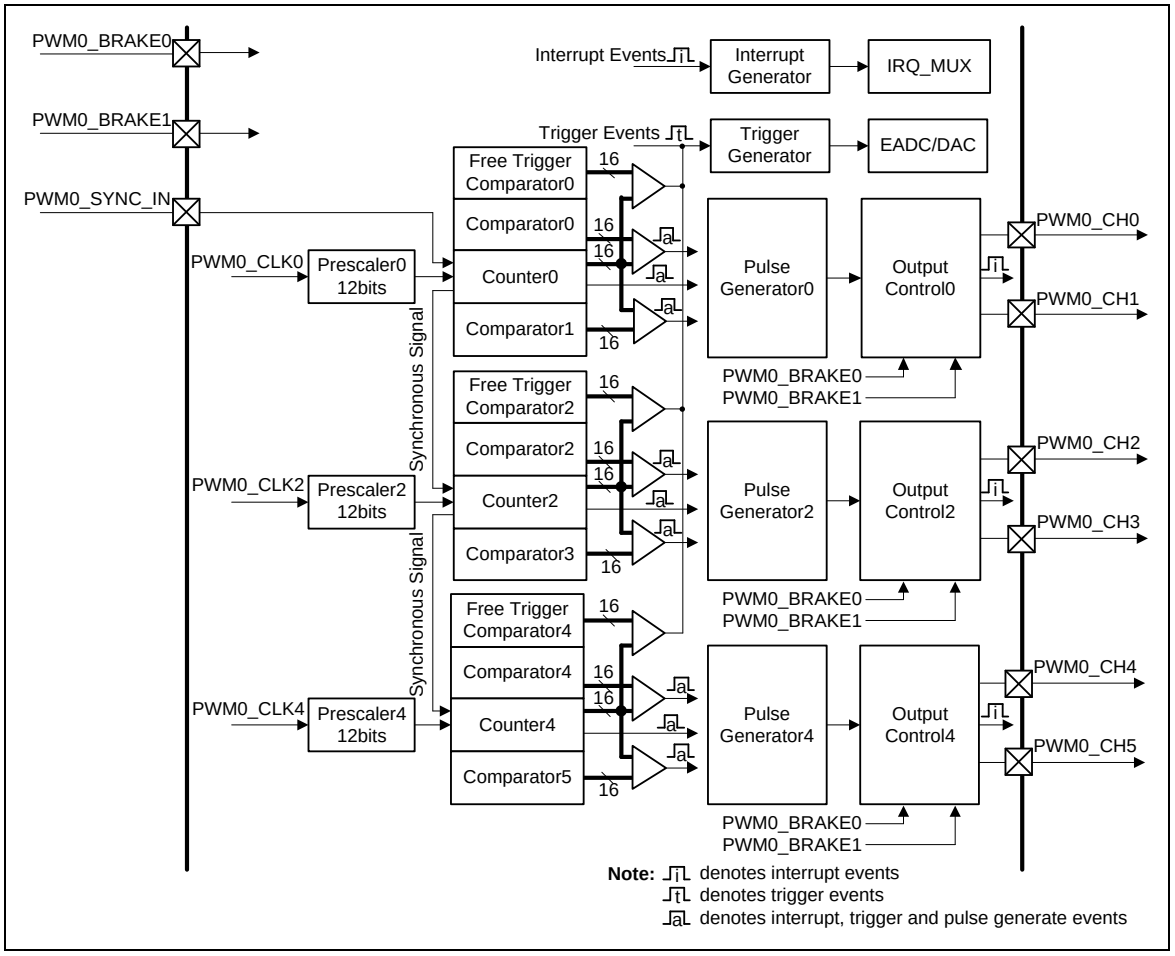


Figure 6.8-5 PWM Complementary Mode Architecture Diagram

6.8.4 Basic Configuration

- Clock source configuration
  - Select the source of PWM0 peripheral clock in PWM0SEL (CLK_CLKSEL2[0])
  - Enable PWM0 peripheral clock in PWM0CKEN (CLK_APBCLK1[16]).
- Reset configuration
  - Reset PWM0 in PWM0RST SYS_IPRST2[16]
- Pin configuration

Group	Pin Name	GPIO	MFP
PWM0	PWM0_BRAKE0	PC.10	MFP2
	PWM0_BRAKE1	PC.11	MFP2
	PWM0_CH0	PB.0	MFP3
		PB.2	MFP1
		PB.4	MFP2

	PWM0_CH1	PB.1	MFP3
		PB.3	MFP1
		PB.5	MFP2
	PWM0_CH2	PB.2	MFP3
		PB.6	MFP2
		PC.4	MFP1
	PWM0_CH3	PB.3	MFP5
		PB.7	MFP2
		PC.13	MFP1
		PD.12	MFP3
	PWM0_CH4	PB.4	MFP5
		PB.8	MFP2
		PC.14	MFP1
	PWM0_CH5	PB.9	MFP2
		PD.7	MFP1
	PWM0_SYNC_IN	PB.0	MFP1
	PWM0_SYNC_OUT	PB.1	MFP1

6.8.5 Functional Description

6.8.5.1 PWM Prescaler

PWM prescaler is used to divide clock source, prescaler counting CLKPSC +1 times, PWM counter only count once. The pre-scale double buffer is setting by CLKPSC (PWM_CLKPSCn[11:0], n = 0, 2, 4) bits. Figure 6.8-6 is an example of PWM channel 0 prescale waveform. The prescale counter will reload CLKPSC at the begin of the next prescale counter down-count.

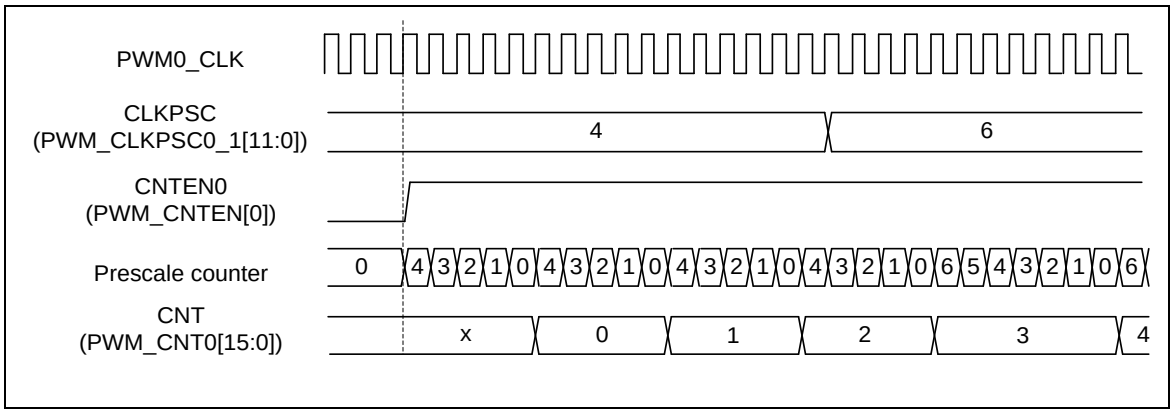


Figure 6.8-6 PWM0_CH0 Prescaler Waveform in Up Counter Type

6.8.5.2 PWM Counter

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

For PWM channel0, CNT(PWM_CNT0[15:0]) can clear to 0x00 by CNTCLR0 (PWM_CNTCLR[0]). CNT will be cleared when prescale counter to 0, and CNTCLR will be set 0 by hardware automatically.

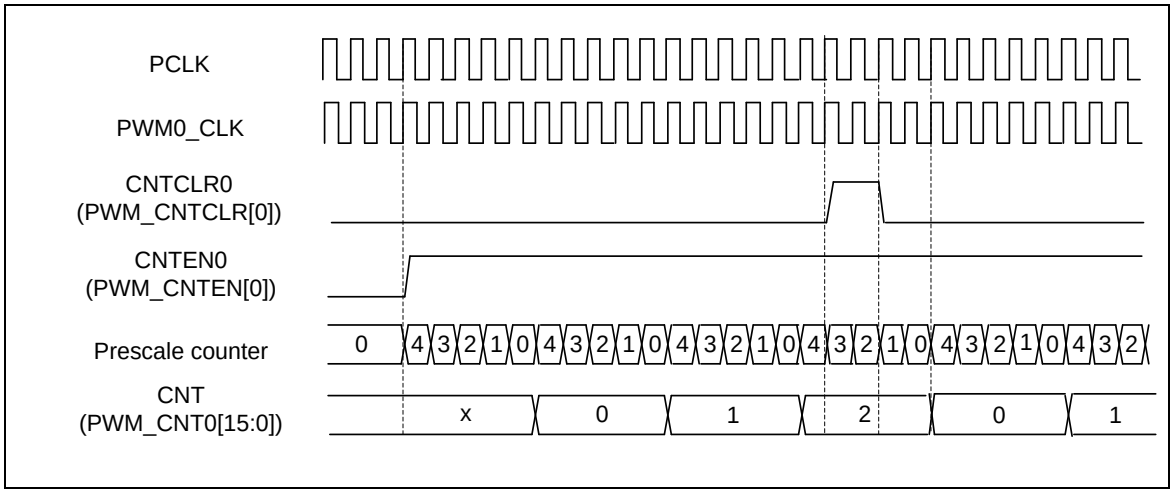


Figure 6.8-7 PWM0 Counter Waveform when set clear counter

6.8.5.3 Up Counter Type

When PWM counter is set to up counter type, CNTTYPEn (PWM_CTL1[2n+1:2n], n = 0,1..5) is 0x0, it starts up-counting from zero to PERIOD (PWM_PERIODn[15:0], where n denotes channel number) to complete a PWM period. The current counter value can be read from CNT (PWM_CNTn[15:0]) bits. PWM generates zero point event when the counter counts to 0 and prescale counts to 0. PWM generates period point event when the counter counts to PERIOD and prescale counts to 0. The Figure 6.8-8 shows an example of up counter, wherein

PWM period time = (PERIOD+1) *(CLKPSC+1)* PWM0_CLK.

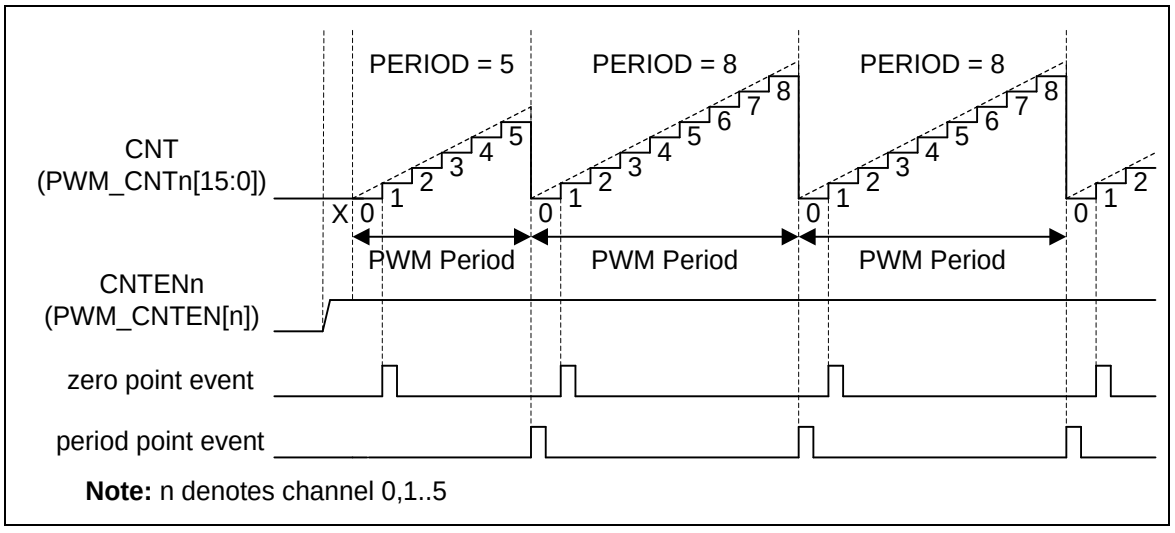


Figure 6.8-8 PWM Up Counter Type

6.8.5.4 Down Counter Type

When PWM counter is set to down counter type, CNTTYPEn (PWM_CTL1[2n+1:2n], n = 0,1..5) is 0x1, it starts down-counting from PERIOD to zero to complete a PWM period. The current counter value can be read from CNT (PWM_CNTn[15:0]) bits. PWM generates zero point event when the counter counts to 0 and prescale counts to 0. PWM generates period point event when the counter counts to PERIOD and prescale counts to 0. The Figure 6.8-9 shows an example of down counter, wherein

PWM period time = (PERIOD+1) * (CLKPSC+1) * PWM0_CLK.

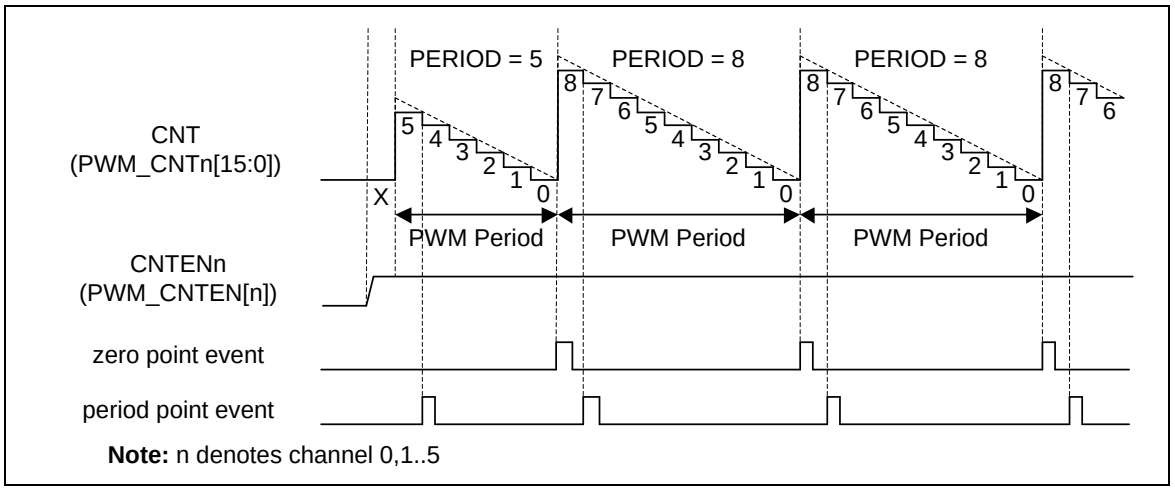


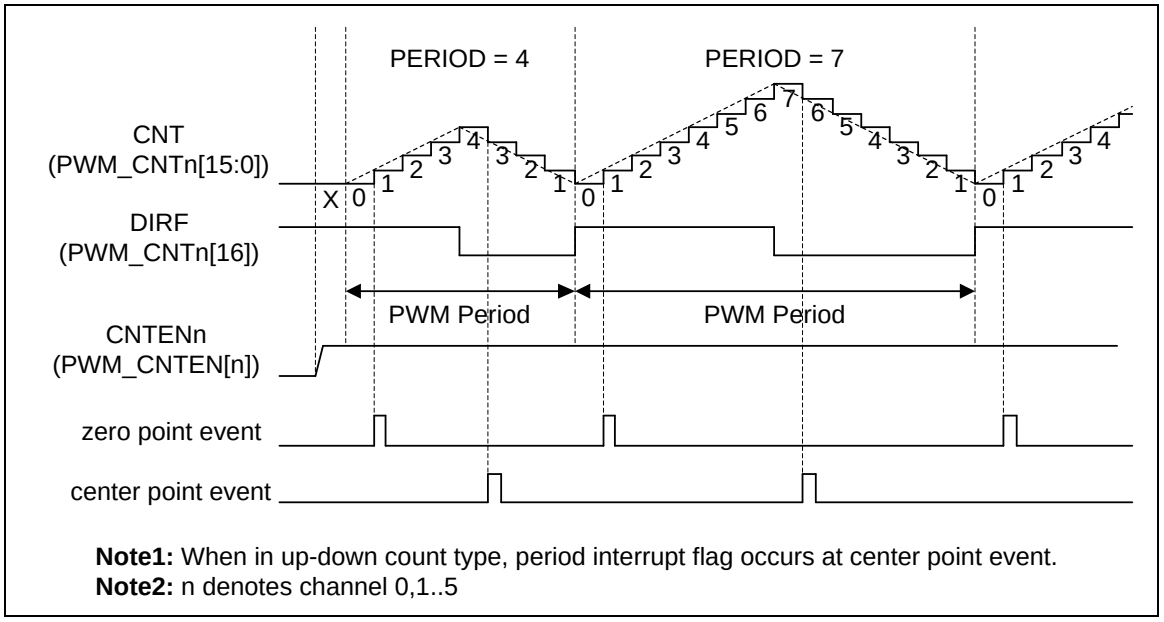
Figure 6.8-9 PWM Down Counter Type

6.8.5.5 Up-Down Counter Type

When PWM counter is set to up-down count type, CNTTYPEn (PWM_CTL1[2n+1:2n], n = 0,1..5) is 0x2, it starts counting-up from zero to PERIOD and then starts counting down to zero to complete a PWM period. The current counter value can be read from CNT (PWM_CNTn[15:0]) bits. PWM generates zero point event when the counter counts to 0 and prescale counts to 0. PWM generates center point event when the counter counts to PERIOD. The Figure 6.8-10 shows an example of up-down counter, wherein

PWM period time =(PERIOD) * (CLKPSC+1) * PWM0_CLK.

The DIRF (PWM_CNTn[16]) bit is counter direction indicator flag, where high is up counting, and low is down counting.



Note1: When in up-down count type, period interrupt flag occurs at center point event.
Note2: n denotes channel 0,1..5

Figure 6.8-10 PWM Up-Down Counter Type

6.8.5.6 PWM Comparator

There are two kinds of comparator registers : one is CMPDATn(n = 0,1..5), and the other is FTCMPDATn_m(n = 0,2,4, m = 1,3,5) register. CMPDATn is a basic comparator register of PWM channel n; In Independent mode each channel only has one comparator, the value of CMPDATn register is continuously compared to the corresponding channel's counter value. In Complementary mode each paired channels has two comparators, and the value of CMPDATn and CMPDATm (n = 0,2,4, m = 1,3,5) registers are continuously compared to the complementary even channel's counter value, because of odd channel's counter is useless. For example, channel 0 and channel 1 are complementary channels, in Complementary mode, channel 1's comparator is continuously compared to channel 0's counter, but not channel 1's. When the counter is equal to value of CMPDAT0 register, PWM generates a compared point event and uses the event to generate PWM pulse, interrupt or use to trigger EADC. In up-down counter type, two events will be generated in a PWM period as shown in Figure 6.8-11. The CMPU is up count compared point event and CMPD is down count compared point event.

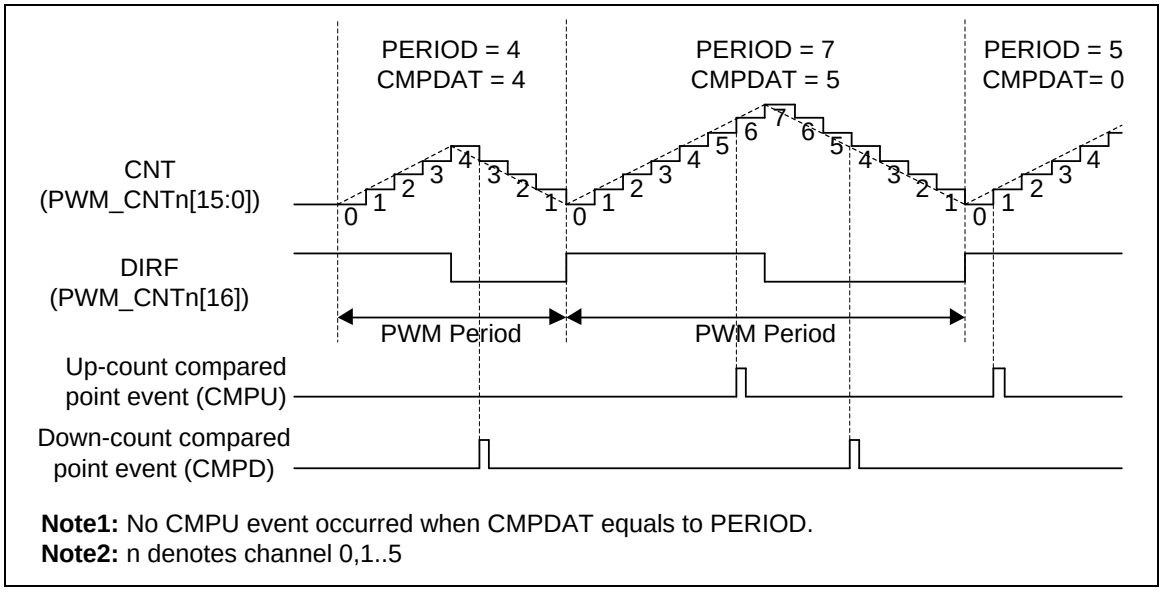


Figure 6.8-11 PWM Compared point Events in Up-Down Counter Type

FTCMPDAT is a free trigger comparator register. Each complementary paired channel only supports one free trigger comparator. The value of FTCMPDATn_m(n = 0,2,4, m = 1,3,5) register is continuously compared to even channel's counter value. When counter is equal to the value of FTCMPDAT register, PWM generates an event and only uses to trigger EADC.

6.8.5.7 PWM Double Buffering

The double buffering uses double buffers to separate software writing and hardware action operation timing. There are four loading modes for loading values to buffer: period loading mode, immediately loading mode, window loading mode and center loading mode. 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 asynchronous operation problem due to software and hardware asynchronicity.

The PWM provides PBUF (PWM_PBUFn[15:0]) as the active PERIOD buffer register, CMPBUF (PWM_CMPBUFn[15:0]) as the active CMPDAT buffer register, FTCMPBUF (PWM_FTCMPBUFn_m[15:0]) as the active FTCMPDAT buffer register and CPSCBUF (PWM_CPSCBUFn_m[15:0]) as the active CLKPSC buffer register. The concept of double buffering is used in loading modes, which are described in the following sections. For example, as shown Figure 6.8-12, in period loading mode, writing PERIOD, CMPDAT and FTCMPDAT registers through software, PWM will load new values to their buffer PBUF (PWM_PBUFn[15:0]), CMPBUF (PWM_CMPBUFn[15:0]) and FTCMPBUF (PWM_FTCBUF[15:0]) at start of the next period without affecting the current period counter operation. FTCMPU denotes up-count free trigger compared point event and FTCMPD denotes down-count free trigger compared event.

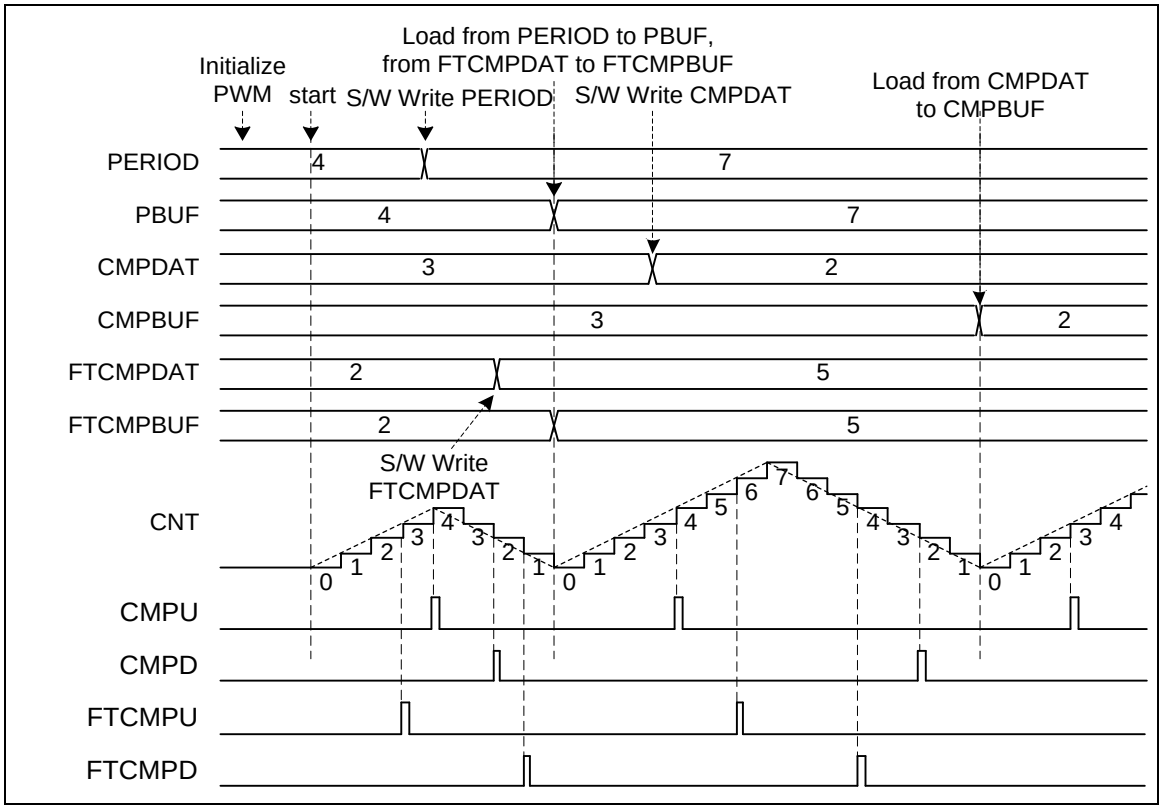


Figure 6.8-12 PWM Double Buffering Illustration

6.8.5.8 Period Loading Mode

When immediately loading mode, window loading mode and center loading mode are disabled that IMMLDENn bits, WINLDENn bits and CTRLDn bits of PWM_CTL0 register are set to 0, PWM operates at period Loading mode. In period Loading mode, CLKPSC(PWM_CLKPSCn_m[11:0]), PERIOD(PWM_PERIODn[15:0]) and CMP(PWM_CMPDATn[15:0]) will all load to their active CPSCBUF, PBUF and CMPBUF registers while each 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 the up-down counter operation.

Figure 6.8-13 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 6.8-13. 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.

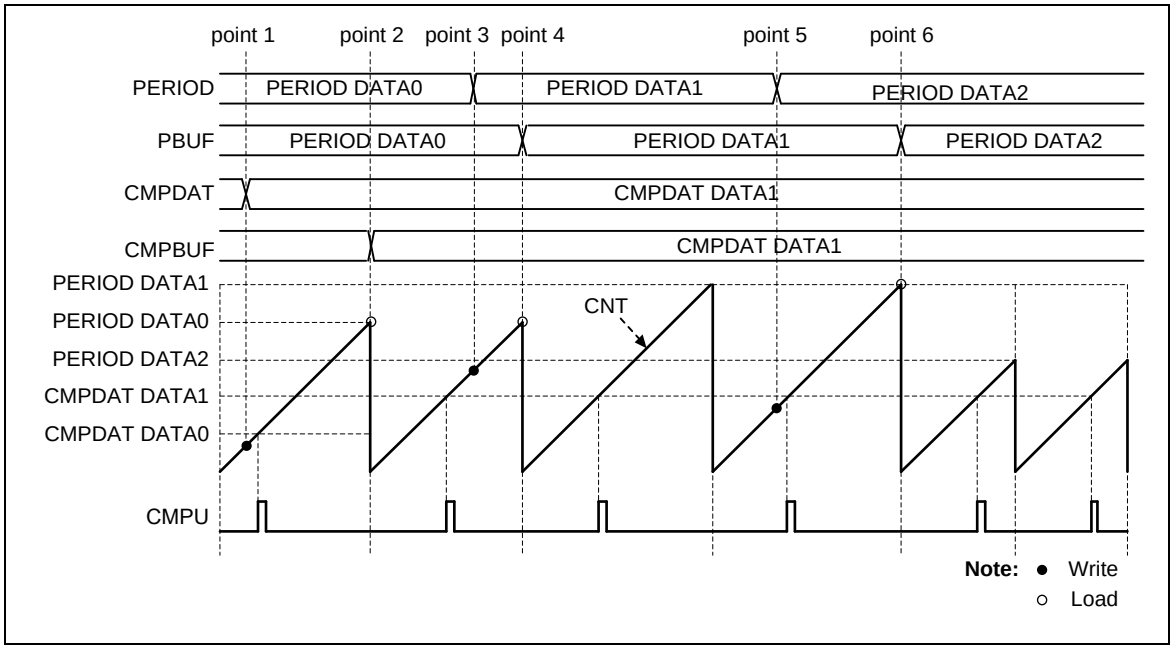


Figure 6.8-13 Period Loading in Up-Count Mode

6.8.5.9 Immediately Loading Mode

If the IMMLDENn (PWM_CTL0[21:16]) bit is set to 1, PWM operates at immediately loading mode. In immediately loading mode, when user update CLKPSC(PWM_CLKPSCn_m[11:0]), PERIOD(PWM_PERIODn[15:0]) or CMP(PWM_CMPDATn[15:0]), PERIOD or CMPDAT will be load to active CPSCBUF (PWM_CPSCBUFn_m[15:0]), PBUF (PWM_PBUFn[15:0]) or CMPBUF (PWM_CMPBUFn[15:0]) after current counter count is completed. If the updated PERIOD value is less than current counter value, counter will count to 0xFFFF, when counter count to 0xFFFF and prescale count to zero, the flag CNTMAXF(PWM0_STATUS[5:0]) will raise, and then 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 6.8-14 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.

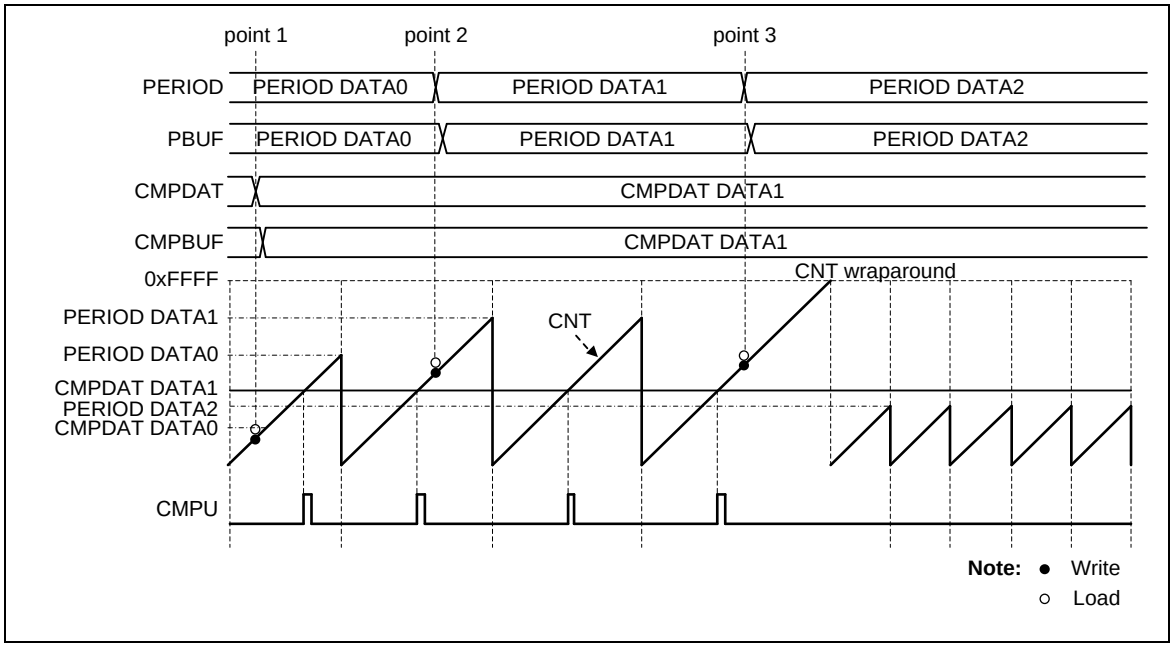


Figure 6.8-14 Immediately Loading in Up-Count Mode

6.8.5.10 Window Loading Mode

When the WINLDENn (PWM_CTL0[13:8]) bit is set to 1, PWM operates at window loading mode. In Window loading mode, CLKPSC(PWM_CLKPSCn_m[11:0]), PERIOD(PWM_PERIODn[15:0]) and CMP(PWM_CMPDATn[15:0]) will all load to their active CPSCBUF, PBUF and CMPBUF registers while each period is completed, but all loading are valid only when load window is opened. Every channel n's load window is opened by setting the corresponding LOADn (PWM_LOAD[5:0]) to 1, and hardware will close the window at the end of PWM period. Figure 6.8-15 shows an example and its steps sequence is described below.

1. Software writes CMPDAT DATA1 at point 1, and the load window is not opened at this period so CMPDAT will not load to CMPBUF.
2. Software writes LOAD to open the load window at point2.
3. Software writes PERIOD DATA1 at point 3.
4. At point 4, load window has been opened, hardware loads CLKPSC DATA1, PERIOD DATA1 and CMPDAT DATA1 to their buffer and closes the load window at the end of PWM period.
5. Software writes PERIOD DATA2 at point 5.
6. Hardware loads CLKPSC DATA2, PERIOD DATA2 to PBUF at the end of PWM period at point 6.
7. Software writes PERIOD DATA3 at point 7.
8. Software writes LOAD to open the load window at point8.
9. Hardware loads CLKPSC DATA3 and PERIOD DATA3 to their buffer and closes the load window at the end of PWM period at point 9.

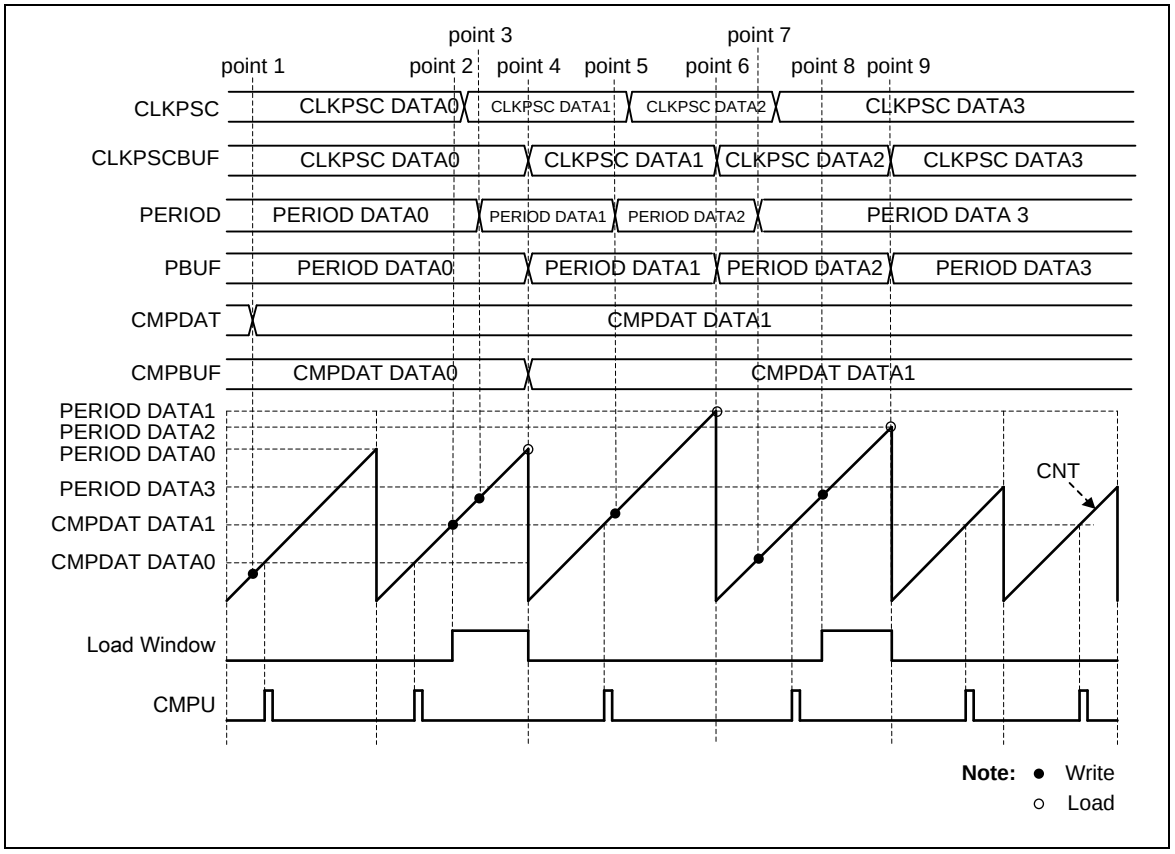


Figure 6.8-15 Window Loading in Up-Count Mode

6.8.5.11 Center Loading Mode

When the CTRLDn (PWM_CTL0[5:0]) bit is set to 1 and PWM counter is set to up-down count type, CNTTYPEn (PWM_CTL1[2n+1:2n], n = 0,1..5) is 0x2, PWM operates at center loading mode. In center loading mode, CMP(PWM_CMPDATn[15:0]) will load to active CMPBUF register in center of each period, that is, counter counts to PERIOD. CLKPSC(PWM_CLKPSCn_m[11:0]) and PERIOD(PWM_PERIODn[15:0]) will all load to their active CPSCBUF and PBUF registers while each period is completed Center loading mode can work with window loading mode, the CMP(PWM_CMPDATn[15:0]) will load to active CMPBUF register in center of each period , but it is valid only at the interval of load window. Figure 6.8-16 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.

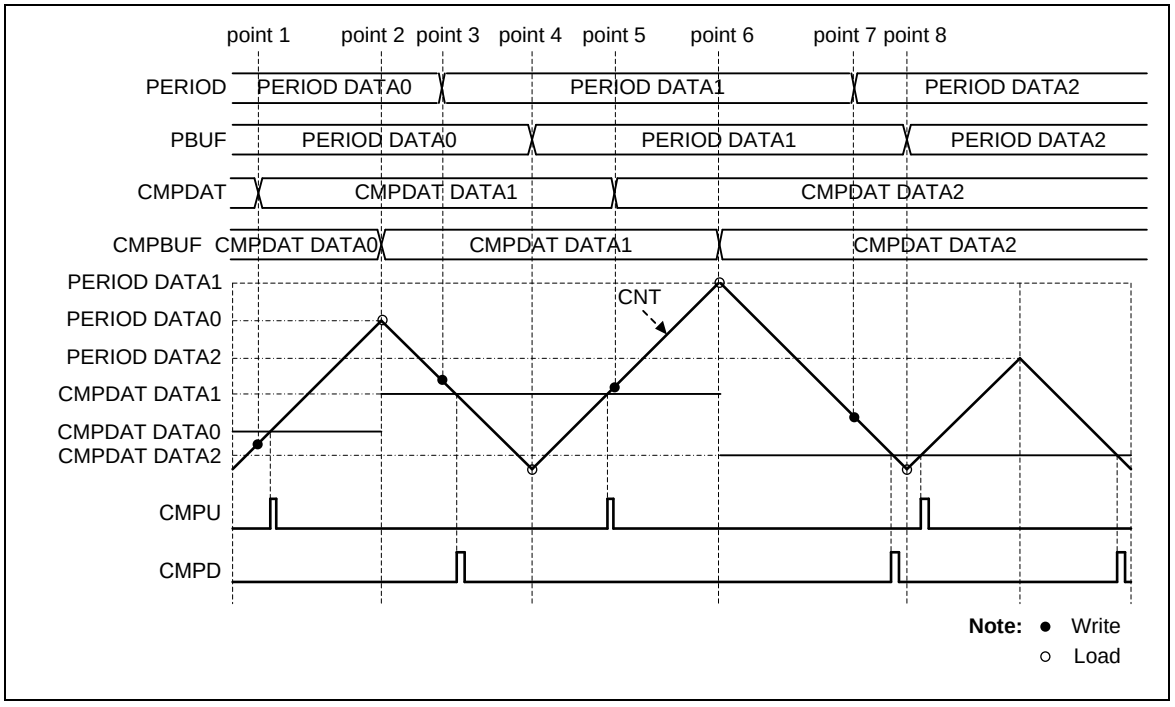


Figure 6.8-16 Center Loading in Up-Down-Count Mode

6.8.5.12 PWM Counter Operation mode

The PWM counter supports two operation modes: One-shot mode and Auto-reload mode. PWM counter will operate in One-shot mode if CNTMODEn (PWM_CTL1[21:16]) bit is set to 1, and operate in Auto-reload mode if set to 0.

In One-shot mode, CMPDAT and PERIOD registers should be written first and then set CNTENn (PWM_CNTEN[5:0]) bit as 1 to enable PWM prescaler and counter start running. After PWM counter counted a period, counter value will keep in zero.

User can re-start next one-shot by writing new value to CMP(PWM_CMPDATn[15:0]) bits. If one-shot counter still running, to update CMPDAT register will cause next one-shot as continuous one-shot. Besides, to write CMPDAT register twice under continuous one-shot operation, latest value in CMPDAT register is valid at next one-shot period and only generate one-shot pulse once. Figure 6.8-17 is an example and following is steps sequence.

1. Software writes PERIOD DATA1 and hardware immediately loading PERIOD DATA1 to PBUF at point 1.
2. Software writes CMPDAT DATA1 which is equal to CMPDAT DATA0 at point 2 and hardware immediately loading CMPDAT DATA1 to CMPBUF, this event also trigger one-shot.
3. Software writes CMPDAT DATA2 and re-trigger next one-shot (continuous one-shot) at point 3.
4. Software writes CMPDAT DATA3 to cover CMPDAT DATA2 and re-trigger next one-shot at point 4.
5. Period loading CMPDAT DATA3 to CMPBUF at point 5.
6. There are no new CMPDAT write in the previous period, and the counter value is kept as zero at point 6.

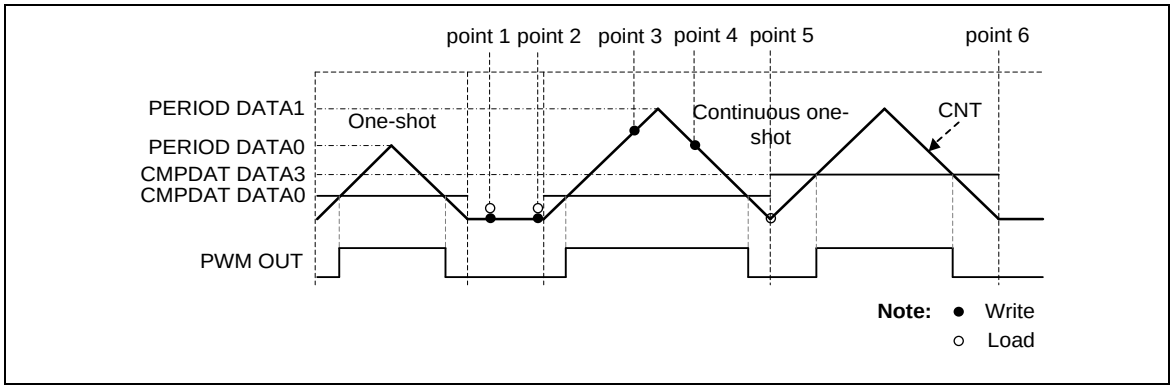


Figure 6.8-17 PWM One-shot Mode Output Waveform

In Auto-reload mode, CMPDAT and PERIOD registers should be written first and then the CNTENn(PWM_CNTEN[n]) bit is set to 1 to enable PWM prescaler and start to run counter. The value of CLKPSC(PWM_CLKPSCn_m[11:0]), PERIOD(PWM_PERIODn[15:0]) and CMP(PWM_CMPDATn[15:0]) will auto reload to their active buffer according different loading mode. If PERIOD(PWM_PERIODn[15:0]) is set to 0, PWM counter will be set to 0.

6.8.5.13 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 the other 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 the PWM_WGCTL0 and PWM_WGCTL1 registers. Using these points can easily generate asymmetric PWM pulse or variant waveform as shown in Figure 6.8-18. 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, or 4, and m denotes odd channel number 1, 3, or 5. n channel and m channel are complementary paired. Complementary mode uses two channels (CH0 and CH1, CH2 and CH3, or CH4 and CH5) as a pair of PWM outputs to generate complement paired waveforms. CMPU denotes CNT(PWM_CNTn[15:0]) is equal to CMP(PWM_CMPDATn[15:0]) when counting up. CMPD denotes CNT bits is equal to CMP bits when counting down.

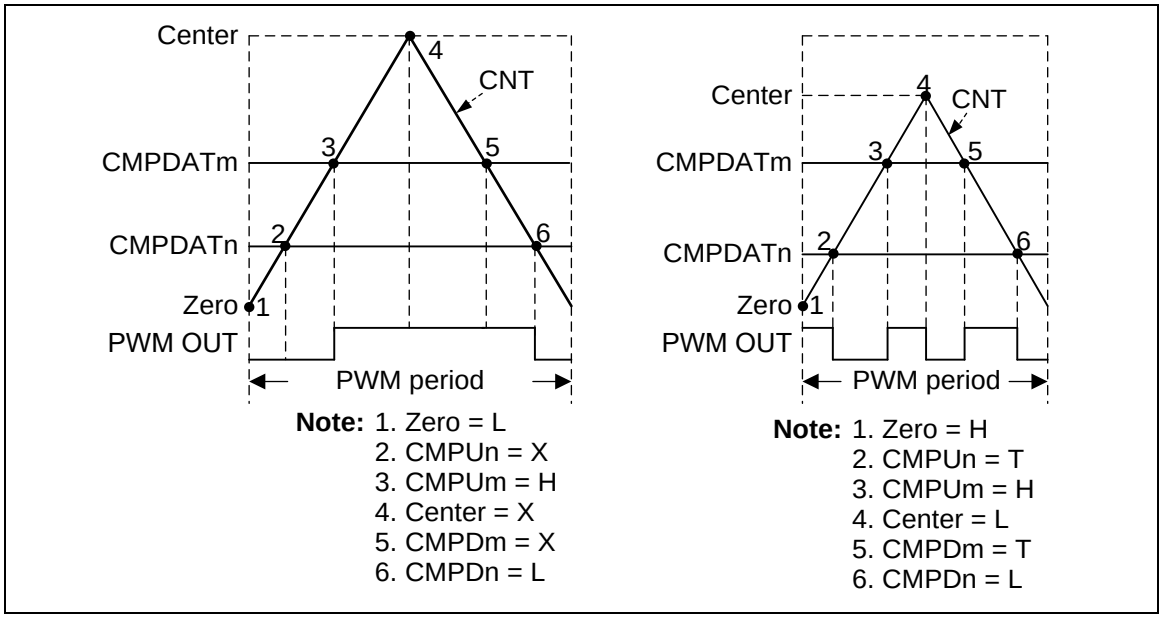


Figure 6.8-18 PWM Pulse Generation

The generation events may sometimes set to the same value, as the reason, events priority between different counter types are list below, up counter type(Table 6.8.5-1), down counter type (Table 6.8.5-2) and up-down counter type (Table 6.8.5-3). By using event priority, user can easily generate 0% to 100% duty pulse as shown in Figure 6.8-19.

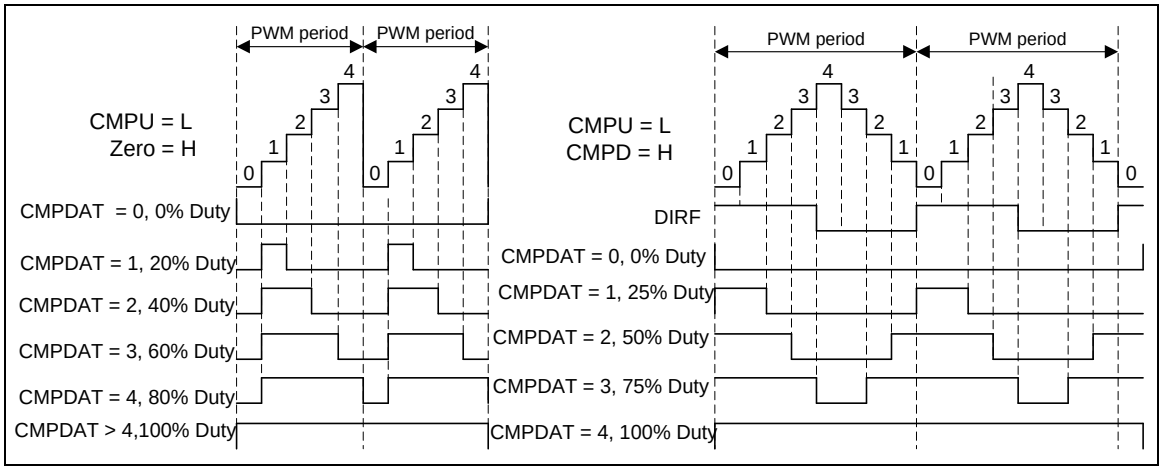


Figure 6.8-19 PWM 0% to 100% Pulse Generation

Priority	Up Event
1 (Highest)	Period event (CNT = PERIOD)
2	Compare up event of odd channel (CNT = CMPUm)
3	Compare up event of even channel (CNT = CMPUn)
4 (Lowest)	Zero event (CNT = zero)

Table 6.8.5-1 PWM Pulse Generation Event Priority for Up-Counter

Priority	Down Event
1 (Highest)	Zero event (CNT = zero)
2	Compare down event of odd channel (CNT = CMPDm)
3	Compare down event of even channel (CNT = CMPDn)
4 (Lowest)	Period event (CNT = PERIOD)

Table 6.8.5-2 PWM Pulse Generation Event Priority for Down-Counter

Priority	Up Event	Down Event
1 (Highest)	Compare up event of odd channel (CNT = CMPUm	Compare down event of odd channel (CNT = CMPDm)
2	Compare up event of even channel (CNT = CMPUn)	Compare down event of even channel (CNT = CMPDn)
3	Zero event (CNT = zero)	Period (center) event (CNT = PERIOD)
4	Compare down event of odd channel (CNT = CMPDm)	Compare up event of odd channel (CNT = CMPUm)
5 (Lowest)	Compare down event of even channel (CNT = CMPDn)	Compare up event of even channel (CNT = CMPUn)

Table 6.8.5-3 PWM Pulse Generation Event Priority for Up-Down-Counter

6.8.5.14 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.

6.8.5.15 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 is 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 6.8-20.

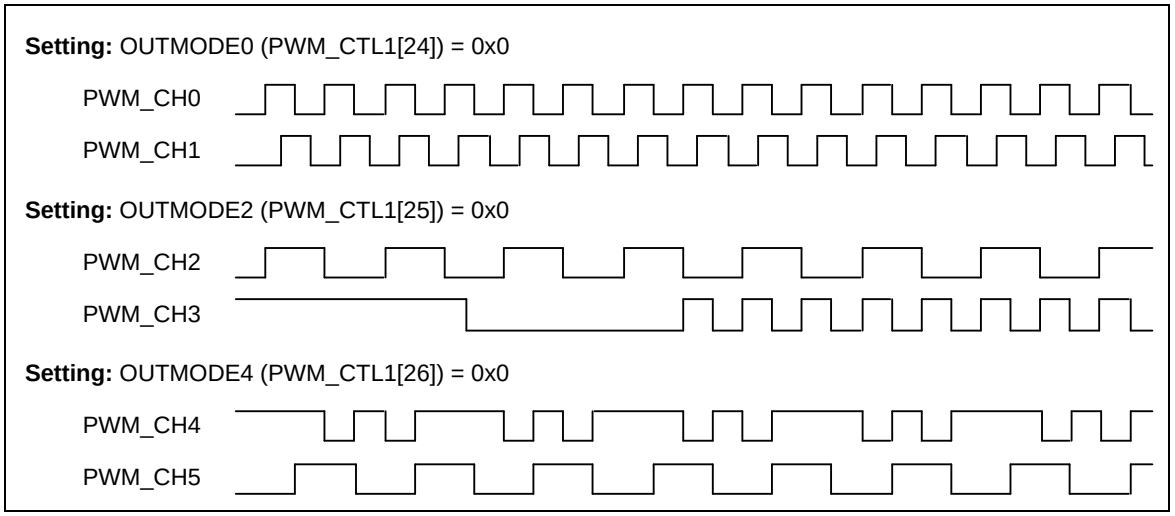


Figure 6.8-20 PWM Independent Mode Waveform

6.8.5.16 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 Complementary modes, 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 6.8-21.

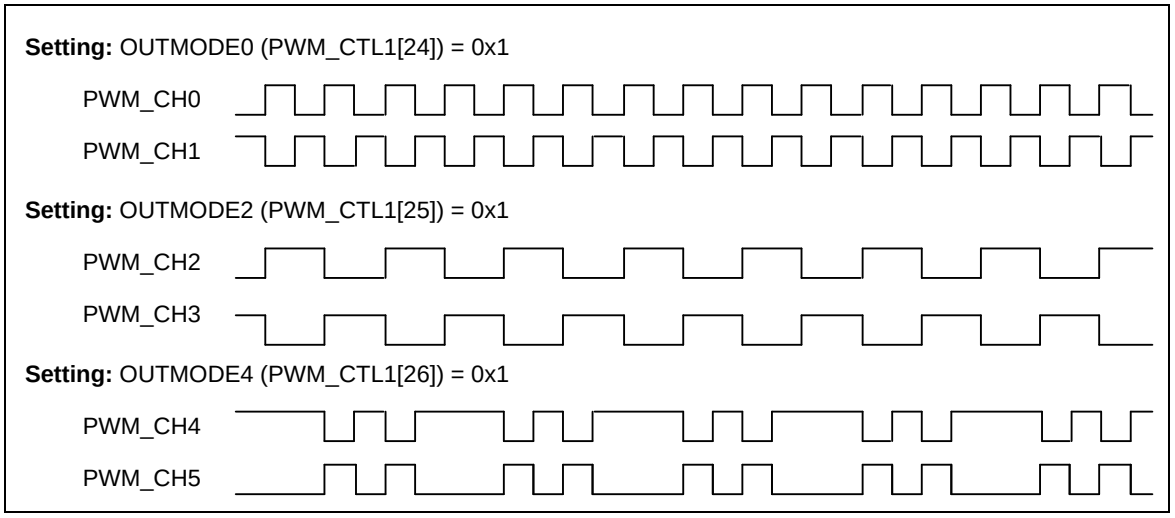


Figure 6.8-21 PWM Complementary Mode Waveform

6.8.5.17 PWM Output Function

Based on the output mode, there are two output functions: group and synchronous functions for advanced output control. Group function, forces the PWM_CH2 and PWM_CH4 synchronous with PWM_CH0 generator and forces the PWM_CH3 and PWM_CH5 synchronous with PWM_CH1, may simplify updating duty control in DC and BLDC motor applications. Besides, Synchronous

function makes any channel of PWM0 in phase, user can control phase value and direction.

6.8.5.18 Group function

Group function is enabled when GROUPEN (PWM_CTL0[24]) is set to 1, no matter in independent or complementary mode. This control allows all even PWM channels output to be controllable by PWM_PERIOD0 and PWM_CMPDAT0 registers and all odd PWM channels output to be controllable by PWM_PERIOD1 and PWM_CMPDAT1 registers. That is, user only needs to set PWM_CH0 to get PWM_CH0, PWM_CH2 and PWM_CH4 output the same pulse, and set PWM_CH1 to get PWM_CH1, PWM_CH3 and PWM_CH5 output the same pulse, as shown in Figure 6.8-22. When operating group function, OUTMODE0, OUTMODE2 and OUTMODE4 bits of PWM_CTL1 register must all set to 0 for independent mode or all set to 1 for complementary mode.

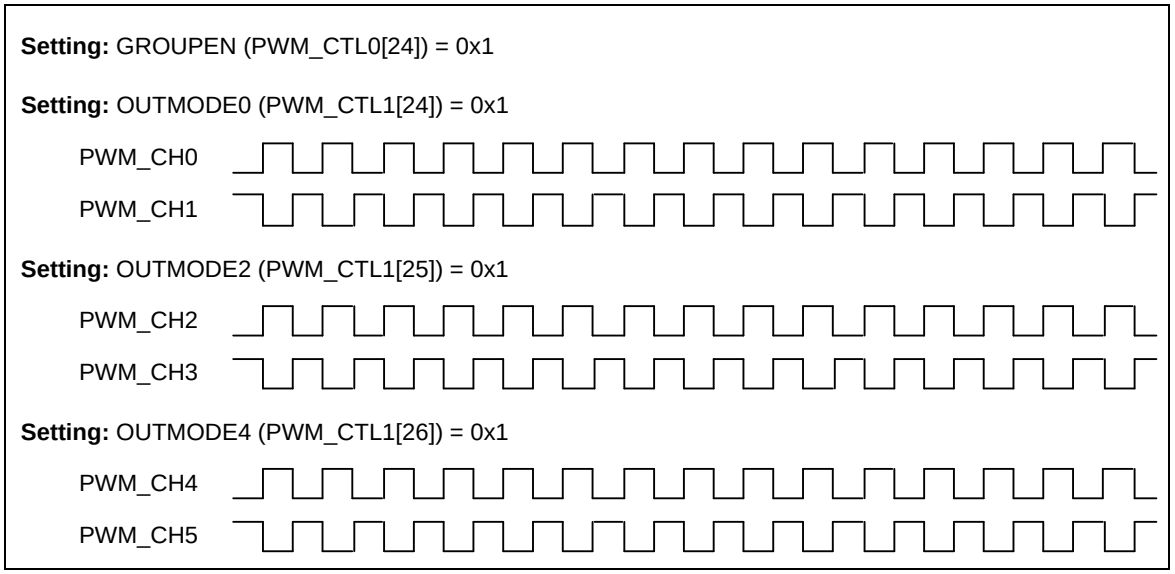


Figure 6.8-22 PWM Group Function Waveform

6.8.5.19 Synchronous function

Synchronous function can only be enabled when complementary mode is enabled. Figure 6.8-24 is counter synchronous function block diagram. Every counter of PWM pairs has a SYNC_IN and a SYNC_OUT signals. The SYNC_IN signal for the first PWM0 pair counter comes from PWM0_SYNC_IN pin, and the others come from the SYNC_OUT signal of the previous PWM pair counter. The input signal from PWM0_SYNC_IN pin will be filtered by a 3-bit noise filter as Figure 6.8-23. In addition, it can be inversed by setting the bit SINPINV (PWM_SYNC[23]) to realize the polarity setup for the input signal. The noise filter sampling clock can be selected by setting bits SFLTCSEL (PWM_SYNC[19:17]) to fit different noise properties. Moreover, by setting the bits SFLTCNT (PWM_SYNC[22:20]), user can define by how many sampling clock cycles a filter will recognize the effective edge of the SYNC_IN signal. Configuring the SNFLTEN (PWM_SYNC[16]) will enable the noise filter function. By default, it is disabled.

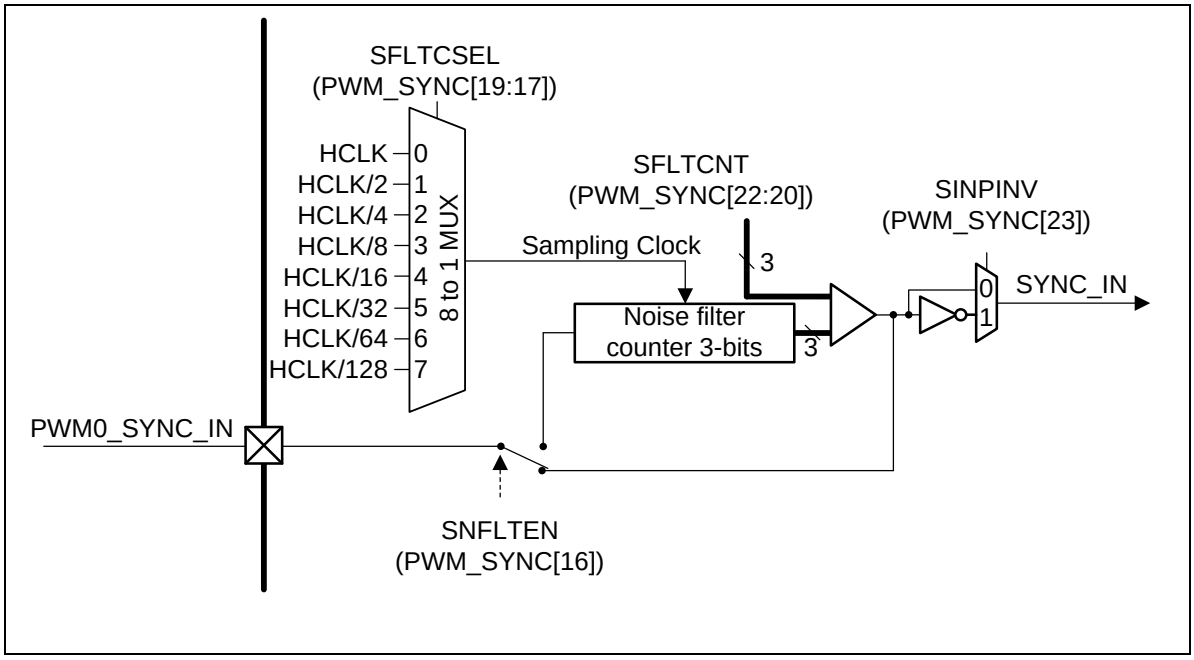


Figure 6.8-23 PWM SYNC_IN Noise Filter Block Diagram

User can use SINSRCn (PWM_SYNC[13:8]) bits to select the synchronize source. When SINSRCn bits is set to 0, user can generate SYNC_IN signal for the next counter's synchronization when PWM0_SYNC_IN pin is high or setting SWSYNCn (PWM_SWSYNC[2:0]) to 1. Synchronizing source can also be selected as CNT = 0 or CNT = PWM_CMPDATm register (if being the up-down counter type, it will synchronize twice in a PWM period) to trigger a sync event or to disable SYNC_OUT signal.

When the PHSENn (PWM_SYNC[2:0]) is enabled and the synchronous source has a happening event, the counter will load a value from the PHS (PWM_PHSn_m[15:0]) register. This method synchronizes counters to different phase in the same time. In the up-down counter type, user can set the value in PHSDIRn (PWM_SYNC[26:24]) to control the counter direction after synchronization. Although the Synchronous function can synchronize channels in phase, it can't work from the beginning of PWM enable. To start PWM counters in the same time, user have to set the PWM Synchronous Start Control Register (PWM_SSCTL[5:0]) to enable the channel counters which are planned to start counting together, followed by setting the PWM Synchronous Start Trigger Register CNTSEN (PWM_SSTRG[0]).

For applications, please do not use Group and Synchronous function simultaneously because the Synchronous function will be inactive.

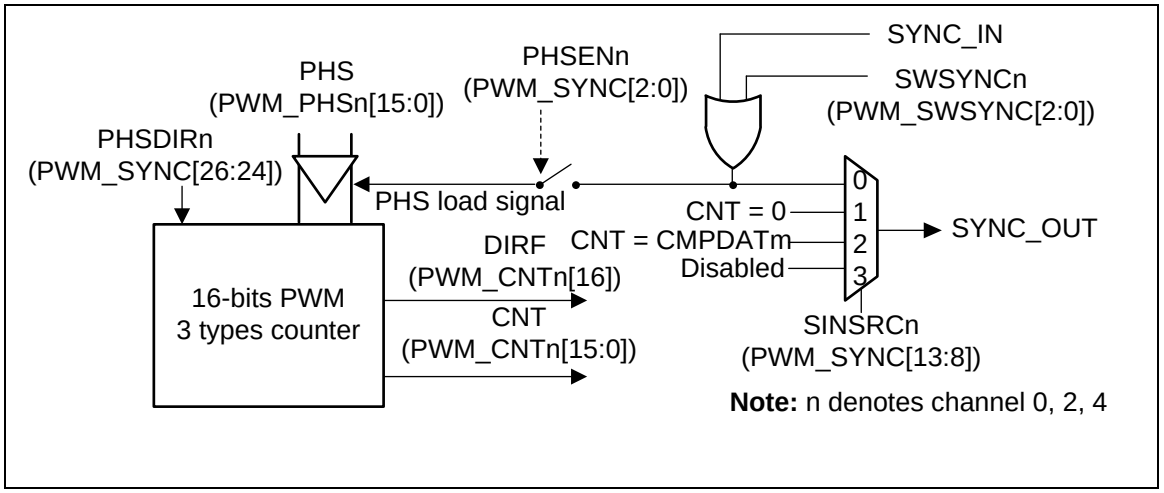


Figure 6.8-24 PWM Counter Synchronous Function Block Diagram

Figure 6.8-25 is an example of the synchronous function in the up-down counter type. In the example, synchronizing source comes from the external PWM SYNC_IN signal. At the beginning, the output waveform of PWM_CH0, PWM_CH2 and PWM_CH4 are in the same phase. Then at Point A, the PWM SYNC input signal comes as a sync event, resulting in phase shifts and counting direction changes for all of the counters. To realize the altered counter behaviors before the sync event coming, user has to setup the corresponding phase value in the PHS of(PWM_PHSn_m[15:0]) as well as the counting direction in the PHSDIRn (PWM_SYNC[26:24]). In this case, one third of phase shifts are made. by setting the corresponding channel n's counter counting direction after synchronizing, as illustrated around the left side of Figure 6.8-25.

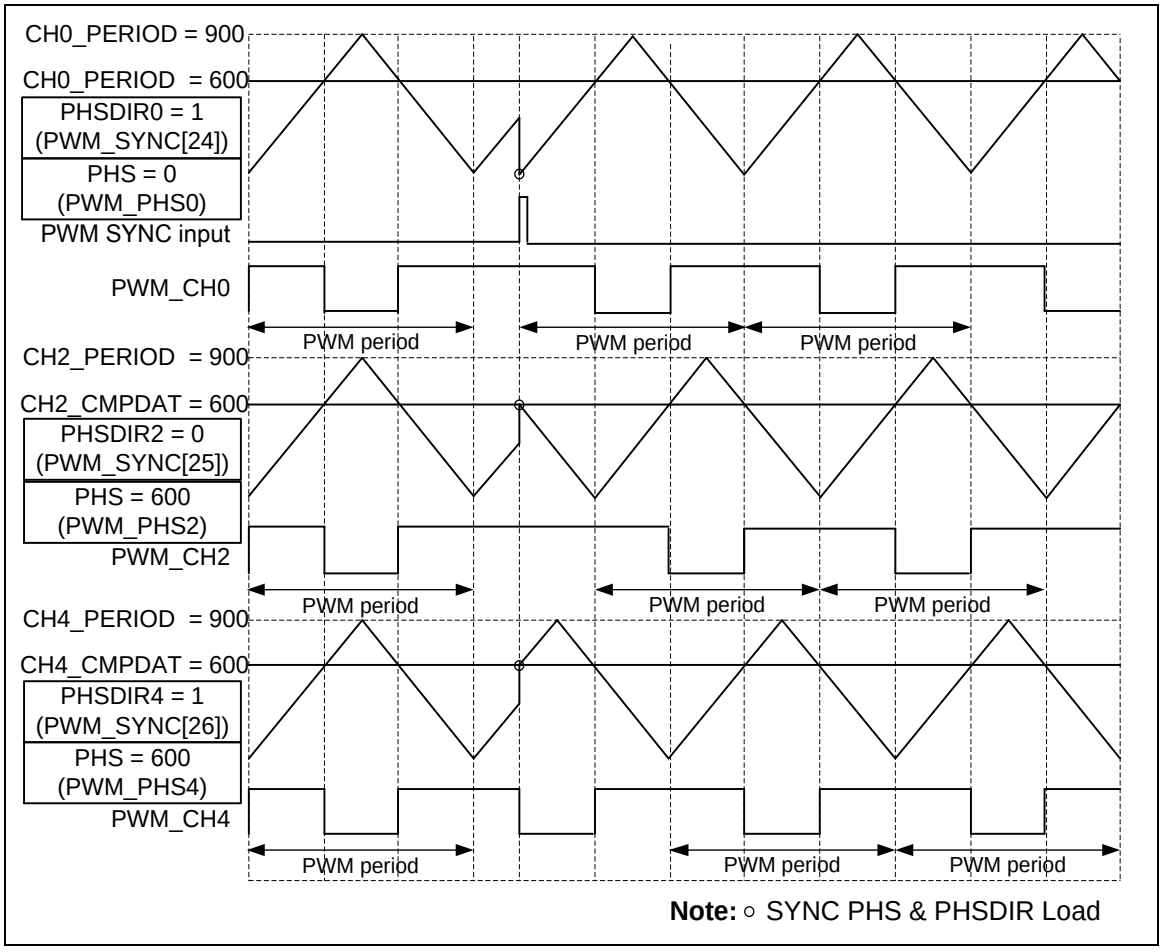


Figure 6.8-25 PWM Synchronous Function with Synchronize source from SYNC_IN Signal

6.8.5.20 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 6.8-26. In complementary mode, it needs two more steps to precede these four steps, Complementary channels and Dead-Time Insertion as shown in Figure 6.8-27.

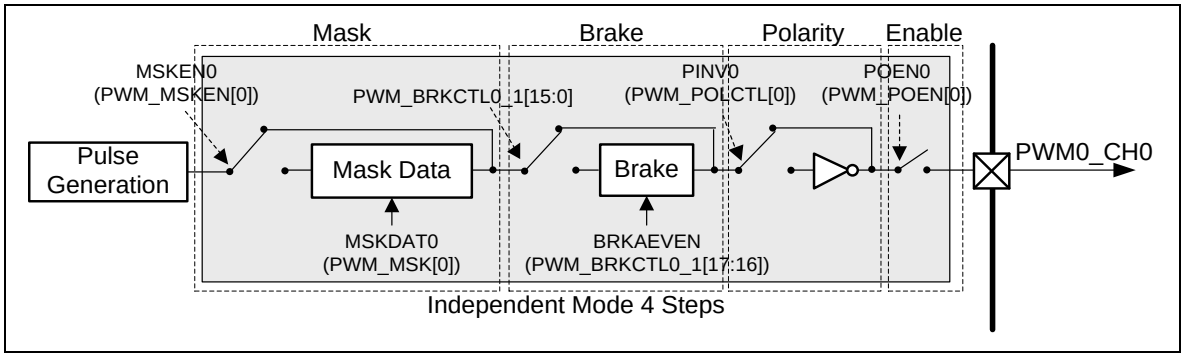


Figure 6.8-26 PWM0_CH0 Output Control in Independent Mode

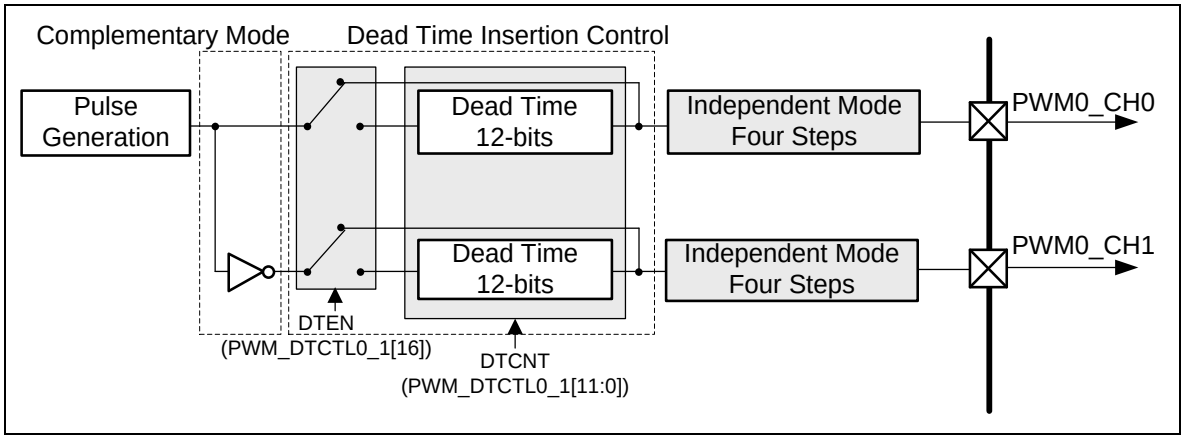


Figure 6.8-27 PWM0_CH0 and PWM0_CH1 Output Control in Complementary Mode

6.8.5.21 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 mechanism to the proper operation of the complementary system. By setting corresponding channel n DTEN (PWM_DTCTLn_m[16]) bit to enable dead-time function and DTCNT (PWM_DTCTLn_m[11:0]) to control dead-time period, the dead-time can be calculated from the following formula:

Dead-time = (DTCNT (PWM_DTCTLn[11:0])+1) * PWM0_CLK period

Dead-time insertion clock source can be selected from prescaler output by setting DTCKSEL (PWM_DTCTLn_m[24]) to 1. By default, clock source is come from PWM_CLK, which is prescaler input. Then the dead-time can be calculated from the following formula:

Dead-time = (DTCNT (PWM_DTCTLn[11:0])+1) *

(CLKPSC (PWM_CLKPSCn [11:0])+1)*PWM0_CLK period

Please note that the PWM_DTCTLn_m are write-protected registers.

Figure 6.8-28 indicates the dead-time insertion for one pair of PWM signals.

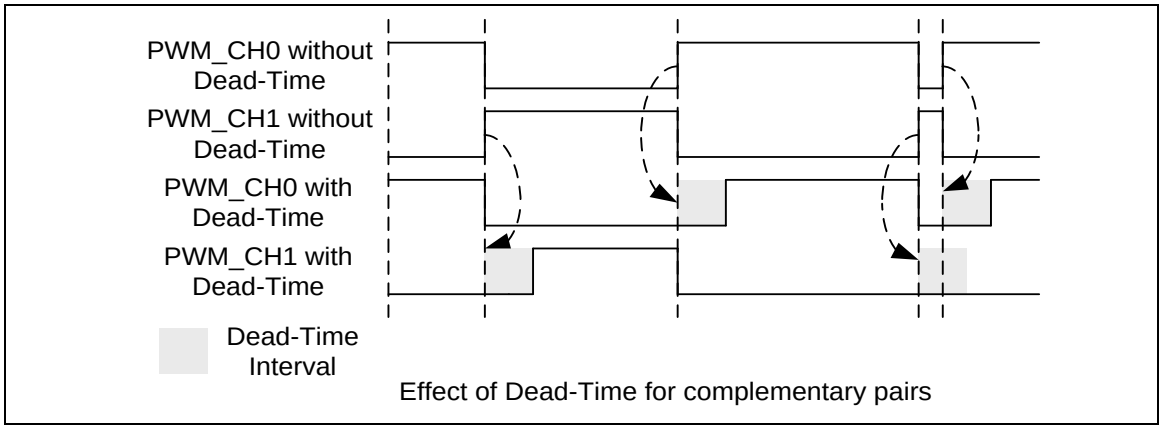


Figure 6.8-28 Dead-Time Insertion

6.8.5.22 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 6.8-29 shows an example of how PWM mask control can be used for the override feature.

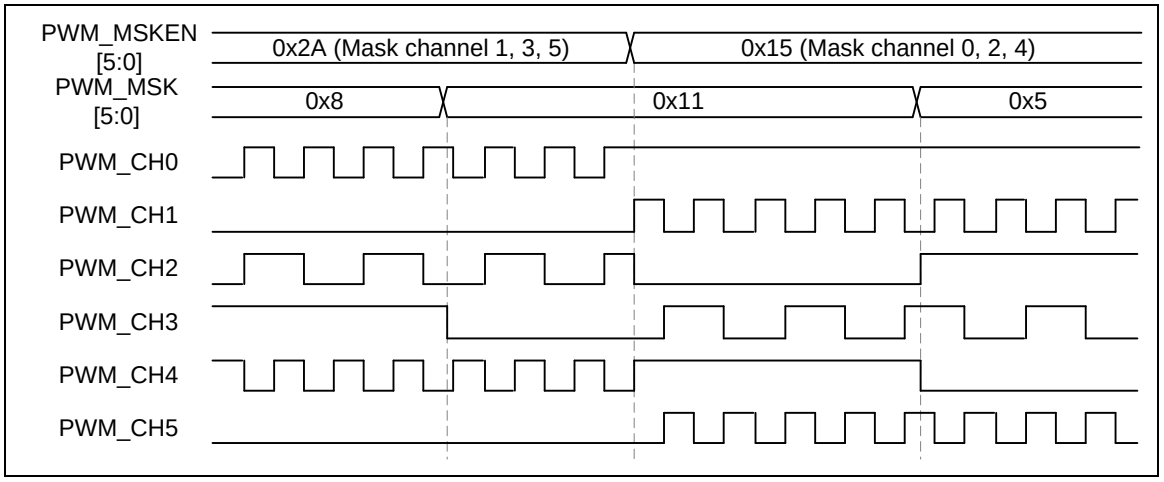


Figure 6.8-29 Illustration of Mask Control Waveform

6.8.5.23 PWM Brake

PWM module has two external input brake control signals. The external signals will be filtered by a 3-bit noise filter. User can enable the noise filter function by BRKxNFEN bits of PWM_BNF register, and noise filter sampling clock can be selected by setting BRKxNFSEL bits of PWM_BNF register to fit different noise properties. Moreover, by setting the BRKxFCNT bits, user can define by how many sampling clock cycles a filter will recognize the effective edge of the

brake signal.

In addition, it can be inversed by setting the BRKxPINV (x denotes input external pin 0 or 1) bits of PWM_BNF register to realize the polarity setup for the brake control signals. Set BRKxPINV bit to 0, brake event will occurred when PWM0_BRAKEy(y=0,1) pin status is from low to high; set BRKxPINV to 1, brake event will occurred when PWM0_BRAKEy pin status is from high to low.

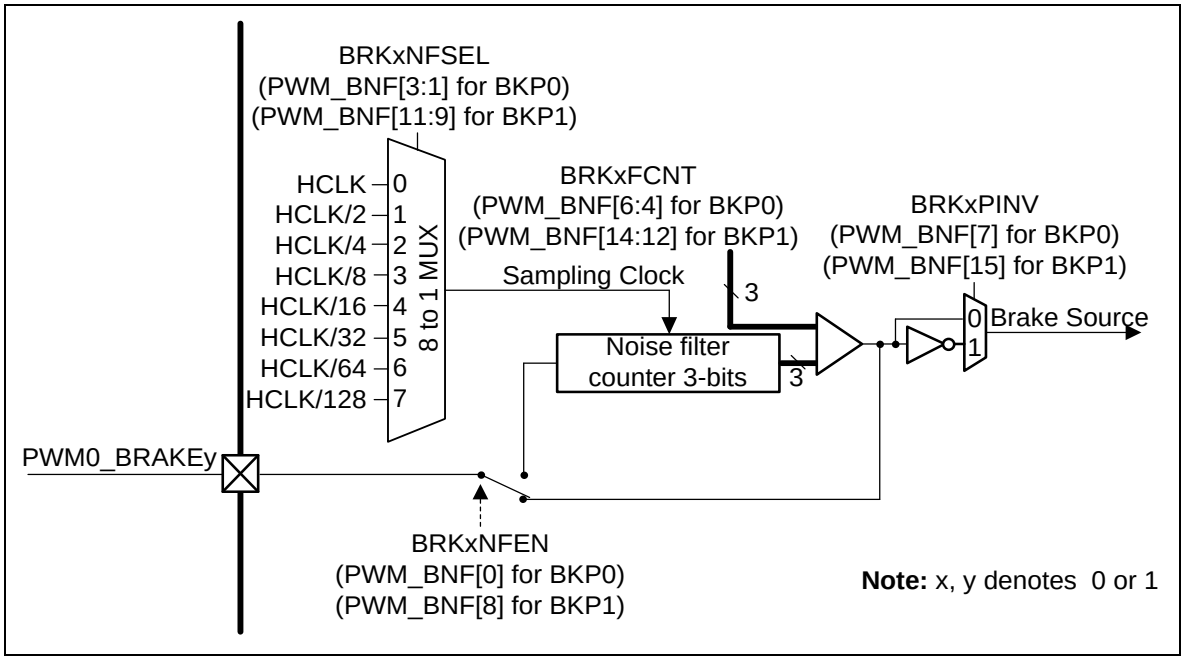


Figure 6.8-30 Brake Noise Filter Block Diagram

For Complementary mode, it is often necessary to set a safe output state to the complement output pairs once the brake event occurs.

Each complementary channel pair shares a PWM brake function, as shown Figure 6.8-31. 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.

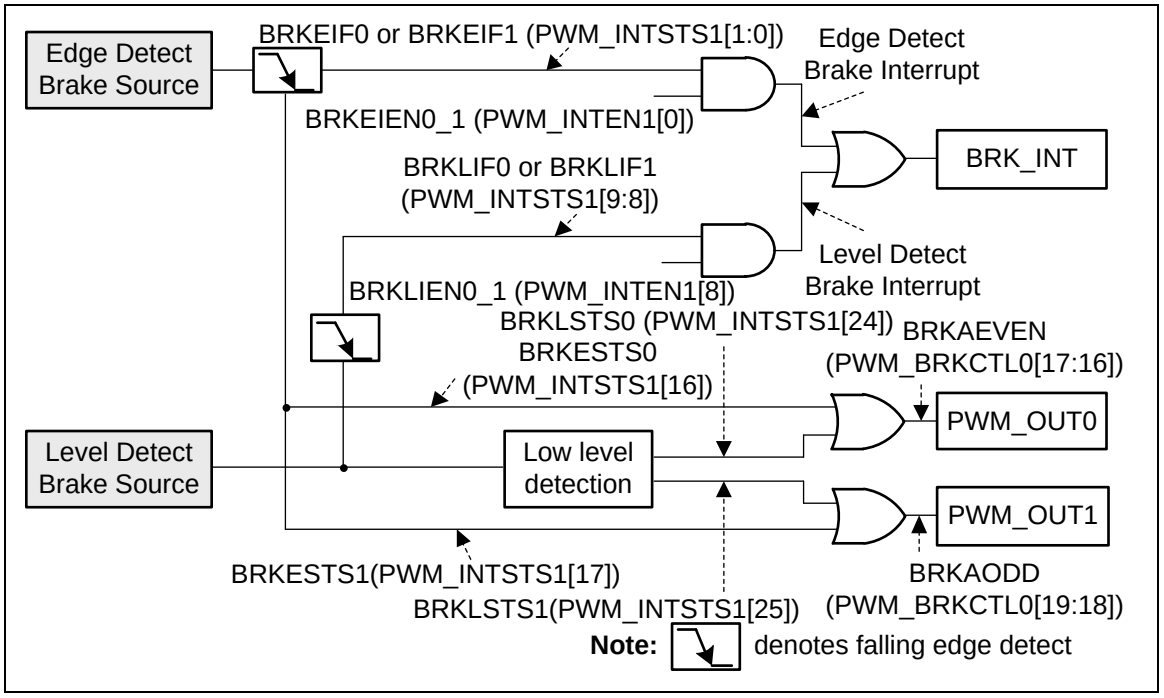


Figure 6.8-31 Brake Block Diagram for PWM0_CH0 and PWM0_CH1 Pair

Figure 6.8-32 illustrates the edge detector waveform for PWM0_CH0 and PWM0_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 bits are also set to indicate brake state of PWM0_CH0 and PWM0_CH1. For the first occurring event, software writes 1 to clear the BRKEIF0 flag. After that, the BRKESTS0 bit is cleared by hardware at the next start of the PWM period. At the same moment, the PWM0_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 flag. Afterward, PWM0_CH1 outputs normally at the next start of the PWM period.

As a contrast to the edge detector example, Figure 6.8-33 illustrates the level detector waveform for PWM0_CH0 and PWM0_CH1 pair. In this case, the BRKLIF0 and BRKLIF1 flags 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 flags are at that moment.

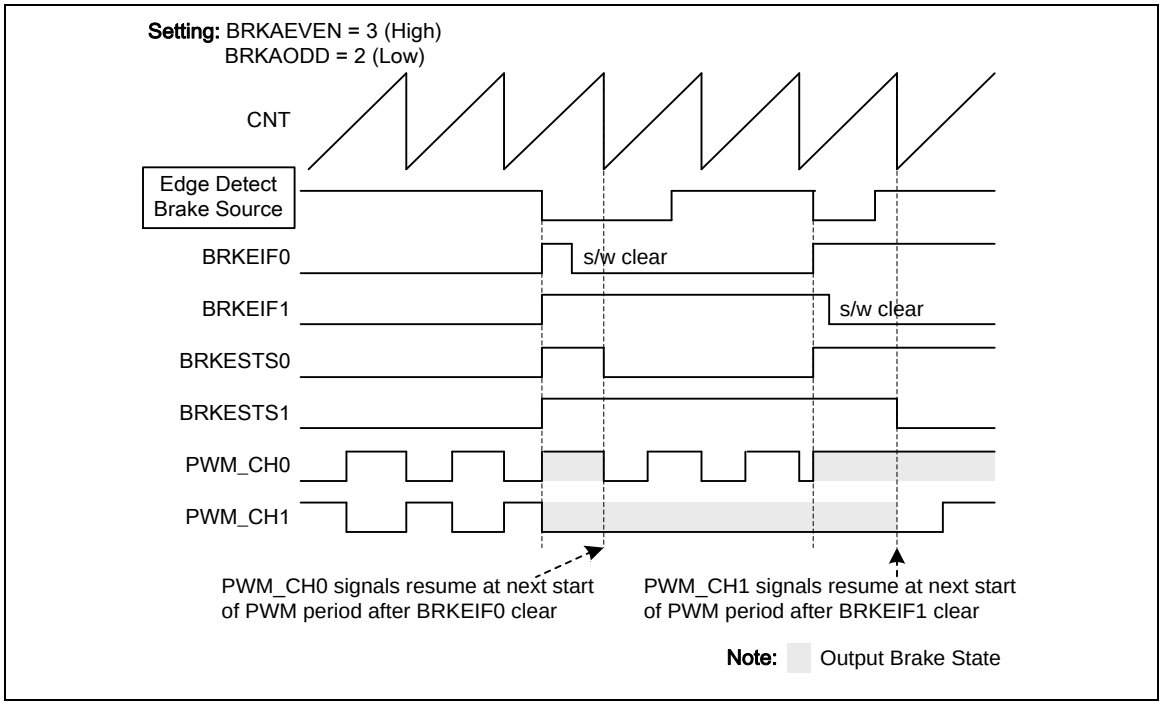


Figure 6.8-32 Edge Detector Waveform for PWM0_CH0 and PWM0_CH1 Pair

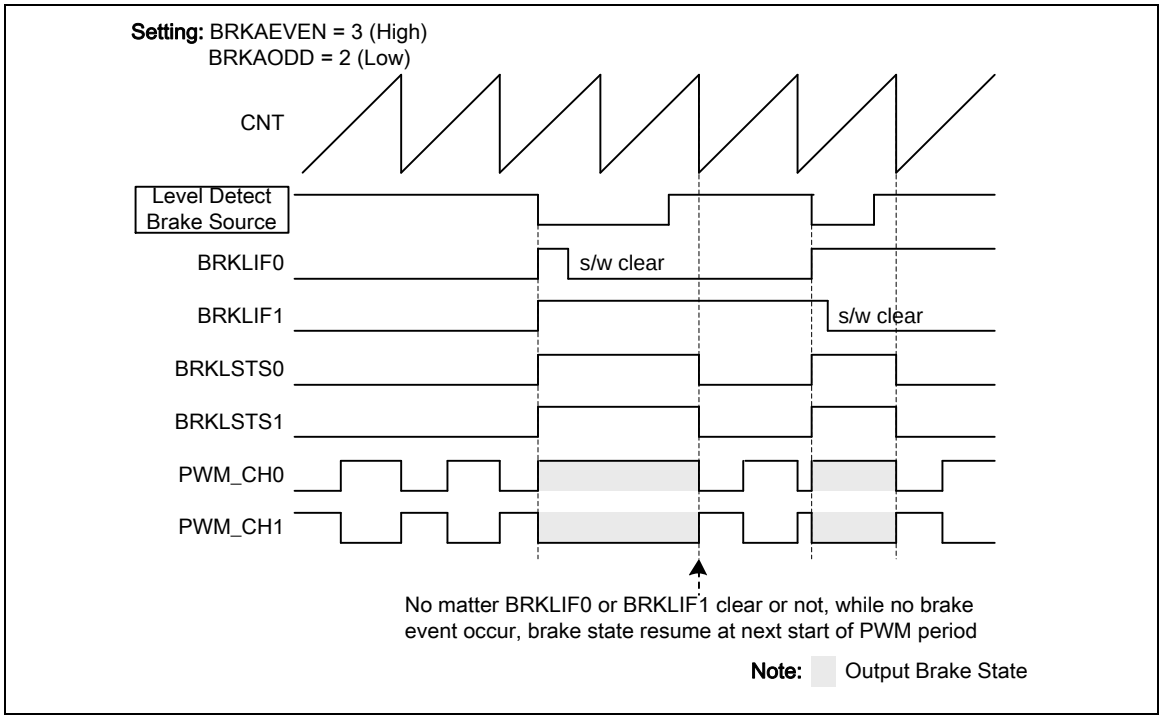


Figure 6.8-33 Level Detector Waveform for PWM0_CH0 and PWM0_CH1 Pair

The two kinds of detectors detect the same five brake sources: two from external input signals, one from ADC result monitor (EADCRM), one from system fail and one from software triggered, that are shown in Figure 6.8-34.

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, SRAM parity check error and Core lockup. Figure 6.8-35 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.

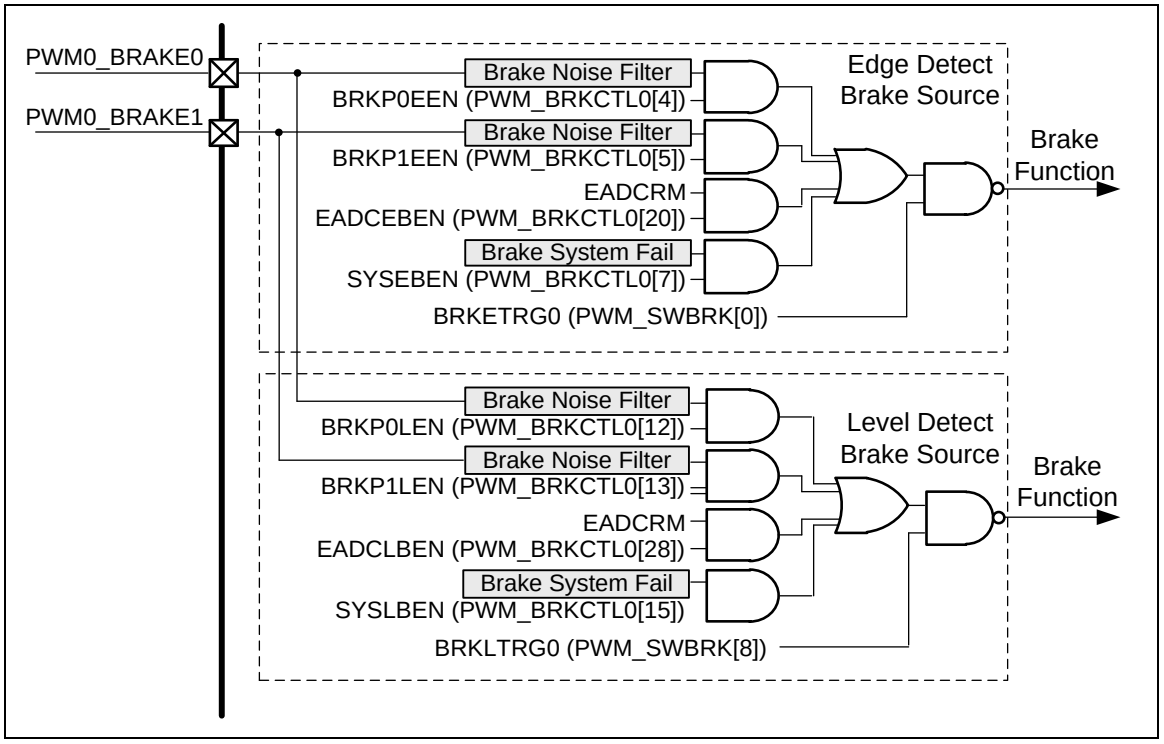


Figure 6.8-34 Brake Source Block Diagram

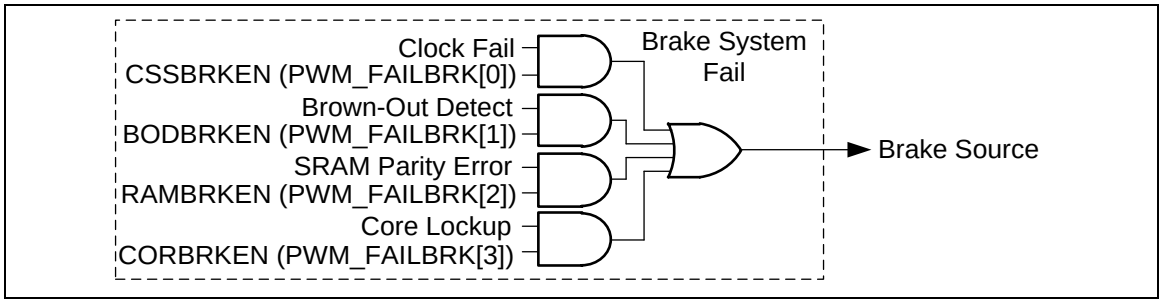
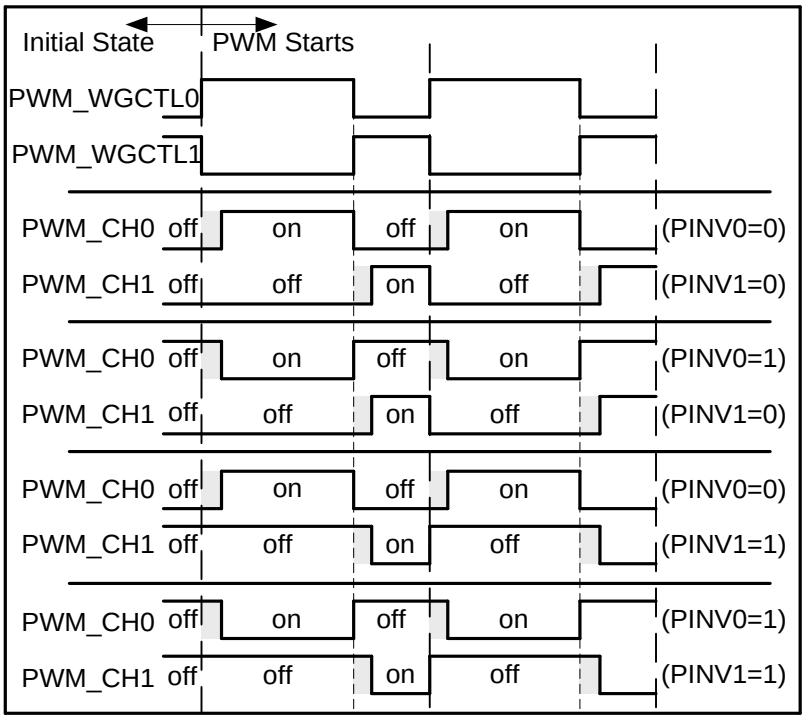


Figure 6.8-35 Brake System Fail Block Diagram

6.8.5.24 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 the 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 6.8-36 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 6.8-36 Initial State and Polarity Control with Rising Edge Dead-Time Insertion

6.8.5.25 PWM Interrupt Generator

There are three independent interrupts for each PWM as shown in Figure 6.8-37.

The 1st PWM interrupt (PWM_INT) comes from PWM complementary pair events. The counter can generate the Zero point Interrupt Flag ZIFn (PWM_INTSTS0[5:0] , n=0,1..5) and the Period point Interrupt Flag PIFn (PWM_INTSTS0[13:8] , n=0,1..5). When PWM channel n's counter equals to the comparator value stored in PWM_CMPDATn register, 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. If the corresponding 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 CAPRLIFn (PWM_CAPIF[5:0]) flag 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 CAPFLIFn (PWM_CAPIF[13:8]) flag 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 details of the BRK_INT is described in the PWM Brake section.

The Figure 6.8-37 demonstrates the architecture of the PWM interrupts.

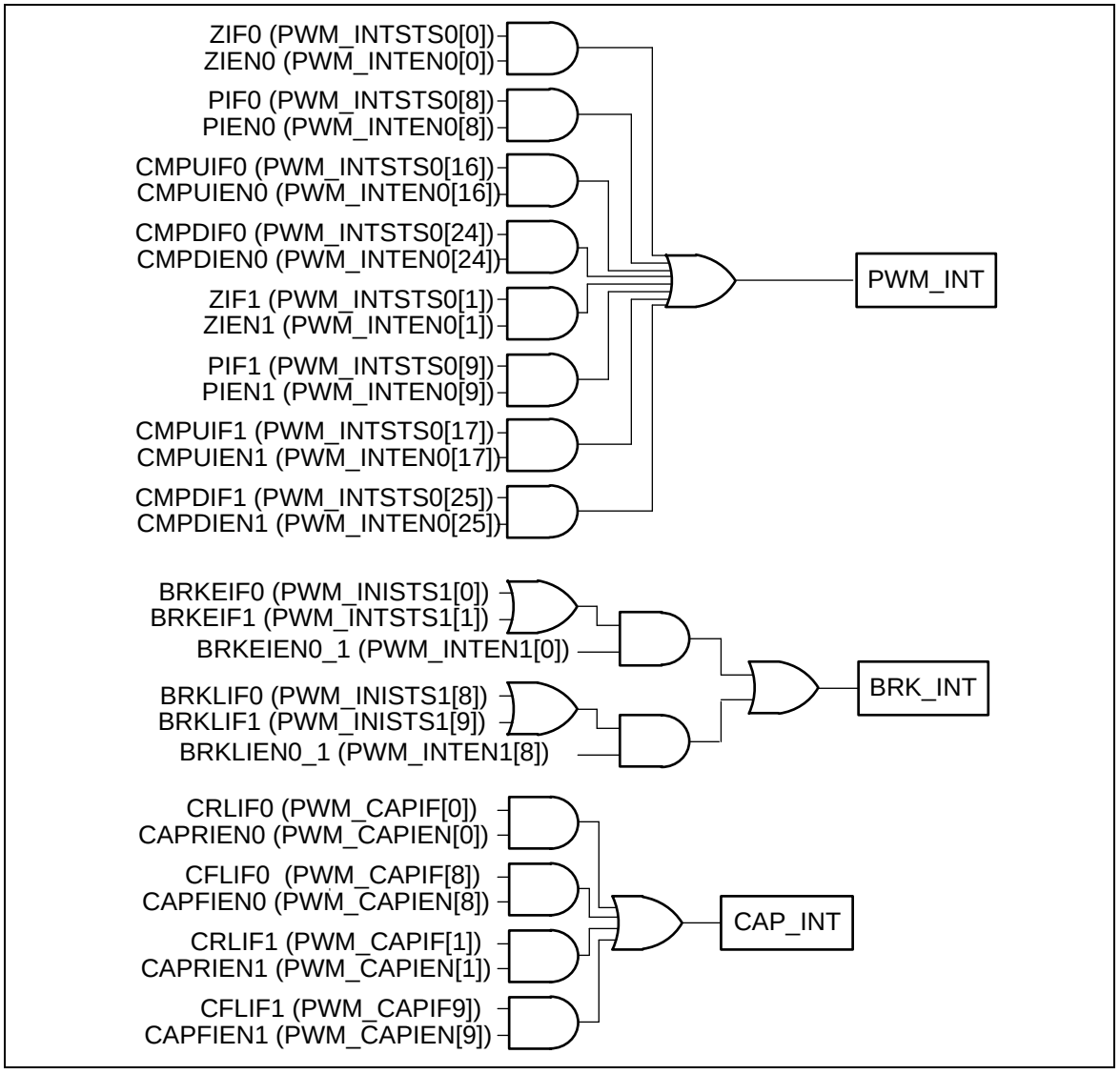


Figure 6.8-37 PWM0_CH0 and PWM0_CH1 Pair Interrupt Architecture Diagram

6.8.5.26 PWM Trigger EADC Generator

PWM can be one of the EADC conversion trigger source. Each PWM pair channels share the same trigger source. Setting TRGSELn bit of PWM_ADCTS0 and PWM_ADCTS1 registers is to select the trigger sources, where TRGSELn bit is TRGSEL0, TRGSEL1, …, and TRGSEL5, which are located in PWM_EADCTS0[3:0], PWM_EADCTS0[11:8], PWM_EADCTS0[19:16], PWM_EADCTS0[27:24], PWM_EADCTS1[3:0] and PWM_EADCTS1[11:8], respectively. Setting TRGENn bit of PWM_EADCTS0 and PWM_EADCTS1 registers is to enable the trigger output to EADC, where TRGENn bit is TRGEN0, TRGEN1, …, TRGEN5, which are located in PWM_EADCTS0[7], PWM_EADCTS0[15], PWM_EADCTS0[23], PWM_EADCTS0[31], PWM_EADCTS1[7] and PWM_EADCTS1[15], respectively. The number n (n = 0,1, ..,5) denotes PWM channel number.

There are 16 PWM events can be selected as the trigger source for one pair of channels which shown in Figure 6.8-38. Figure 6.8-39 is the trigger EADC timing waveform in the up-down counter type.

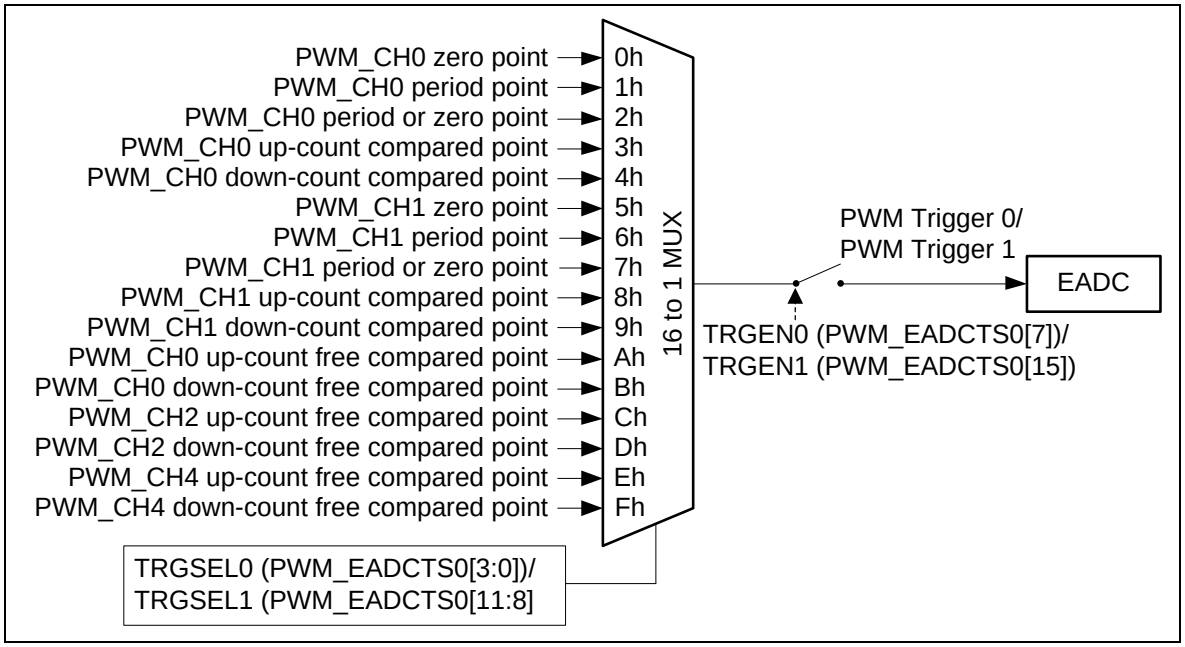


Figure 6.8-38 PWM0_CH0 and PWM0_CH1 Pair Trigger EADC Block Diagram

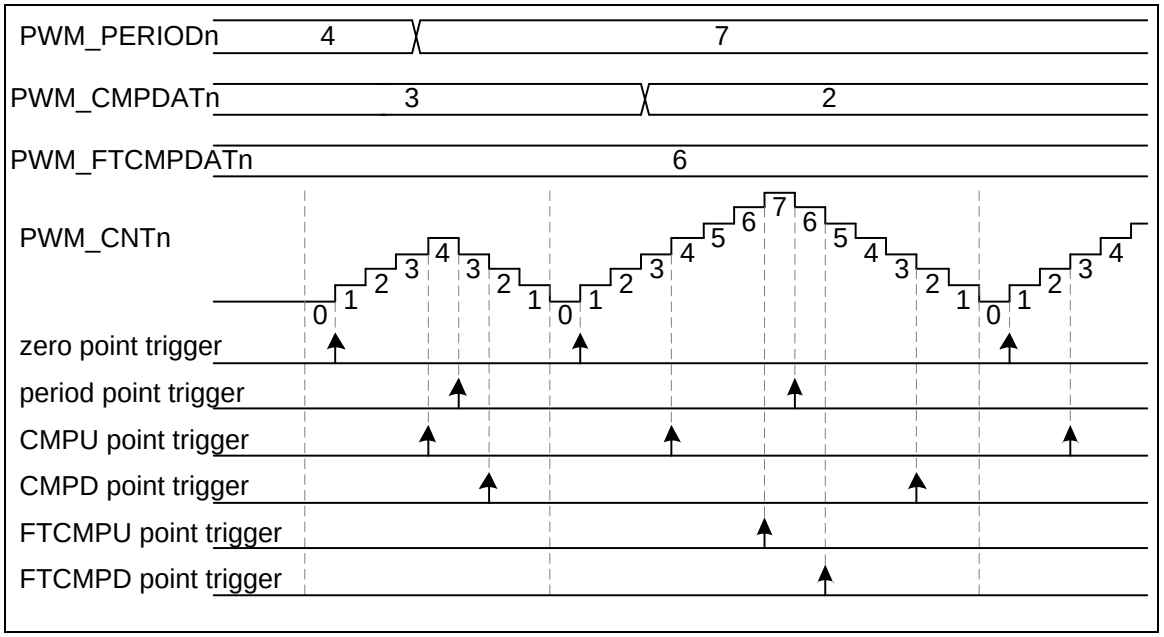


Figure 6.8-39 PWM Trigger EADC in Up-Down Counter Type Timing Waveform

6.8.5.27 Capture Operation

The channels of the capture input and the PWM output share the same pin and counter. The counter can operating in up or down counter type. The capture function will always latch the PWM counter to the RCAPDATn (PWM_RCAPDATn[15:0]) bits or the FCAPDATn (PWM_FCAPDATn[15:0]) bits, 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]) bit is for the rising edge and the CAPFIENn (PWM_CAPIEN[13:8]) bit is for the falling edge. When rising or falling latch occurs, the corresponding PWM counter may be reloaded with the value of PWM_PERIODn register, depending on the setting of RCRLDENn or FCRLDENn bits (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]) bits for the corresponding capture channel n. Figure 6.8-40 is the capture block diagram of channel 0.

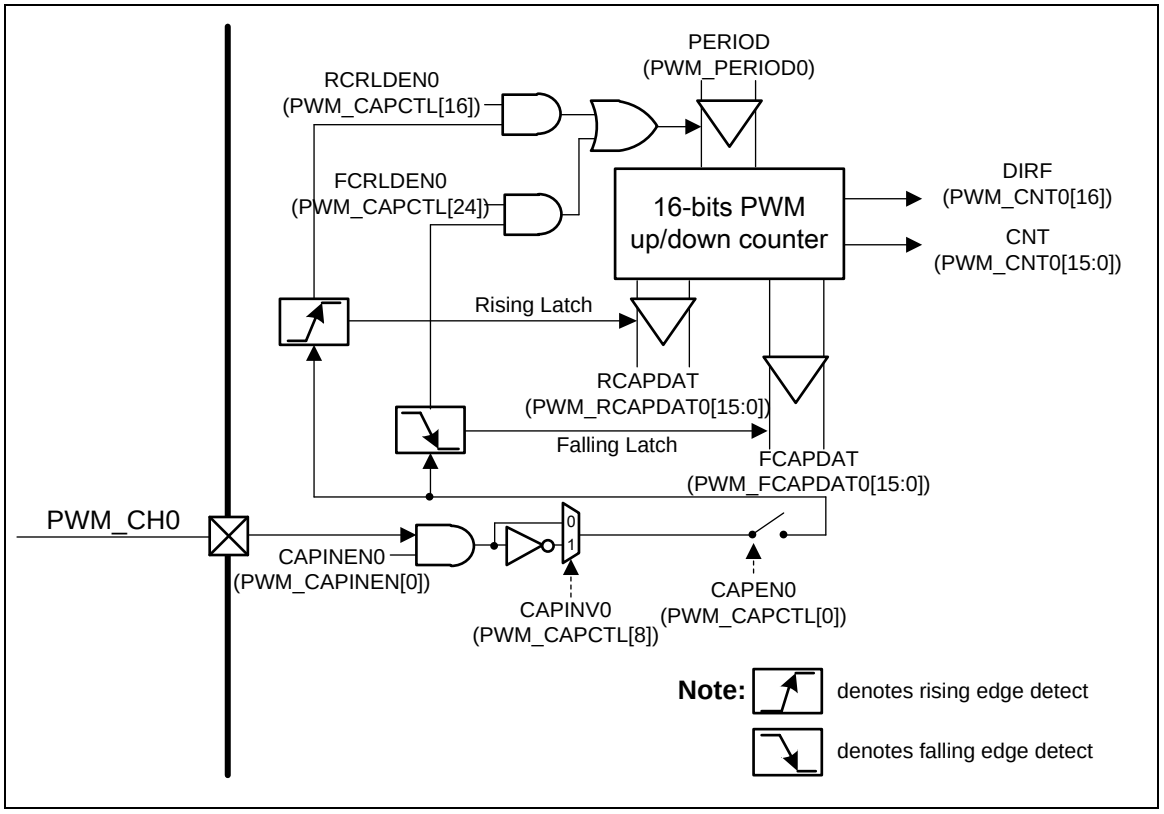


Figure 6.8-40 PWM0_CH0 Capture Block Diagram

Figure 6.8-41 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 the counter value to the PWM_FCAPDATn register. When detecting the rising edge, it latches the counter value to the PWM_RCAPDATn register. 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 bit is enabled. But at the second time, the falling edge does not result in a reload because of the disabled FCRLDENn bit. In this example, the counter also reloads at the rising edge of the capture input because the RCRLDENn bit 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.

Figure 6.8-41 also illustrates the timing example for the interrupt and interrupt flag generation. When the rising edge at channel n is detected, the corresponding CRLIFn (PWM_CAPIF[5:0]) bit is set by hardware. Similarly, a falling edge detection at channel n causes the corresponding CFLIFn (PWM_CAPIF[13:8]) bit is set by hardware. CRLIFn and CFLIFn bits can be cleared by

software by writing ‘1’. If the CRLIFn bit is set and the CAPRIENn bit is enabled, the capture function generates an interrupt. If the CFLIFn bit is set and the CAPFIENn bit is enabled, the interrupt also happens.

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

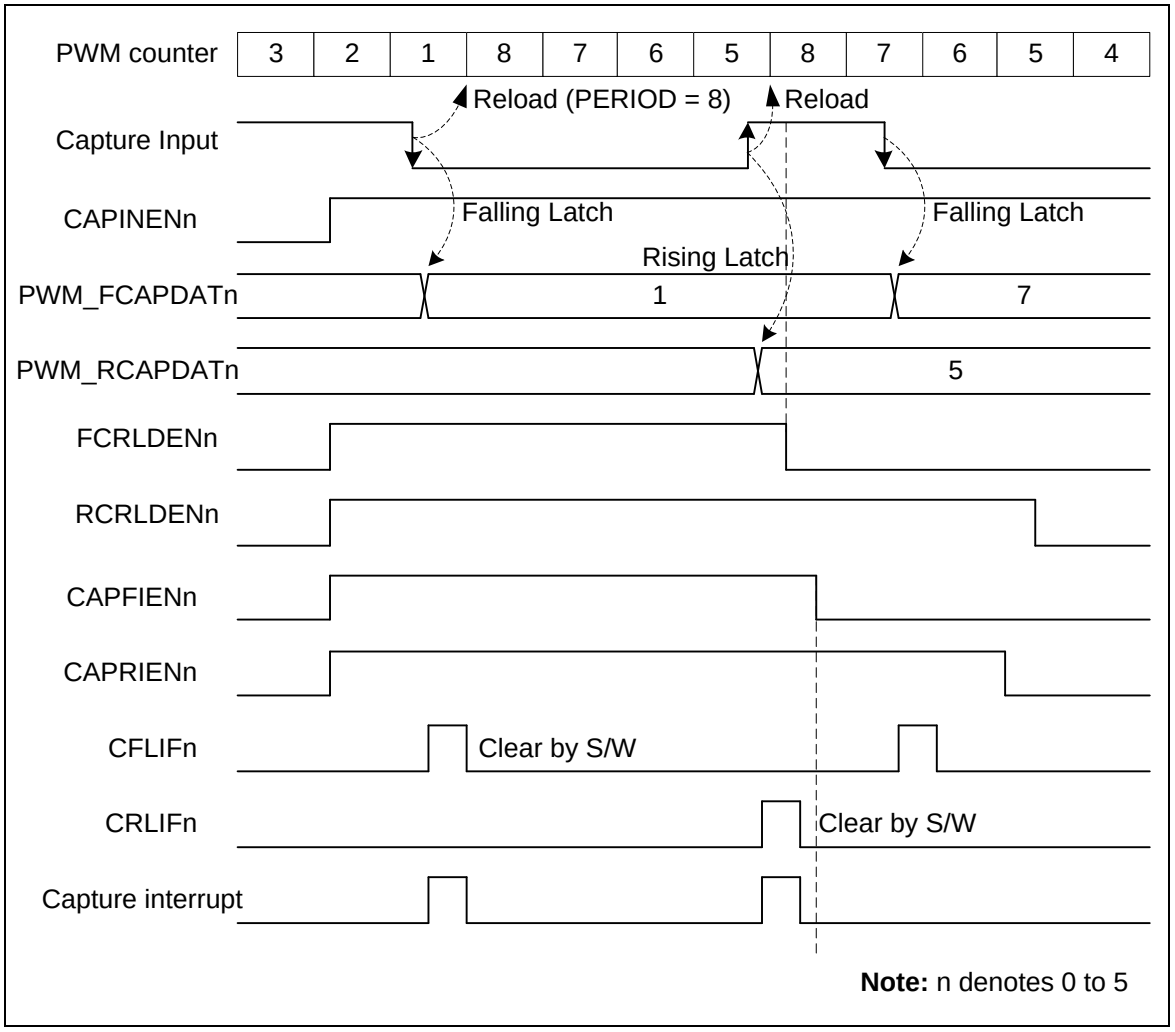


Figure 6.8-41 Capture Operation Waveform

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) PWM counter time, where one PWM counter time is (CLKPSC+1) * PWM0_CLK clock time. In Figure 6.8-41, the low pulse width is 8+1-5 = 4 PWM counter time.

For the positive pulse case, the channel high pulse width is calculated as (PWM_PERIODn + 1 - PWM_FCAPDATn) PWM counter time, where one PWM counter time is (CLKPSC+1) * PWM0_CLK clock time. In Figure 6.8-41, the high pulse width is 8+1-7 = 2 PWM counter time.

6.8.5.28 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]) bits, 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]) bit 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])) bit is used to decide either channel n or channel m can be serviced by the PDMA channel.

Figure 6.8-42 is capture PDMA waveform. In this case, the CHSEL0_1 (PWM_PDMACTL[4]) bit is set to 0. Hence the PDMA will service channel 0 for the capture data transfer. CAPMOD0_1 (PWM_PDMACTL[2:1]) bits are 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[1]) bit is set to 1, so the rising edge data will be the first data to transfer and following is the falling edge data to transfer. As the figure shows, the last assertions of the CAPRIF0 CRLIF0 and CAPFIF0 CFLIF0 signal have some overlap. The value of PWM_RCAPDAT0 register is 11 will be loaded to PWM_PDMACAP0_1 register to wait for transfer but not the PWM_FCAPDAT0 value. The PWM_PDMACAP0_1 register 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.

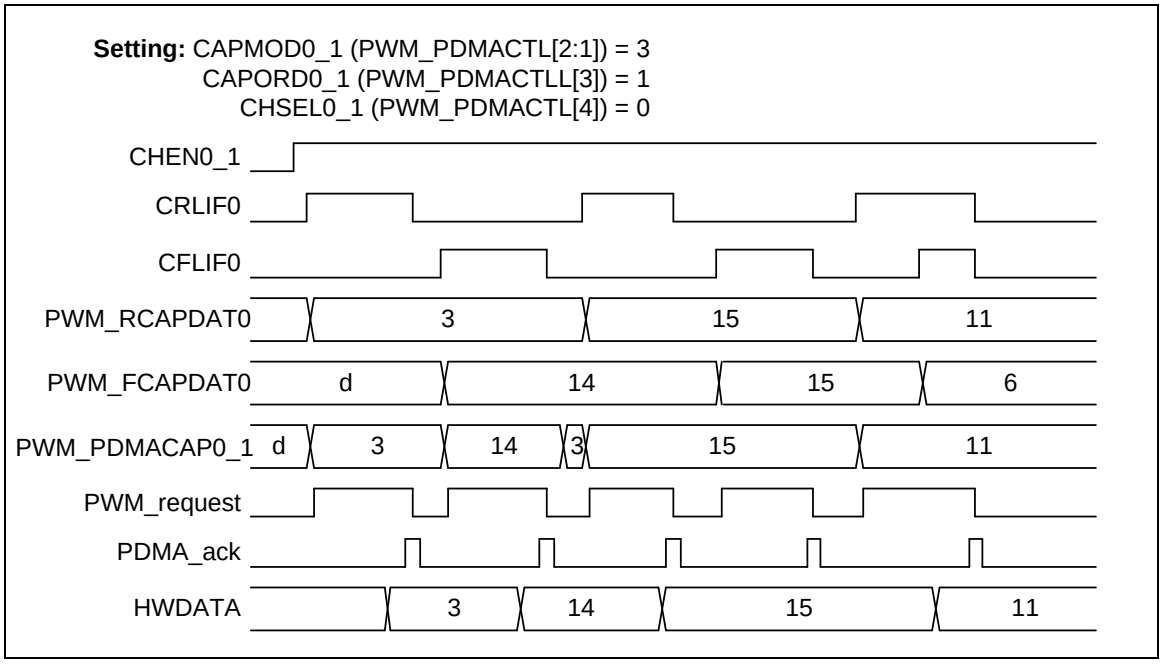


Figure 6.8-42 Capture PDMA Operation Waveform of Channel 0

6.8.6 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 = 0x4005_8000**				
PWM_CTL0	PWM0_BA+0x00	R/W	PWM Control Register 0	0x0000_0000
PWM_CTL1	PWM0_BA+0x04	R/W	PWM Control Register 1	0x0000_0000
PWM_SYNC	PWM0_BA+0x08	R/W	PWM Synchronization Register	0x0000_0000
PWM_SWSYNC	PWM0_BA+0x0C	R/W	PWM Software Control Synchronization Register	0x0000_0000
PWM_CLKSRC	PWM0_BA+0x10	R/W	PWM Clock Source Register	0x0000_0000
PWM_CLKPSC0_1	PWM0_BA+0x14	R/W	PWM Clock Pre-scale Register 0/1	0x0000_0000
PWM_CLKPSC2_3	PWM0_BA+0x18	R/W	PWM Clock Pre-scale Register 2/3	0x0000_0000
PWM_CLKPSC4_5	PWM0_BA+0x1C	R/W	PWM Clock Pre-scale Register 4/5	0x0000_0000
PWM_CNTEN	PWM0_BA+0x20	R/W	PWM Counter Enable Register	0x0000_0000
PWM_CNTCLR	PWM0_BA+0x24	R/W	PWM Clear Counter Register	0x0000_0000
PWM_LOAD	PWM0_BA+0x28	R/W	PWM Load Register	0x0000_0000
PWM_PERIOD0	PWM0_BA+0x30	R/W	PWM Period Register 0	0x0000_0000
PWM_PERIOD1	PWM0_BA+0x34	R/W	PWM Period Register 1	0x0000_0000
PWM_PERIOD2	PWM0_BA+0x38	R/W	PWM Period Register 2	0x0000_0000
PWM_PERIOD3	PWM0_BA+0x3C	R/W	PWM Period Register 3	0x0000_0000
PWM_PERIOD4	PWM0_BA+0x40	R/W	PWM Period Register 4	0x0000_0000
PWM_PERIOD5	PWM0_BA+0x44	R/W	PWM Period Register 5	0x0000_0000
PWM_CMPDAT0	PWM0_BA+0x50	R/W	PWM Comparator Register 0	0x0000_0000
PWM_CMPDAT1	PWM0_BA+0x54	R/W	PWM Comparator Register 1	0x0000_0000
PWM_CMPDAT2	PWM0_BA+0x58	R/W	PWM Comparator Register 2	0x0000_0000
PWM_CMPDAT3	PWM0_BA+0x5C	R/W	PWM Comparator Register 3	0x0000_0000
PWM_CMPDAT4	PWM0_BA+0x60	R/W	PWM Comparator Register 4	0x0000_0000
PWM_CMPDAT5	PWM0_BA+0x64	R/W	PWM Comparator Register 5	0x0000_0000
PWM_DTCTL0_1	PWM0_BA+0x70	R/W	PWM Dead-Time Control Register 0/1	0x0000_0000
PWM_DTCTL2_3	PWM0_BA+0x74	R/W	PWM Dead-Time Control Register 2/3	0x0000_0000
PWM_DTCTL4_5	PWM0_BA+0x78	R/W	PWM Dead-Time Control Register 4/5	0x0000_0000

Register	Offset	R/W	Description	Reset Value
PWM Base Address: **PWM0_BA = 0x4005_8000**				
PWM_PHS0_1	PWM0_BA+0x80	R/W	PWM Counter Phase Register 0/1	0x0000_0000
PWM_PHS2_3	PWM0_BA+0x84	R/W	PWM Counter Phase Register 2/3	0x0000_0000
PWM_PHS4_5	PWM0_BA+0x88	R/W	PWM Counter Phase Register 4/5	0x0000_0000
PWM_CNT0	PWM0_BA+0x90	R	PWM Counter Register 0	0x0000_0000
PWM_CNT1	PWM0_BA+0x94	R	PWM Counter Register 1	0x0000_0000
PWM_CNT2	PWM0_BA+0x98	R	PWM Counter Register 2	0x0000_0000
PWM_CNT3	PWM0_BA+0x9C	R	PWM Counter Register 3	0x0000_0000
PWM_CNT4	PWM0_BA+0xA0	R	PWM Counter Register 4	0x0000_0000
PWM_CNT5	PWM0_BA+0xA4	R	PWM Counter Register 5	0x0000_0000
PWM_WGCTL0	PWM0_BA+0xB0	R/W	PWM Generation Register 0	0x0000_0000
PWM_WGCTL1	PWM0_BA+0xB4	R/W	PWM Generation Register 1	0x0000_0000
PWM_MSKEN	PWM0_BA+0xB8	R/W	PWM Mask Enable Register	0x0000_0000
PWM_MSK	PWM0_BA+0xBC	R/W	PWM Mask Data Register	0x0000_0000
PWM_BNF	PWM0_BA+0xC0	R/W	PWM Brake Noise Filter Register	0x0000_0000
PWM_FAILBRK	PWM0_BA+0xC4	R/W	PWM System Fail Brake Control Register	0x0000_0000
PWM_BRKCTL0_1	PWM0_BA+0xC8	R/W	PWM Brake Edge Detect Control Register 0/1	0x0000_0000
PWM_BRKCTL2_3	PWM0_BA+0xCC	R/W	PWM Brake Edge Detect Control Register 2/3	0x0000_0000
PWM_BRKCTL4_5	PWM0_BA+0xD0	R/W	PWM Brake Edge Detect Control Register 4/5	0x0000_0000
PWM_POLCTL	PWM0_BA+0xD4	R/W	PWM Pin Polar Inverse Register	0x0000_0000
PWM_POEN	PWM0_BA+0xD8	R/W	PWM Output Enable Register	0x0000_0000
PWM_SWBRK	PWM0_BA+0xDC	W	PWM Software Brake Control Register	0x0000_0000
PWM_INTEN0	PWM0_BA+0xE0	R/W	PWM Interrupt Enable Register 0	0x0000_0000
PWM_INTEN1	PWM0_BA+0xE4	R/W	PWM Interrupt Enable Register 1	0x0000_0000
PWM_INTSTS0	PWM0_BA+0xE8	R/W	PWM Interrupt Flag Register 0	0x0000_0000
PWM_INTSTS1	PWM0_BA+0xEC	R/W	PWM Interrupt Flag Register 1	0x0000_0000
PWM_EADCTS0	PWM0_BA+0xF8	R/W	PWM Trigger EADC Source Select Register 0	0x0000_0000
PWM_EADCTS1	PWM0_BA+0xFC	R/W	PWM Trigger EADC Source Select Register 1	0x0000_0000
PWM_FTCMPDAT0_1	PWM0_BA+0x100	R/W	PWM Free Trigger Compare Register 0/1	0x0000_0000

Register	Offset	R/W	Description	Reset Value
PWM Base Address: **PWM0_BA = 0x4005_8000**				
PWM_FTCMPDAT2_3	PWM0_BA+0x104	R/W	PWM Free Trigger Compare Register 2/3	0x0000_0000
PWM_FTCMPDAT4_5	PWM0_BA+0x108	R/W	PWM Free Trigger Compare Register 4/5	0x0000_0000
PWM_SSCTL	PWM0_BA+0x110	R/W	PWM Synchronous Start Control Register	0x0000_0000
PWM_SSTRG	PWM0_BA+0x114	W	PWM Synchronous Start Trigger Register	0x0000_0000
PWM_STATUS	PWM0_BA+0x120	R/W	PWM Status Register	0x0000_0000
PWM_CAPINEN	PWM0_BA+0x200	R/W	PWM Capture Input Enable Register	0x0000_0000
PWM_CAPCTL	PWM0_BA+0x204	R/W	PWM Capture Control Register	0x0000_0000
PWM_CAPSTS	PWM0_BA+0x208	R	PWM Capture Status Register	0x0000_0000
PWM_RCAPDAT0	PWM0_BA+0x20C	R	PWM Rising Capture Data Register 0	0x0000_0000
PWM_FCAPDAT0	PWM0_BA+0x210	R	PWM Falling Capture Data Register 0	0x0000_0000
PWM_RCAPDAT1	PWM0_BA+0x214	R	PWM Rising Capture Data Register 1	0x0000_0000
PWM_FCAPDAT1	PWM0_BA+0x218	R	PWM Falling Capture Data Register 1	0x0000_0000
PWM_RCAPDAT2	PWM0_BA+0x21C	R	PWM Rising Capture Data Register 2	0x0000_0000
PWM_FCAPDAT2	PWM0_BA+0x220	R	PWM Falling Capture Data Register 2	0x0000_0000
PWM_RCAPDAT3	PWM0_BA+0x224	R	PWM Rising Capture Data Register 3	0x0000_0000
PWM_FCAPDAT3	PWM0_BA+0x228	R	PWM Falling Capture Data Register 3	0x0000_0000
PWM_RCAPDAT4	PWM0_BA+0x22C	R	PWM Rising Capture Data Register 4	0x0000_0000
PWM_FCAPDAT4	PWM0_BA+0x230	R	PWM Falling Capture Data Register 4	0x0000_0000
PWM_RCAPDAT5	PWM0_BA+0x234	R	PWM Rising Capture Data Register 5	0x0000_0000
PWM_FCAPDAT5	PWM0_BA+0x238	R	PWM Falling Capture Data Register 5	0x0000_0000
PWM_PDMACTL	PWM0_BA+0x23C	R/W	PWM PDMA Control Register	0x0000_0000
PWM_PDMACAP0_1	PWM0_BA+0x240	R	PWM Capture Channel 01 PDMA Register	0x0000_0000
PWM_PDMACAP2_3	PWM0_BA+0x244	R	PWM Capture Channel 23 PDMA Register	0x0000_0000
PWM_PDMACAP4_5	PWM0_BA+0x248	R	PWM Capture Channel 45 PDMA Register	0x0000_0000
PWM_CAPIEN	PWM0_BA+0x250	R/W	PWM Capture Interrupt Enable Register	0x0000_0000
PWM_CAPIF	PWM0_BA+0x254	R/W	PWM Capture Interrupt Flag Register	0x0000_0000
PWM_PBUF0	PWM0_BA+0x304	R	PWM PERIOD0 Buffer	0x0000_0000
PWM_PBUF1	PWM0_BA+0x308	R	PWM PERIOD1 Buffer	0x0000_0000

Register	Offset	R/W	Description	Reset Value
PWM Base Address: **PWM0_BA = 0x4005_8000**				
PWM_PBUF2	PWM0_BA+0x30C	R	PWM PERIOD2 Buffer	0x0000_0000
PWM_PBUF3	PWM0_BA+0x310	R	PWM PERIOD3 Buffer	0x0000_0000
PWM_PBUF4	PWM0_BA+0x314	R	PWM PERIOD4 Buffer	0x0000_0000
PWM_PBUF5	PWM0_BA+0x318	R	PWM PERIOD5 Buffer	0x0000_0000
PWM_CMPBUF0	PWM0_BA+0x31C	R	PWM CMPDAT0 Buffer	0x0000_0000
PWM_CMPBUF1	PWM0_BA+0x320	R	PWM CMPDAT1 Buffer	0x0000_0000
PWM_CMPBUF2	PWM0_BA+0x324	R	PWM CMPDAT2 Buffer	0x0000_0000
PWM_CMPBUF3	PWM0_BA+0x328	R	PWM CMPDAT3 Buffer	0x0000_0000
PWM_CMPBUF4	PWM0_BA+0x32C	R	PWM CMPDAT4 Buffer	0x0000_0000
PWM_CMPBUF5	PWM0_BA+0x330	R	PWM CMPDAT5 Buffer	0x0000_0000
PWM_CPSCBUF0_1	PWM0_BA+0x334	R	PWM CLKPSC0_1 Buffer	0x0000_0000
PWM_CPSCBUF2_3	PWM0_BA+0x338	R	PWM CLKPSC2_3 Buffer	0x0000_0000
PWM_CPSCBUF4_5	PWM0_BA+0x33C	R	PWM CLKPSC4_5 Buffer	0x0000_0000
PWM_FTCBUF0_1	PWM0_BA+0x340	R	PWM FTCMPDAT0_1 Buffer	0x0000_0000
PWM_FTCBUF2_3	PWM0_BA+0x344	R	PWM FTCMPDAT2_3 Buffer	0x0000_0000
PWM_FTCBUF4_5	PWM0_BA+0x348	R	PWM FTCMPDAT4_5 Buffer	0x0000_0000
PWM_FTCI	PWM0_BA+0x34C	R/W	PWM FTCMPDAT Indicator Register	0x0000_0000

Note:

1. Any register not listed here is reserved and must not be written. The result of a read operation on these bits is undefined.
2. The reserved register fields that listed in register description must be written to their reset value. Writing reserved fields with other than reset values may produce undefined results.

6.8.7 Register Description

PWM Control Register 0 (PWM_CTL0)

Register	Offset	R/W	Description	Reset Value
PWM_CTL0	PWM0_BA+0x00	R/W	PWM Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
DBGTRIOFF	DBGHALT	Reserved					GROUPEN
23	22	21	20	19	18	17	16
Reserved		IMMLDEN5	IMMLDEN4	IMMLDEN3	IMMLDEN2	IMMLDEN1	IMMLDEN0
15	14	13	12	11	10	9	8
Reserved		WINLDEN5	WINLDEN4	WINLDEN3	WINLDEN2	WINLDEN1	WINLDEN0
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 Protected)** 0 = ICE debug mode acknowledgement effects 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 register is write protected. Refer to SYS_REGLCTL register.
[30]	**DBGHALT**	**ICE Debug Mode Counter Halt (Write Protected)** 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 register is write protected. Refer to SYS_REGLCTL register.
[29:26]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[24]	**GROUPEN**	**Group Function Enable Bit(S)** 0 = The output waveform of each PWM channel are independent. 1 = Unify the PWM_CH2 and PWM_CH4 to output the same waveform as PWM_CH0 and unify the PWM_CH3 and PWM_CH5 to output the same waveform as PWM_CH1.
[23:22]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[21]	**IMMLDEN5**	**PWM Channel 5 Immediately Load Enable Bits** 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 IMMLDEN5 is enabled, WINLDEN5 and CTRLD5 will be invalid.

[20]	**IMMLDEN4**	**PWM Channel 4 Immediately Load Enable Bits** 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 IMMLDEN4 is enabled, WINLDEN4 and CTRLD4 will be invalid.
[19]	**IMMLDEN3**	**PWM Channel 3 Immediately Load Enable Bits** 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 IMMLDEN3 is enabled, WINLDEN3 and CTRLD3 will be invalid.
[18]	**IMMLDEN2**	**PWM Channel 2 Immediately Load Enable Bits** 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 IMMLDEN2 is enabled, WINLDEN2 and CTRLD2 will be invalid.
[17]	**IMMLDEN1**	**PWM Channel 1 Immediately Load Enable Bits** 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 IMMLDEN1 is enabled, WINLDEN1 and CTRLD1 will be invalid.
[16]	**IMMLDEN0**	**PWM Channel 0 Immediately Load Enable Bits** 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 IMMLDEN0 is enabled, WINLDEN0 and CTRLD0 will be invalid.
[15:14]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[13]	**WINLDEN5**	**PWM Channel 5 Window Load Enable Bits** 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 will load to PBUF at the end point of each period. CMPDAT will load to CMPBUF at the end point of each period when valid reload window is set. The valid reload window is set by software write 1 to PWM_LOAD register and cleared by hardware after load success.
[12]	**WINLDEN4**	**PWM Channel 4 Window Load Enable Bits** 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 will load to PBUF at the end point of each period. CMPDAT will load to CMPBUF at the end point of each period when valid reload window is set. The valid reload window is set by software write 1 to PWM_LOAD register and cleared by hardware after load success.
[11]	**WINLDEN3**	**PWM Channel 3 Window Load Enable Bits** 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 will load to PBUF at the end point of each period. CMPDAT will load to CMPBUF at the end point of each period when valid reload window is set. The valid reload window is set by software write 1 to PWM_LOAD register and cleared by hardware after load success.

[10]	**WINLDEN2**	**PWM Channel 2 Window Load Enable Bits** 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 will load to PBUF at the end point of each period. CMPDAT will load to CMPBUF at the end point of each period when valid reload window is set. The valid reload window is set by software write 1 to PWM_LOAD register and cleared by hardware after load success.
[9]	**WINLDEN1**	**PWM Channel 1 Window Load Enable Bits** 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 will load to PBUF at the end point of each period. CMPDAT will load to CMPBUF at the end point of each period when valid reload window is set. The valid reload window is set by software write 1 to PWM_LOAD register and cleared by hardware after load success.
[8]	**WINLDEN0**	**PWM Channel 0 Window Load Enable Bits** 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 will load to PBUF at the end point of each period. CMPDAT will load to CMPBUF at the end point of each period when valid reload window is set. The valid reload window is set by software write 1 to PWM_LOAD register and cleared by hardware after load success.
[7:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5]	**CTRLD5**	**PWM Channel 5 Center Re-load** 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.
[4]	**CTRLD4**	**PWM Channel 4 Center Re-load** 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.
[3]	**CTRLD3**	**PWM Channel 3 Center Re-load** 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.
[2]	**CTRLD2**	**PWM Channel 2 Center Re-load** 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.
[1]	**CTRLD1**	**PWM Channel 1 Center Re-load** 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.
[0]	**CTRLD0**	**PWM Channel 0 Center Re-load** 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	PWM0_BA+0x04	R/W	PWM Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					OUTMODE4	OUTMODE2	OUTMODE0
23	22	21	20	19	18	17	16
Reserved		CNTMODE5	CNTMODE4	CNTMODE3	CNTMODE2	CNTMODE1	CNTMODE0
15	14	13	12	11	10	9	8
Reserved				CNTTYPE5		CNTTYPE4	
7	6	5	4	3	2	1	0
CNTTYPE3		CNTTYPE2		CNTTYPE1		CNTTYPE0	

Bits	Description	
[31:27]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[26]	**OUTMODE4**	**PWM Channel 4 Output Mode** 0 = PWM independent mode. 1 = PWM complementary mode. **Note:** When operating in group function, these bits must all set to the same mode.
[25]	**OUTMODE2**	**PWM Channel 2 Output Mode** 0 = PWM independent mode. 1 = PWM complementary mode. **Note:** When operating in group function, these bits must all set to the same mode.
[24]	**OUTMODE0**	**PWM Channel 0 Output Mode** 0 = PWM independent mode. 1 = PWM complementary mode. **Note:** When operating in group function, these bits must all set to the same mode.
[23:22]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[21]	**CNTMODE5**	**PWM Channel 5 Counter Mode** 0 = Auto-reload mode. 1 = One-shot mode.
[20]	**CNTMODE4**	**PWM Channel 4 Counter Mode** 0 = Auto-reload mode. 1 = One-shot mode.
[19]	**CNTMODE3**	**PWM Channel 3 Counter Mode** 0 = Auto-reload mode. 1 = One-shot mode.

[18]	**CNTMODE2**	**PWM Channel 2 Counter Mode** 0 = Auto-reload mode. 1 = One-shot mode.
[17]	**CNTMODE1**	**PWM Channel 1 Counter Mode** 0 = Auto-reload mode. 1 = One-shot mode.
[16]	**CNTMODE0**	**PWM Channel 0 Counter Mode** 0 = Auto-reload mode. 1 = One-shot mode.
[15:12]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[11:10]	**CNTTYPE5**	**PWM Channel 5 Counter Behavior Type** 00 = Up counter type (supports in capture mode). 01 = Down count type (supports in capture mode). 10 = Up-down counter type. 11 = Reserved. Do not use.
[9:8]	**CNTTYPE4**	**PWM Channel 4 Counter Behavior Type** 00 = Up counter type (supports in capture mode). 01 = Down count type (supports in capture mode). 10 = Up-down counter type. 11 = Reserved. Do not use.
[7:6]	**CNTTYPE3**	**PWM Channel 3 Counter Behavior Type** 00 = Up counter type (supports in capture mode). 01 = Down count type (supports in capture mode). 10 = Up-down counter type. 11 = Reserved. Do not use.
[5:4]	**CNTTYPE2**	**PWM Channel 2 Counter Behavior Type** 00 = Up counter type (supports in capture mode). 01 = Down count type (supports in capture mode). 10 = Up-down counter type. 11 = Reserved. Do not use.
[3:2]	**CNTTYPE1**	**PWM Channel 1 Counter Behavior Type** 00 = Up counter type (supports in capture mode). 01 = Down count type (supports in capture mode). 10 = Up-down counter type. 11 = Reserved. Do not use.
[1:0]	**CNTTYPE0**	**PWM Channel 0 Counter Behavior Type** 00 = Up counter type (supports in capture mode). 01 = Down count type (supports in capture mode). 10 = Up-down counter type. 11 = Reserved. Do not use.

PWM Synchronization Register (PWM_SYNC)

Register	Offset	R/W	Description	Reset Value
PWM_SYNC	PWM0_BA+0x08	R/W	PWM Synchronization Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					PHSDIR4	PHSDIR2	PHSDIR0
23	22	21	20	19	18	17	16
SINPINV	SFLTCNT			SFLTCSEL			SNFLTEN
15	14	13	12	11	10	9	8
Reserved		SINSRC4		SINSRC2		SINSRC0	
7	6	5	4	3	2	1	0
Reserved					PHSEN4	PHSEN2	PHSEN0

Bits	Description	
[31:27]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[26]	**PHSDIR4**	**PWM Channel 4 Phase Direction Control** 0 = Control PWM counter count decrement after synchronizing. 1 = Control PWM counter count increment after synchronizing.
[25]	**PHSDIR2**	**PWM Channel 2 Phase Direction Control** 0 = Control PWM counter count decrement after synchronizing. 1 = Control PWM counter count increment after synchronizing.
[24]	**PHSDIR0**	**PWM Channel 0 Phase Direction Control** 0 = Control PWM counter count decrement after synchronizing. 1 = Control PWM counter count increment after synchronizing.
[23]	**SINPINV**	**SYNC Input Pin Inverse** 0 = The state of pin SYNC is passed to the negative edge detector. 1 = The inversed state of pin SYNC is passed to the negative edge detector.
[22:20]	**SFLTCNT**	**SYNC Edge Detector Filter Count** The register bits control the counter number of edge detector.
[19:17]	**SFLTCSEL**	**SYNC 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.

[16]	**SNFLTEN**	**PWM0_SYNC_IN Noise Filter Enable Bits** 0 = Noise filter of input pin PWM0_SYNC_IN is Disabled. 1 = Noise filter of input pin PWM0_SYNC_IN is Enabled.
[15:14]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[13:12]	**SINSRC4**	**PWM Channel 4 PWM0_SYNC_IN Source Selection** 00 = Synchronize source from SYNC_IN or SWSYNC. 01 = Counter equal to 0. 10 = Counter equal to PWM_CMPDATm, m denotes 1, 3, 5. 11 = SYNC_OUT will not be generated.
[11:10]	**SINSRC2**	**PWM Channel 2 PWM0_SYNC_IN Source Selection** 00 = Synchronize source from SYNC_IN or SWSYNC. 01 = Counter equal to 0. 10 = Counter equal to PWM_CMPDATm, m denotes 1, 3, 5. 11 = SYNC_OUT will not be generated.
[9:8]	**SINSRC0**	**PWM Channel 0 PWM0_SYNC_IN Source Selection** 00 = Synchronize source from SYNC_IN or SWSYNC. 01 = Counter equal to 0. 10 = Counter equal to PWM_CMPDATm, m denotes 1, 3, 5. 11 = SYNC_OUT will not be generated.
[7:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**PHSEN4**	**PWM Channel 4 SYNC Phase Enable Bits** 0 = PWM counter disable to load PHS value. 1 = PWM counter enable to load PHS value.
[1]	**PHSEN2**	**PWM Channel 2 SYNC Phase Enable Bits** 0 = PWM counter disable to load PHS value. 1 = PWM counter enable to load PHS value.
[0]	**PHSEN0**	**PWM Channel 0 SYNC Phase Enable Bits** 0 = PWM counter disable to load PHS value. 1 = PWM counter enable to load PHS value.

PWM Software Control Synchronization Register (PWM_SWSYNC)

Register	Offset	R/W	Description	Reset Value
PWM_SWSYNC	PWM0_BA+0x0C	R/W	PWM Software Control Synchronization Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**SWSYNC4**	**SWSYNC2**	**SWSYNC0**

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**SWSYNC4**	**PWM Channel 4 Software SYNC Function** When SINSRC4 (PWM_SYNC[13:12]) is selected to 0, SYNC_OUT source is come from SYNC_IN or this bit.
[1]	**SWSYNC2**	**PWM Channel 2 Software SYNC Function** When SINSRC2 (PWM_SYNC[11:10]) is selected to 0, SYNC_OUT source is come from SYNC_IN or this bit.
[0]	**SWSYNC0**	**PWM Channel 0 Software SYNC Function** When SINSRC0 (PWM_SYNC[9:8]) is selected to 0, SYNC_OUT source is come from SYNC_IN or this bit.

PWM Clock Source Register (PWM_CLKSRC)

Register	Offset	R/W	Description	Reset Value
PWM_CLKSRC	PWM0_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. Any values read should be ignored. When writing to this field always write with reset value.
[18:16]	**ECLKSRC4**	**PWM_CH45 External Clock Source Select** 000 = PWM0_CLK. 001 = TIMER0 overflow. 010 = TIMER1 overflow. 011 = TIMER2 overflow. 100 = TIMER3 overflow. Others = Reserved. Do not use.
[15:11]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[10:8]	**ECLKSRC2**	**PWM_CH23 External Clock Source Select** 000 = PWM0_CLK. 001 = TIMER0 overflow. 010 = TIMER1 overflow. 011 = TIMER2 overflow. 100 = TIMER3 overflow. Others = Reserved. Do not use.
[7:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**ECLKSRC0**	**PWM_CH01 External Clock Source Select** 000 = PWM0_CLK. 001 = TIMER0 overflow. 010 = TIMER1 overflow. 011 = TIMER2 overflow. 100 = TIMER3 overflow. Others = Reserved. Do not use.

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	PWM0_BA+0x14	R/W	PWM Clock Pre-scale Register 0/1	0x0000_0000
PWM_CLKPSC2_3	PWM0_BA+0x18	R/W	PWM Clock Pre-scale Register 2/3	0x0000_0000
PWM_CLKPSC4_5	PWM0_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. Any values read should be ignored. When writing to this field always write with reset value.
[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	PWM0_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		CNTEN5	CNTEN4	CNTEN3	CNTEN2	CNTEN1	CNTEN0

Bits	Description	
[31:6]	Reserved	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5]	CNTEN5	**PWM Channel 5 Counter Enable Bits** 0 = PWM Counter and clock prescaler Stop Running. 1 = PWM Counter and clock prescaler Start Running.
[4]	CNTEN4	**PWM Channel 4 Counter Enable Bits** 0 = PWM Counter and clock prescaler Stop Running. 1 = PWM Counter and clock prescaler Start Running.
[3]	CNTEN3	**PWM Channel 3 Counter Enable Bits** 0 = PWM Counter and clock prescaler Stop Running. 1 = PWM Counter and clock prescaler Start Running.
[2]	CNTEN2	**PWM Channel 2 Counter Enable Bits** 0 = PWM Counter and clock prescaler Stop Running. 1 = PWM Counter and clock prescaler Start Running.
[1]	CNTEN1	**PWM Channel 1 Counter Enable Bits** 0 = PWM Counter and clock prescaler Stop Running. 1 = PWM Counter and clock prescaler Start Running.
[0]	CNTEN0	**PWM Channel 0 Counter Enable Bits** 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	PWM0_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		**CNTCLR5**	**CNTCLR4**	**CNTCLR3**	**CNTCLR2**	**CNTCLR1**	**CNTCLR0**

Bits	Description	
[31:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5]	**CNTCLR5**	**PWM Channel 5 Clear PWM Counter Control Bit** It is automatically cleared by hardware. 0 = No effect. 1 = Clear 16-bit PWM counter to 0000H.
[4]	**CNTCLR4**	**PWM Channel 4 Clear PWM Counter Control Bit** It is automatically cleared by hardware. 0 = No effect. 1 = Clear 16-bit PWM counter to 0000H.
[3]	**CNTCLR3**	**PWM Channel 3 Clear PWM Counter Control Bit** It is automatically cleared by hardware. 0 = No effect. 1 = Clear 16-bit PWM counter to 0000H.
[2]	**CNTCLR2**	**PWM Channel 2 Clear PWM Counter Control Bit** It is automatically cleared by hardware. 0 = No effect. 1 = Clear 16-bit PWM counter to 0000H.
[1]	**CNTCLR1**	**PWM Channel 1 Clear PWM Counter Control Bit** It is automatically cleared by hardware. 0 = No effect. 1 = Clear 16-bit PWM counter to 0000H.
[0]	**CNTCLR0**	**PWM Channel 0 Clear PWM Counter Control Bit** It is automatically cleared by hardware. 0 = No effect. 1 = Clear 16-bit PWM counter to 0000H.

PWM Load Register (PWM_LOAD)

Register	Offset	R/W	Description	Reset Value
PWM_LOAD	PWM0_BA+0x28	R/W	PWM Load Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		LOAD5	LOAD4	LOAD3	LOAD2	LOAD1	LOAD0

Bits	Description	
[31:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5]	**LOAD5**	**PWM Channel 5 Re-load PWM Comparator Register (CMPDAT) Control Bit** This bit is software write, hardware clear when current PWM period end. Write Operation: 0 = No effect. 1 = Set load window of window loading mode. Read Operation: 0 = No load window is set. 1 = Load window is set. **Note:** This bit only use in window loading mode, WINLDEN5(PWM_CTL0[13]) = 1.
[4]	**LOAD4**	**PWM Channel 4 Re-load PWM Comparator Register (CMPDAT) Control Bit** This bit is software write, hardware clear when current PWM period end. Write Operation: 0 = No effect. 1 = Set load window of window loading mode. Read Operation: 0 = No load window is set. 1 = Load window is set. **Note:** This bit only use in window loading mode, WINLDEN4(PWM_CTL0[12]) = 1.
[3]	**LOAD3**	**PWM Channel 3 Re-load PWM Comparator Register (CMPDAT) Control Bit** This bit is software write, hardware clear when current PWM period end. Write Operation: 0 = No effect. 1 = Set load window of window loading mode. Read Operation: 0 = No load window is set.

		1 = Load window is set. **Note:** This bit only use in window loading mode, WINLDEN3(PWM_CTL0[11]) = 1.
[2]	**LOAD2**	**PWM Channel 2 Re-load PWM Comparator Register (CMPDAT) Control Bit** This bit is software write, hardware clear when current PWM period end. Write Operation: 0 = No effect. 1 = Set load window of window loading mode. Read Operation: 0 = No load window is set. 1 = Load window is set. **Note:** This bit only use in window loading mode, WINLDEN2(PWM_CTL0[10]) = 1.
[1]	**LOAD1**	**PWM Channel 1 Re-load PWM Comparator Register (CMPDAT) Control Bit** This bit is software write, hardware clear when current PWM period end. Write Operation: 0 = No effect. 1 = Set load window of window loading mode. Read Operation: 0 = No load window is set. 1 = Load window is set. **Note:** This bit only use in window loading mode, WINLDEN1(PWM_CTL0[9]) = 1.
[0]	**LOAD0**	**PWM Channel 0 Re-load PWM Comparator Register (CMPDAT) Control Bit** This bit is software write, hardware clear when current PWM period end. Write Operation: 0 = No effect. 1 = Set load window of window loading mode. Read Operation: 0 = No load window is set. 1 = Load window is set. **Note:** This bit only use in window loading mode, WINLDEN0(PWM_CTL0[8]) = 1.

PWM Period Register 0~5 (PWM_PERIOD0~5)

Register	Offset	R/W	Description	Reset Value
PWM_PERIOD0	PWM0_BA+0x30	R/W	PWM Period Register 0	0x0000_0000
PWM_PERIOD1	PWM0_BA+0x34	R/W	PWM Period Register 1	0x0000_0000
PWM_PERIOD2	PWM0_BA+0x38	R/W	PWM Period Register 2	0x0000_0000
PWM_PERIOD3	PWM0_BA+0x3C	R/W	PWM Period Register 3	0x0000_0000
PWM_PERIOD4	PWM0_BA+0x40	R/W	PWM Period Register 4	0x0000_0000
PWM_PERIOD5	PWM0_BA+0x44	R/W	PWM Period 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**
PERIOD							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PERIOD							

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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	PWM0_BA+0x50	R/W	PWM Comparator Register 0	0x0000_0000
PWM_CMPDAT1	PWM0_BA+0x54	R/W	PWM Comparator Register 1	0x0000_0000
PWM_CMPDAT2	PWM0_BA+0x58	R/W	PWM Comparator Register 2	0x0000_0000
PWM_CMPDAT3	PWM0_BA+0x5C	R/W	PWM Comparator Register 3	0x0000_0000
PWM_CMPDAT4	PWM0_BA+0x60	R/W	PWM Comparator Register 4	0x0000_0000
PWM_CMPDAT5	PWM0_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. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**CMP**	**PWM Comparator Register** CMP use to compare with CNTR to generate PWM waveform, interrupt and trigger EADC. In independent mode, CMPDAT0~5 denote as 6 independent PWM_CH0~5 compared point. In complementary mode, CMPDAT0, 2, 4 denote as first compared point, and 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	PWM0_BA+0x70	R/W	PWM Dead-Time Control Register 0/1	0x0000_0000
PWM_DTCTL2_3	PWM0_BA+0x74	R/W	PWM Dead-Time Control Register 2/3	0x0000_0000
PWM_DTCTL4_5	PWM0_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. Any values read should be ignored. When writing to this field always write with reset value.
[24]	**DTCKSEL**	**Dead-time Clock Select (Write Protected)** 0 = Dead-time clock source from PWM_CLK. 1 = Dead-time clock source from prescaler output. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[23:17]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[16]	**DTEN**	**Enable Dead-time Insertion for PWM Pair (PWM_CH0, PWM_CH1) (PWM_CH2, PWM_CH3) (PWM_CH4, PWM_CH5) (Write Protected)** 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 register is write protected. Refer to SYS_REGLCTL register.
[15:12]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[11:0]	**DTCNT**	**Dead-time Counter (Write Protected)** The dead-time can be calculated from the following formula: Dead-time = (DTCNT[11:0]+1) * PWM_CLK period. **Note:** This register is write protected. Refer to SYS_REGLCTL register.

PWM Counter Phase Register 0_1, 2_3, 4_5 (PWM_PHS0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
PWM_PHS0_1	PWM0_BA+0x80	R/W	PWM Counter Phase Register 0/1	0x0000_0000
PWM_PHS2_3	PWM0_BA+0x84	R/W	PWM Counter Phase Register 2/3	0x0000_0000
PWM_PHS4_5	PWM0_BA+0x88	R/W	PWM Counter Phase 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**
PHS							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PHS							

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**PHS**	**PWM Synchronous Start Phase Bits** PHS determines the PWM synchronous start phase value. These bits only use in synchronous function.

PWM Counter Register 0~5 (PWM_CNT0~5)

Register	Offset	R/W	Description	Reset Value
PWM_CNT0	PWM0_BA+0x90	R	PWM Counter Register 0	0x0000_0000
PWM_CNT1	PWM0_BA+0x94	R	PWM Counter Register 1	0x0000_0000
PWM_CNT2	PWM0_BA+0x98	R	PWM Counter Register 2	0x0000_0000
PWM_CNT3	PWM0_BA+0x9C	R	PWM Counter Register 3	0x0000_0000
PWM_CNT4	PWM0_BA+0xA0	R	PWM Counter Register 4	0x0000_0000
PWM_CNT5	PWM0_BA+0xA4	R	PWM Counter Register 5	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. Any values read should be ignored. When writing to this field always write with reset value.
[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 CNTR 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	PWM0_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. Any values read should be ignored. When writing to this field always write with reset value.
[27:26]	**PRDPCTL5**	**PWM Channel 5 Period (Center) Point Control** 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.
[25:24]	**PRDPCTL4**	**PWM Channel 4 Period (Center) Point Control** 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.
[23:22]	**PRDPCTL3**	**PWM Channel 3 Period (Center) Point Control** 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.
[21:20]	**PRDPCTL2**	**PWM Channel 2 Period (Center) Point Control**

		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.
[19:18]	**PRDPCTL1**	**PWM Channel 1 Period (Center) Point Control** 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.
[17:16]	**PRDPCTL0**	**PWM Channel 0 Period (Center) Point Control** 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. Any values read should be ignored. When writing to this field always write with reset value.
[11:10]	**ZPCTL5**	**PWM Channel 5 Zero Point Control** 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.
[9:8]	**ZPCTL4**	**PWM Channel 4 Zero Point Control** 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.
[7:6]	**ZPCTL3**	**PWM Channel 3 Zero Point Control** 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.
[5:4]	**ZPCTL2**	**PWM Channel 2 Zero Point Control** 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.
[3:2]	**ZPCTL1**	**PWM Channel 1 Zero Point Control** 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.
[1:0]	**ZPCTL0**	**PWM Channel 0 Zero Point Control** 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	PWM0_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. Any values read should be ignored. When writing to this field always write with reset value.
[27:26]	**CMPDCTL5**	**PWM Channel 5 Compare Down Point Control** 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.
[25:24]	**CMPDCTL4**	**PWM Channel 4 Compare Down Point Control** 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.
[23:22]	**CMPDCTL3**	**PWM Channel 3 Compare Down Point Control** 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.
[21:20]	**CMPDCTL2**	**PWM Channel 2 Compare Down Point Control**

		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.
[19:18]	**CMPDCTL1**	**PWM Channel 1 Compare Down Point Control** 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.
[17:16]	**CMPDCTL0**	**PWM Channel 0 Compare Down Point Control** 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. Any values read should be ignored. When writing to this field always write with reset value.
[11:10]	**CMPUCTL5**	**PWM Channel 5 Compare Up Point Control** 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.
[9:8]	**CMPUCTL4**	**PWM Channel 4 Compare Up Point Control** 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.
[7:6]	**CMPUCTL3**	**PWM Channel 3 Compare Up Point Control** 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.

[5:4]	**CMPUCTL2**	**PWM Channel 2 Compare Up Point Control** 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.
[3:2]	**CMPUCTL1**	**PWM Channel 1 Compare Up Point Control** 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.
[1:0]	**CMPUCTL0**	**PWM Channel 0 Compare Up Point Control** 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	PWM0_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. Any values read should be ignored. When writing to this field always write with reset value.
[5]	**MSKEN5**	**PWM Channel 5 Mask Enable Bits** The PWM output signal will be masked when this bit is enabled and output MSKDAT5 (PWM_MSK[5]) data. 0 = PWM output signal is non-masked. 1 = PWM output signal is masked and output MSKDAT5 (PWM_MSK[5]) data.
[4]	**MSKEN4**	**PWM Channel 4 Mask Enable Bits** The PWM output signal will be masked when this bit is enabled and output MSKDAT4 (PWM_MSK[4]) data. 0 = PWM output signal is non-masked. 1 = PWM output signal is masked and output MSKDAT4 (PWM_MSK[4]) data.
[3]	**MSKEN3**	**PWM Channel 3 Mask Enable Bits** The PWM output signal will be masked when this bit is enabled and output MSKDAT3 (PWM_MSK[3]) data. 0 = PWM output signal is non-masked. 1 = PWM output signal is masked and output MSKDAT3 (PWM_MSK[3]) data.
[2]	**MSKEN2**	**PWM Channel 2 Mask Enable Bits** The PWM output signal will be masked when this bit is enabled and output MSKDAT2 (PWM_MSK[2]) data. 0 = PWM output signal is non-masked. 1 = PWM output signal is masked and output MSKDAT2 (PWM_MSK[2]) data.
[1]	**MSKEN1**	**PWM Channel 1 Mask Enable Bits** The PWM output signal will be masked when this bit is enabled and output MSKDAT1 (PWM_MSK[1]) data. 0 = PWM output signal is non-masked. 1 = PWM output signal is masked and output MSKDAT1 (PWM_MSK[1]) data.

[0]	**MSKEN0**	**PWM Channel 0 Mask Enable Bits** The PWM output signal will be masked when this bit is enabled. 0 = PWM output signal is non-masked. 1 = PWM output signal is masked and output MSKDAT0 (PWM_MSK[0]) data.

PWM Mask DATA Register (PWM_MSK)

Register	Offset	R/W	Description	Reset Value
PWM_MSK	PWM0_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. Any values read should be ignored. When writing to this field always write with reset value.
[5]	**MSKDAT5**	**PWM Channel 5 Mask Data Bit** This bit control the state of output pin, if MSKEN5 (PWM_MSKEN[5]) is enabled. 0 = Output logic low to PWM5. 1 = Output logic high to PWM5.
[4]	**MSKDAT4**	**PWM Channel 4 Mask Data Bit** This bit control the state of output pin, if MSKEN4 (PWM_MSKEN[4]) is enabled. 0 = Output logic low to PWM4. 1 = Output logic high to PWM4.
[3]	**MSKDAT3**	**PWM Channel 3 Mask Data Bit** This bit control the state of output pin, if MSKEN3 (PWM_MSKEN[3]) is enabled. 0 = Output logic low to PWM3. 1 = Output logic high to PWM3.
[2]	**MSKDAT2**	**PWM Channel 2 Mask Data Bit** This bit control the state of output pin, if MSKEN2 (PWM_MSKEN[2]) is enabled. 0 = Output logic low to PWM2. 1 = Output logic high to PWM2.
[1]	**MSKDAT1**	**PWM Channel 1 Mask Data Bit** This bit control the state of output pin, if MSKEN1 (PWM_MSKEN[1]) is enabled. 0 = Output logic low to PWM1. 1 = Output logic high to PWM1.
[0]	**MSKDAT0**	**PWM Channel 0 Mask Data Bit** This bit control the state of output pin, if MSKEN0 (PWM_MSKEN[0]) is enabled. 0 = Output logic low to PWM0. 1 = Output logic high to PWM0.

PWM Brake Noise Filter Register (PWM_BNF)

Register	Offset	R/W	Description	Reset Value
PWM_BNF	PWM0_BA+0xC0	R/W	PWM Brake Noise Filter Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
BRK1PINV	BRK1FCNT			BRK1NFSEL			BRK1NFEN
7	6	5	4	3	2	1	0
BRK0PINV	BRK0FCNT			BRK0NFSEL			BRK0NFEN

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15]	**BRK1PINV**	**Brake 1 Pin Inverse** 0 = The state of pin PWM0_BRAKE1 is passed to the negative edge detector. 1 = The inversed state of pin PWM0_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]	**BRK1NFSEL**	**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]	**BRK1NFEN**	**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 PWM0_BRAKE0 is passed to the negative edge detector. 1 = The inversed state of pin PWM0_BRAKE1 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]	**BRK0NFSEL**	**Brake 0 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.
[0]	**BRK0NFEN**	**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	PWM0_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**	**RAMBRKEN**	**BODBRKEN**	**CSSBRKEN**

Bits	Description	
[31:4]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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]	**RAMBRKEN**	**SRAM Parity Error Detection Trigger PWM Brake Function 0 Enable Bit** 0 = Brake Function triggered by SRAM parity error detection Disabled. 1 = Brake Function triggered by SRAM parity error detection Enabled.
[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.

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	PWM0_BA+0xC8	R/W	PWM Brake Edge Detect Control Register 0/1	0x0000_0000
PWM_BRKCTL2_3	PWM0_BA+0xCC	R/W	PWM Brake Edge Detect Control Register 2/3	0x0000_0000
PWM_BRKCTL4_5	PWM0_BA+0xD0	R/W	PWM Brake Edge Detect Control Register 4/5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			EADCLBEN	Reserved			
23	22	21	20	19	18	17	16
Reserved			EADCEBEN	BRKAODD		BRKAEVEN	
15	14	13	12	11	10	9	8
SYSLBEN	Reserved	BRKP1LEN	BRKP0LEN	Reserved			
7	6	5	4	3	2	1	0
SYSEBEN	Reserved	BRKP1EEN	BRKP0EEN	Reserved			

Bits	Description	
[31:29]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[28]	**EADCLBEN**	**Enable EADC Result Monitor (EADCRM) As Level-detect Brake Source (Write Protected)** 0 = EADCRM as level-detect brake source Disabled. 1 = EADCRM as level-detect brake source Enabled. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[27:21]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[20]	**EADCEBEN**	**Enable ADC Result Monitor (EADCRM) As Edge-detect Brake Source (Write Protected)** 0 = EADCRM as edge-detect brake source Disabled. 1 = EADCRM as edge-detect brake source Enabled. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[19:18]	**BRKAODD**	**PWM Brake Action Select for Odd Channel (Write Protected)** 00 = PWM0 brake event will not affect odd channels output. 01 = PWM odd channel output tri-state when PWM0 brake event happened. 10 = PWM odd channel output low level when PWM0 brake event happened. 11 = PWM odd channel output high level when PWM0 brake event happened. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[17:16]	**BRKAEVEN**	**PWM Brake Action Select for Even Channel (Write Protected)** 00 = PWM0 brake event will not affect even channels output. 01 = PWM even channel output tri-state when PWM0 brake event happened. 10 = PWM even channel output low level when PWM0 brake event happened.

		11 = PWM even channel output high level when PWM0 brake event happened. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[15]	**SYSLBEN**	**Enable System Fail As Level-detect Brake Source (Write Protected)** 0 = System Fail condition as level-detect brake source Disabled. 1 = System Fail condition as level-detect brake source Enabled. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[14]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[13]	**BRKP1LEN**	**Enable BKP1 Pin As Level-detect Brake Source (Write Protected)** 0 = PWM0_BRAKE1 pin as level-detect brake source Disabled. 1 = PWM0_BRAKE1 pin as level-detect brake source Enabled. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[12]	**BRKP0LEN**	**Enable BKP0 Pin As Level-detect Brake Source (Write Protected)** 0 = PWM0_BRAKE0 pin as level-detect brake source Disabled. 1 = PWM0_BRAKE0 pin as level-detect brake source Enabled. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[11:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7]	**SYSEBEN**	**Enable System Fail As Edge-detect Brake Source (Write Protected)** 0 = System Fail condition as edge-detect brake source Disabled. 1 = System Fail condition as edge-detect brake source Enabled. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5]	**BRKP1EEN**	**Enable PWM0_BRAKE1 Pin As Edge-detect Brake Source (Write Protected)** 0 = PWM0_BRAKE1 pin as edge-detect brake source Disabled. 1 = PWM0_BRAKE1 pin as edge-detect brake source Enabled. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[4]	**BRKP0EEN**	**Enable PWM0_BRAKE0 Pin As Edge-detect Brake Source (Write Protected)** 0 = PWM0_BRAKE0 pin as edge-detect brake source Disabled. 1 = PWM0_BRAKE0 pin as edge-detect brake source Enabled. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[3:0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

PWM Pin Polar Inverse Control (PWM_POLCTL)

Register	Offset	R/W	Description	Reset Value
PWM_POLCTL	PWM0_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. Any values read should be ignored. When writing to this field always write with reset value.
[5]	**PINV5**	**PWM Channel 5 PIN Polar Inverse Control** The register controls polarity state of PWM output. 0 = PWM output polar inverse Disabled. 1 = PWM output polar inverse Enabled.
[4]	**PINV4**	**PWM Channel 4 PIN Polar Inverse Control** The register controls polarity state of PWM output. 0 = PWM output polar inverse Disabled. 1 = PWM output polar inverse Enabled.
[3]	**PINV3**	**PWM Channel 3 PIN Polar Inverse Control** The register controls polarity state of PWM output. 0 = PWM output polar inverse Disabled. 1 = PWM output polar inverse Enabled.
[2]	**PINV2**	**PWM Channel 2 PIN Polar Inverse Control** The register controls polarity state of PWM output. 0 = PWM output polar inverse Disabled. 1 = PWM output polar inverse Enabled.
[1]	**PINV1**	**PWM Channel 1 PIN Polar Inverse Control** The register controls polarity state of PWM output. 0 = PWM output polar inverse Disabled. 1 = PWM output polar inverse Enabled.
[0]	**PINV0**	**PWM Channel 0 PIN Polar Inverse Control** The register controls polarity state of PWM output. 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	PWM0_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. Any values read should be ignored. When writing to this field always write with reset value.
[5]	**POEN5**	**PWM Channel 5 Pin Output Enable Bits** 0 = PWM pin at tri-state. 1 = PWM pin in output mode.
[4]	**POEN4**	**PWM Channel 4 Pin Output Enable Bits** 0 = PWM pin at tri-state. 1 = PWM pin in output mode.
[3]	**POEN3**	**PWM Channel 3 Pin Output Enable Bits** 0 = PWM pin at tri-state. 1 = PWM pin in output mode.
[2]	**POEN2**	**PWM Channel 2 Pin Output Enable Bits** 0 = PWM pin at tri-state. 1 = PWM pin in output mode.
[1]	**POEN1**	**PWM Channel 1 Pin Output Enable Bits** 0 = PWM pin at tri-state. 1 = PWM pin in output mode.
[0]	**POEN0**	**PWM Channel 0 Pin Output Enable Bits** 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	PWM0_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. Any values read should be ignored. When writing to this field always write with reset value.
[10]	**BRKLTRG4**	**PWM Pair 4 Level Brake Software Trigger (Write Only) (Write Protected)** Write 1 to this bit will trigger level brake, and set BRKLIF4 to 1 in PWM_INTSTS1 register. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[9]	**BRKLTRG2**	**PWM Pair 2 Level Brake Software Trigger (Write Only) (Write Protected)** Write 1 to this bit will trigger level brake, and set BRKLIF2 to 1 in PWM_INTSTS1 register. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[8]	**BRKLTRG0**	**PWM Pair 0 Level Brake Software Trigger (Write Only) (Write Protected)** Write 1 to this bit will trigger level brake, and set BRKLIF0 to 1 in PWM_INTSTS1 register. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[7:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**BRKETRG4**	**PWM Pair 4 Edge Brake Software Trigger (Write Only) (Write Protected)** Write 1 to this bit will trigger edge brake, and set BRKEIF4 to 1 in PWM_INTSTS1 register. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[1]	**BRKETRG2**	**PWM Pair 2 Edge Brake Software Trigger (Write Only) (Write Protected)** Write 1 to this bit will trigger edge brake, and set BRKEIF2 to 1 in PWM_INTSTS1 register. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[0]	**BRKETRG0**	**PWM Pair 0 Edge Brake Software Trigger (Write Only) (Write Protected)** Write 1 to this bit will trigger edge brake, and set BRKEIF0 to 1 in PWM_INTSTS1 register. **Note:** This register is write protected. Refer to SYS_REGLCTL register.

PWM Interrupt Enable Register 0 (PWM_INTEN0)

Register	Offset	R/W	Description	Reset Value
PWM_INTEN0	PWM0_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		PIEN5	PIEN4	PIEN3	PIEN2	PIEN1	PIEN0
7	6	5	4	3	2	1	0
Reserved		ZIEN5	ZIEN4	ZIEN3	ZIEN2	ZIEN1	ZIEN0

Bits	Description	
[31:30]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[29]	**CMPDIEN5**	**PWM Channel 5 Compare Down Count Interrupt Enable Bits** 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.
[28]	**CMPDIEN4**	**PWM Channel 4 Compare Down Count Interrupt Enable Bits** 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.
[27]	**CMPDIEN3**	**PWM Channel 3 Compare Down Count Interrupt Enable Bits** 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.
[26]	**CMPDIEN2**	**PWM Channel 2 Compare Down Count Interrupt Enable Bits** 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.
[25]	**CMPDIEN1**	**PWM Channel 1 Compare Down Count Interrupt Enable Bits** 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.

[24]	**CMPDIEN0**	**PWM Channel 0 Compare Down Count Interrupt Enable Bits** 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. Any values read should be ignored. When writing to this field always write with reset value.
[21]	**CMPUIEN5**	**PWM Channel 5 Compare Up Count Interrupt Enable Bits** 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.
[20]	**CMPUIEN4**	**PWM Channel 4 Compare Up Count Interrupt Enable Bits** 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.
[19]	**CMPUIEN3**	**PWM Channel 3 Compare Up Count Interrupt Enable Bits** 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.
[18]	**CMPUIEN2**	**PWM Channel 2 Compare Up Count Interrupt Enable Bits** 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.
[17]	**CMPUIEN1**	**PWM Channel 1 Compare Up Count Interrupt Enable Bits** 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.
[16]	**CMPUIEN0**	**PWM Channel 0 Compare Up Count Interrupt Enable Bits** 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.
[15:14]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[13]	**PIEN5**	**PWM Channel 5 Period Point Interrupt Enable Bits** 0 = Period point interrupt Disabled. 1 = Period point interrupt Enabled. **Note1:** When up-down counter type period point means center point. **Note2:** This channels will read always 0 at complementary mode.
[12]	**PIEN4**	**PWM Channel 4 Period Point Interrupt Enable Bits** 0 = Period point interrupt Disabled. 1 = Period point interrupt Enabled. **Note:** When up-down counter type period point means center point.

[11]	**PIEN3**	**PWM Channel 3 Period Point Interrupt Enable Bits** 0 = Period point interrupt Disabled. 1 = Period point interrupt Enabled. **Note1:** When up-down counter type period point means center point. **Note2:** This channels will read always 0 at complementary mode.
[10]	**PIEN2**	**PWM Channel 2 Period Point Interrupt Enable Bits** 0 = Period point interrupt Disabled. 1 = Period point interrupt Enabled. **Note:** When up-down counter type period point means center point.
[9]	**PIEN1**	**PWM Channel 1 Period Point Interrupt Enable Bits** 0 = Period point interrupt Disabled. 1 = Period point interrupt Enabled. **Note1:** When up-down counter type period point means center point. **Note2:** This channels will read always 0 at complementary mode.
[8]	**PIEN0**	**PWM Channel 0 Period Point Interrupt Enable Bits** 0 = Period point interrupt Disabled. 1 = Period point interrupt Enabled. **Note:** When up-down counter type period point means center point.
[7:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5]	**ZIEN5**	**PWM Channel 5 Zero Point Interrupt Enable Bits** 0 = Zero point interrupt Disabled. 1 = Zero point interrupt Enabled. **Note:** This channel will read always 0 at complementary mode.
[4]	**ZIEN4**	**PWM Channel 4 Zero Point Interrupt Enable Bits** 0 = Zero point interrupt Disabled. 1 = Zero point interrupt Enabled.
[3]	**ZIEN3**	**PWM Channel 3 Zero Point Interrupt Enable Bits** 0 = Zero point interrupt Disabled. 1 = Zero point interrupt Enabled. **Note:** This channel will read always 0 at complementary mode.
[2]	**ZIEN2**	**PWM Channel 2 Zero Point Interrupt Enable Bits** 0 = Zero point interrupt Disabled. 1 = Zero point interrupt Enabled.
[1]	**ZIEN1**	**PWM Channel 1 Zero Point Interrupt Enable Bits** 0 = Zero point interrupt Disabled. 1 = Zero point interrupt Enabled. **Note:** This channel will read always 0 at complementary mode.
[0]	**ZIEN0**	**PWM Channel 0 Zero Point Interrupt Enable Bits** 0 = Zero point interrupt Disabled. 1 = Zero point interrupt Enabled.

PWM Interrupt Enable Register 1 (PWM_INTEN1)

Register	Offset	R/W	Description	Reset Value
PWM_INTEN1	PWM0_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. Any values read should be ignored. When writing to this field always write with reset value.
[10]	**BRKLIEN4_5**	**PWM Level-detect Brake Interrupt Enable for Channel4/5 (Write Protected)** 0 = Level-detect Brake interrupt for channel4/5 Disabled. 1 = Level-detect Brake interrupt for channel4/5 Enabled. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[9]	**BRKLIEN2_3**	**PWM Level-detect Brake Interrupt Enable for Channel2/3 (Write Protected)** 0 = Level-detect Brake interrupt for channel2/3 Disabled. 1 = Level-detect Brake interrupt for channel2/3 Enabled. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[8]	**BRKLIEN0_1**	**PWM Level-detect Brake Interrupt Enable for Channel0/1 (Write Protected)** 0 = Level-detect Brake interrupt for channel0/1 Disabled. 1 = Level-detect Brake interrupt for channel0/1 Enabled. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[7:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**BRKEIEN4_5**	**PWM Edge-detect Brake Interrupt Enable for Channel4/5 (Write Protected)** 0 = Edge-detect Brake interrupt for channel4/5 Disabled. 1 = Edge-detect Brake interrupt for channel4/5 Enabled. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[1]	**BRKEIEN2_3**	**PWM Edge-detect Brake Interrupt Enable for Channel2/3 (Write Protected)** 0 = Edge-detect Brake interrupt for channel2/3 Disabled. 1 = Edge-detect Brake interrupt for channel2/3 Enabled. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[0]	**BRKEIEN0_1**	**PWM Edge-detect Brake Interrupt Enable for Channel0/1 (Write Protected)** 0 = Edge-detect Brake interrupt for channel0/1 Disabled.

		1 = Edge-detect Brake interrupt for channel0/1 Enabled. **Note:** This register is write protected. Refer to SYS_REGLCTL register.

PWM Interrupt Flag Register 0 (PWM_INTSTS0)

Register	Offset	R/W	Description	Reset Value
PWM_INTSTS0	PWM0_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		PIF5	PIF4	PIF3	PIF2	PIF1	PIF0
7	6	5	4	3	2	1	0
Reserved		ZIF5	ZIF4	ZIF3	ZIF2	ZIF1	ZIF0

Bits	Description	
[31:30]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[29]	**CMPDIF5**	**PWM Channel 4 Compare Down Count Interrupt Flag** Flag is set by hardware when PWM counter down count and reaches PWM_CMPDAT5, 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.
[28]	**CMPDIF4**	**PWM Channel 4 Compare Down Count Interrupt Flag** Flag is set by hardware when PWM counter down count and reaches PWM_CMPDAT4, 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.
[27]	**CMPDIF3**	**PWM Channel 3 Compare Down Count Interrupt Flag** Flag is set by hardware when PWM counter down count and reaches PWM_CMPDAT3, 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.
[26]	**CMPDIF2**	**PWM Channel 2 Compare Down Count Interrupt Flag** Flag is set by hardware when PWM counter down count and reaches PWM_CMPDAT2, 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.

[25]	**CMPDIF1**	**PWM Channel 1 Compare Down Count Interrupt Flag** Flag is set by hardware when PWM counter down count and reaches PWM_CMPDAT1, 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.
[24]	**CMPDIF0**	**PWM Channel 0 Compare Down Count Interrupt Flag** Flag is set by hardware when PWM counter down count and reaches PWM_CMPDAT0, 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. Any values read should be ignored. When writing to this field always write with reset value.
[21]	**CMPUIF5**	**PWM Channel 5 Compare Up Count Interrupt Flag** Flag is set by hardware when PWM counter up count and reaches PWM_CMPDAT5, software can clear this bit by writing 1 to it. **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.
[20]	**CMPUIF4**	**PWM Channel 4 Compare Up Count Interrupt Flag** Flag is set by hardware when PWM counter up count and reaches PWM_CMPDAT4, software can clear this bit by writing 1 to it. **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.
[19]	**CMPUIF3**	**PWM Channel 3 Compare Up Count Interrupt Flag** Flag is set by hardware when PWM counter up count and reaches PWM_CMPDAT3, software can clear this bit by writing 1 to it. **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.
[18]	**CMPUIF2**	**PWM Channel 2 Compare Up Count Interrupt Flag** Flag is set by hardware when PWM counter up count and reaches PWM_CMPDAT2, software can clear this bit by writing 1 to it. **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.
[17]	**CMPUIF1**	**PWM Channel 1 Compare Up Count Interrupt Flag** Flag is set by hardware when PWM counter up count and reaches PWM_CMPDAT1, software can clear this bit by writing 1 to it. **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.
[16]	**CMPUIF0**	**PWM Channel 0 Compare Up Count Interrupt Flag** Flag is set by hardware when PWM counter up count and reaches PWM_CMPDAT0, software can clear this bit by writing 1 to it. **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.
[15:14]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[13]	**PIF5**	**PWM Channel 5 Period Point Interrupt Flag** This bit is set by hardware when PWM counter reaches PWM_PERIOD5, software can write 1 to clear this bit to zero.
[12]	**PIF4**	**PWM Channel 4 Period Point Interrupt Flag** This bit is set by hardware when PWM counter reaches PWM_PERIOD4, software can write 1 to clear this bit to zero.
[11]	**PIF3**	**PWM Channel 3 Period Point Interrupt Flag** This bit is set by hardware when PWM counter reaches PWM_PERIOD3, software can write 1 to clear this bit to zero.
[10]	**PIF2**	**PWM Channel 2 Period Point Interrupt Flag** This bit is set by hardware when PWM counter reaches PWM_PERIOD2, software can write 1 to clear this bit to zero.
[9]	**PIF1**	**PWM Channel 1 Period Point Interrupt Flag** This bit is set by hardware when PWM counter reaches PWM_PERIOD1, software can write 1 to clear this bit to zero.
[8]	**PIF0**	**PWM Channel 0 Period Point Interrupt Flag** This bit is set by hardware when PWM counter reaches PWM_PERIOD0, software can write 1 to clear this bit to zero..
[7:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5]	**ZIF5**	**PWM Channel 5 Zero Point Interrupt Flag** This bit is set by hardware when PWM counter reaches zero, software can write 1 to clear this bit to zero.
[4]	**ZIF4**	**PWM Channel 4 Zero Point Interrupt Flag** This bit is set by hardware when PWM counter reaches zero, software can write 1 to clear this bit to zero.
[3]	**ZIF3**	**PWM Channel 3 Zero Point Interrupt Flag** This bit is set by hardware when PWM counter reaches zero, software can write 1 to clear this bit to zero.
[2]	**ZIF2**	**PWM Channel 2 Zero Point Interrupt Flag** This bit is set by hardware when PWM counter reaches zero, software can write 1 to clear this bit to zero.
[1]	**ZIF1**	**PWM Channel 1 Zero Point Interrupt Flag** This bit is set by hardware when PWM counter reaches zero, software can write 1 to clear this bit to zero.
[0]	**ZIF0**	**PWM Channel 0 Zero Point Interrupt Flag** This bit is set by hardware when PWM 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	PWM0_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. Any values read should be ignored. When writing to this field always write with reset value.
[29]	**BRKLSTS5**	**PWM Channel 5 Level-detect Brake Status (Read Only)** 0 = PWM channel 5 level-detect brake state is released. 1 = When PWM channel 5 level-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel 5 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 Channel 4 Level-detect Brake Status (Read Only)** 0 = PWM channel 4 level-detect brake state is released. 1 = When PWM channel 4 level-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel 4 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 Channel 3 Level-detect Brake Status (Read Only)** 0 = PWM channel 3 level-detect brake state is released. 1 = When PWM channel 3 level-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel 3 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 Channel 2 Level-detect Brake Status (Read Only)** 0 = PWM channel 2 level-detect brake state is released. 1 = When PWM channel 2 level-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel 2 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 Channel 1 Level-detect Brake Status (Read Only)** 0 = PWM channel 1 level-detect brake state is released. 1 = When PWM channel 1 level-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel 1 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 Channel 0 Level-detect Brake Status (Read Only)** 0 = PWM channel 0 level-detect brake state is released. 1 = When PWM channel 0 level-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel 0 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. Any values read should be ignored. When writing to this field always write with reset value.
[21]	**BRKESTS5**	**PWM Channel 5 Edge-detect Brake Status** 0 = PWM channel 5 edge-detect brake state is released. 1 = When PWM channel 5 edge-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel 5 at brake state, writing 1 to clear.
[20]	**BRKESTS4**	**PWM Channel 4 Edge-detect Brake Status** 0 = PWM channel 4 edge-detect brake state is released. 1 = When PWM channel 4 edge-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel 4 at brake state, writing 1 to clear.
[19]	**BRKESTS3**	**PWM Channel 3 Edge-detect Brake Status** 0 = PWM channel 3 edge-detect brake state is released. 1 = When PWM channel 3 edge-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel 3 at brake state, writing 1 to clear.
[18]	**BRKESTS2**	**PWM Channel 2 Edge-detect Brake Status** 0 = PWM channel 2 edge-detect brake state is released. 1 = When PWM channel 2 edge-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel 2 at brake state, writing 1 to clear.
[17]	**BRKESTS1**	**PWM Channel 1 Edge-detect Brake Status** 0 = PWM channel 1 edge-detect brake state is released. 1 = When PWM channel 1 edge-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel 1 at brake state, writing 1 to clear.
[16]	**BRKESTS0**	**PWM Channel 0 Edge-detect Brake Status** 0 = PWM channel 0 edge-detect brake state is released. 1 = When PWM channel 0 edge-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel 0 at brake state, writing 1 to clear.
[15:14]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[13]	**BRKLIF5**	**PWM Channel 5 Level-detect Brake Interrupt Flag (Write Protected)** 0 = PWM channel 5 level-detect brake event do not happened. 1 = When PWM channel 5 level-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[12]	**BRKLIF4**	**PWM Channel 4 Level-detect Brake Interrupt Flag (Write Protected)** 0 = PWM channel 4 level-detect brake event do not happened.

		1 = When PWM channel 4 level-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[11]	**BRKLIF3**	**PWM Channel 3 Level-detect Brake Interrupt Flag (Write Protected)** 0 = PWM channel 3 level-detect brake event do not happened. 1 = When PWM channel 3 level-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[10]	**BRKLIF2**	**PWM Channel 2 Level-detect Brake Interrupt Flag (Write Protected)** 0 = PWM channel 2 level-detect brake event do not happened. 1 = When PWM channel 2 level-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[9]	**BRKLIF1**	**PWM Channel 1 Level-detect Brake Interrupt Flag (Write Protected)** 0 = PWM channel 1 level-detect brake event do not happened. 1 = When PWM channel 1 level-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[8]	**BRKLIF0**	**PWM Channel 0 Level-detect Brake Interrupt Flag (Write Protected)** 0 = PWM channel 0 level-detect brake event do not happened. 1 = When PWM channel 0 level-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[7:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5]	**BRKEIF5**	**PWM Channel 5 Edge-detect Brake Interrupt Flag (Write Protected)** 0 = PWM channel 5 edge-detect brake event do not happened. 1 = When PWM channel 5 edge-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[4]	**BRKEIF4**	**PWM Channel 4 Edge-detect Brake Interrupt Flag (Write Protected)** 0 = PWM channel 4 edge-detect brake event do not happened. 1 = When PWM channel 4 edge-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[3]	**BRKEIF3**	**PWM Channel 3 Edge-detect Brake Interrupt Flag (Write Protected)** 0 = PWM channel 3 edge-detect brake event do not happened. 1 = When PWM channel 3 edge-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[2]	**BRKEIF2**	**PWM Channel 2 Edge-detect Brake Interrupt Flag (Write Protected)** 0 = PWM channel 2 edge-detect brake event do not happened. 1 = When PWM channel 2 edge-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This register is write protected. Refer to SYS_REGLCTL register.
[1]	**BRKEIF1**	**PWM Channel 1 Edge-detect Brake Interrupt Flag (Write Protected)** 0 = PWM channel 1 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 register is write protected. Refer to SYS_REGLCTL register.
[0]	**BRKEIF0**	**PWM Channel 0 Edge-detect Brake Interrupt Flag (Write Protected)** 0 = PWM channel 0 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 register is write protected. Refer to SYS_REGLCTL register.

PWM Trigger EADC Source Select Register 0 (PWM_EADCTS0)

Register	Offset	R/W	Description	Reset Value
PWM_EADCTS0	PWM0_BA+0xF8	R/W	PWM Trigger EADC 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 EADC enable bit
[30:28]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[27:24]	**TRGSEL3**	**PWM_CH3 Trigger EADC 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 = PWM_CH3 zero point. 0110 = PWM_CH3 period point. 0111 = PWM_CH3 zero or period point. 1000 = PWM_CH3 up-count CMPDAT point. 1001 = PWM_CH3 down-count CMPDAT point. 1010 = PWM_CH0 up-count free CMPDAT point. 1011 = PWM_CH0 down-count free CMPDAT point. 1100 = PWM_CH2 up-count free CMPDAT point. 1101 = PWM_CH2 down-count free CMPDAT point. 1110 = PWM_CH4 up-count free CMPDAT point. 1111 = PWM_CH4 down-count free CMPDAT point.
[23]	**TRGEN2**	PWM_CH2 Trigger EADC enable bit
[22:20]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[19:16]	**TRGSEL2**	**PWM_CH2 Trigger EADC 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 = PWM_CH3 zero point. 0110 = PWM_CH3 period point. 0111 = PWM_CH3 zero or period point. 1000 = PWM_CH3 up-count CMPDAT point. 1001 = PWM_CH3 down-count CMPDAT point. 1010 = PWM_CH0 up-count free CMPDAT point. 1011 = PWM_CH0 down-count free CMPDAT point. 1100 = PWM_CH2 up-count free CMPDAT point. 1101 = PWM_CH2 down-count free CMPDAT point. 1110 = PWM_CH4 up-count free CMPDAT point. 1111 = PWM_CH4 down-count free CMPDAT point.
[15]	**TRGEN1**	PWM_CH1 Trigger EADC enable bit
[14:12]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[11:8]	**TRGSEL1**	**PWM_CH1 Trigger EADC 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 = PWM_CH1 zero point. 0110 = PWM_CH1 period point. 0111 = PWM_CH1 zero or period point. 1000 = PWM_CH1 up-count CMPDAT point. 1001 = PWM_CH1 down-count CMPDAT point. 1010 = PWM_CH0 up-count free CMPDAT point. 1011 = PWM_CH0 down-count free CMPDAT point. 1100 = PWM_CH2 up-count free CMPDAT point. 1101 = PWM_CH2 down-count free CMPDAT point. 1110 = PWM_CH4 up-count free CMPDAT point. 1111 = PWM_CH4 down-count free CMPDAT point.
[7]	**TRGEN0**	PWM_CH0 Trigger EADC enable bit
[6:4]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[3:0]	**TRGSEL0**	**PWM_CH0 Trigger EADC 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 = PWM_CH1 zero point. 0110 = PWM_CH1 period point. 0111 = PWM_CH1 zero or period point. 1000 = PWM_CH1 up-count CMPDAT point.

		1001 = PWM_CH1 down-count CMPDAT point. 1010 = PWM_CH0 up-count free CMPDAT point. 1011 = PWM_CH0 down-count free CMPDAT point. 1100 = PWM_CH2 up-count free CMPDAT point. 1101 = PWM_CH2 down-count free CMPDAT point. 1110 = PWM_CH4 up-count free CMPDAT point. 1111 = PWM_CH4 down-count free CMPDAT point.

PWM Trigger EADC Source Select Register 1 (PWM_EADCTS1)

Register	Offset	R/W	Description	Reset Value
PWM_EADCTS1	PWM0_BA+0xFC	R/W	PWM Trigger EADC 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. Any values read should be ignored. When writing to this field always write with reset value.
[15]	**TRGEN5**	PWM_CH5 Trigger EADC enable bit
[14:12]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[11:8]	**TRGSEL5**	**PWM_CH5 Trigger EADC 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 = PWM_CH5 zero point. 0110 = PWM_CH5 period point. 0111 = PWM_CH5 zero or period point. 1000 = PWM_CH5 up-count CMPDAT point. 1001 = PWM_CH5 down-count CMPDAT point. 1010 = PWM_CH0 up-count free CMPDAT point. 1011 = PWM_CH0 down-count free CMPDAT point. 1100 = PWM_CH2 up-count free CMPDAT point. 1101 = PWM_CH2 down-count free CMPDAT point. 1110 = PWM_CH4 up-count free CMPDAT point. 1111 = PWM_CH4 down-count free CMPDAT point.
[7]	**TRGEN4**	PWM_CH4 Trigger EADC enable bit
[6:4]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[3:0]	**TRGSEL4**	**PWM_CH4 Trigger EADC 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 = PWM_CH5 zero point. 0110 = PWM_CH5 period point. 0111 = PWM_CH5 zero or period point. 1000 = PWM_CH5 up-count CMPDAT point. 1001 = PWM_CH5 down-count CMPDAT point. 1010 = PWM_CH0 up-count free CMPDAT point. 1011 = PWM_CH0 down-count free CMPDAT point. 1100 = PWM_CH2 up-count free CMPDAT point. 1101 = PWM_CH2 down-count free CMPDAT point. 1110 = PWM_CH4 up-count free CMPDAT point. 1111 = PWM_CH4 down-count free CMPDAT point.

PWM Free Trigger Compare Register 0_1, 2_3, 4_5 (PWM_FTCMPDAT0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
PWM_FTCMPDAT0_1	PWM0_BA+0x100	R/W	PWM Free Trigger Compare Register 0/1	0x0000_0000
PWM_FTCMPDAT2_3	PWM0_BA+0x104	R/W	PWM Free Trigger Compare Register 2/3	0x0000_0000
PWM_FTCMPDAT4_5	PWM0_BA+0x108	R/W	PWM Free Trigger Compare 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**
FTCMP							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FTCMP							

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**FTCMP**	**PWM Free Trigger Compare Register** FTCMP use to compare with even CNTR to trigger EADC. FTCMPDAT0_1, 2_3, 4_5 corresponding complementary pairs PWM_CH0 and PWM_CH1, PWM_CH2 and PWM_CH3, PWM_CH4 and PWM_CH5.

PWM Synchronous Start Control Register (PWM_SSCTL)

Register	Offset	R/W	Description	Reset Value
PWM_SSCTL	PWM0_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		SSEN5	SSEN4	SSEN3	SSEN2	SSEN1	SSEN0

Bits	Description	
[31:10]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[9:8]	**SSRC**	**PWM Synchronous Start Source Select Bits** 00 = Synchronous start source come from PWM0. Other = No synchronous function.
[7:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5]	**SSEN5**	**PWM Channel 5 Synchronous Start Function Enable Bits** When synchronous start function is enabled, the PWM counter enable register (PWM_CNTEN) can be enabled by writing PWM synchronous start trigger bit (CNTSEN). 0 = PWM synchronous start function Disabled. 1 = PWM synchronous start function Enabled.
[4]	**SSEN4**	**PWM Channel 4 Synchronous Start Function Enable Bits** When synchronous start function is enabled, the PWM counter enable register (PWM_CNTEN) can be enabled by writing PWM synchronous start trigger bit (CNTSEN). 0 = PWM synchronous start function Disabled. 1 = PWM synchronous start function Enabled.
[3]	**SSEN3**	**PWM Channel 3 Synchronous Start Function Enable Bits** When synchronous start function is enabled, the PWM counter enable register (PWM_CNTEN) can be enabled by writing PWM synchronous start trigger bit (CNTSEN). 0 = PWM synchronous start function Disabled. 1 = PWM synchronous start function Enabled.
[2]	**SSEN2**	**PWM Channel 2 Synchronous Start Function Enable Bits** When synchronous start function is enabled, the PWM counter enable register (PWM_CNTEN) can be enabled by writing PWM synchronous start trigger bit (CNTSEN). 0 = PWM synchronous start function Disabled. 1 = PWM synchronous start function Enabled.

[1]	**SSEN1**	**PWM Channel 1 Synchronous Start Function Enable Bits** When synchronous start function is enabled, the PWM counter enable register (PWM_CNTEN) can be enabled by writing PWM synchronous start trigger bit (CNTSEN). 0 = PWM synchronous start function Disabled. 1 = PWM synchronous start function Enabled.
[0]	**SSEN0**	**PWM Channel 0 Synchronous Start Function Enable Bits** When synchronous start function is enabled, the PWM counter enable register (PWM_CNTEN) 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	PWM0_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. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**CNTSEN**	**PWM Counter Synchronous Start Enable (Write Only)** PMW counter synchronous enable function is used to make selected PWM channels (include PWM0_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	PWM0_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		ADCTRGF5	ADCTRGF4	ADCTRGF3	ADCTRGF2	ADCTRGF1	ADCTRGF0
15	14	13	12	11	10	9	8
Reserved					SYNCINF4	SYNCINF2	SYNCINF0
7	6	5	4	3	2	1	0
Reserved		CNTMAXF5	CNTMAXF4	CNTMAXF3	CNTMAXF2	CNTMAXF1	CNTMAXF0

Bits	Description	
[31:22]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[21]	**ADCTRGF5**	**PWM Channel 5 EADC Start of Conversion Flag** 0 = Indicates no EADC start of conversion trigger event has occurred. 1 = Indicates an EADC start of conversion trigger event has occurred, software can write 1 to clear this bit.
[20]	**ADCTRGF4**	**PWM Channel 4 EADC Start of Conversion Flag** 0 = Indicates no EADC start of conversion trigger event has occurred. 1 = Indicates an EADC start of conversion trigger event has occurred, software can write 1 to clear this bit.
[19]	**ADCTRGF3**	**PWM Channel 3 EADC Start of Conversion Flag** 0 = Indicates no EADC start of conversion trigger event has occurred. 1 = Indicates an EADC start of conversion trigger event has occurred, software can write 1 to clear this bit.
[18]	**ADCTRGF2**	**PWM Channel 2 EADC Start of Conversion Flag** 0 = Indicates no EADC start of conversion trigger event has occurred. 1 = Indicates an EADC start of conversion trigger event has occurred, software can write 1 to clear this bit.
[17]	**ADCTRGF1**	**PWM Channel 1 EADC Start of Conversion Flag** 0 = Indicates no EADC start of conversion trigger event has occurred. 1 = Indicates an EADC start of conversion trigger event has occurred, software can write 1 to clear this bit.
[16]	**ADCTRGF0**	**PWM Channel 0 EADC Start of Conversion Flag** 0 = Indicates no EADC start of conversion trigger event has occurred. 1 = Indicates an EADC start of conversion trigger event has occurred, software can write 1 to clear this bit.
[15:11]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[10]	**SYNCINF4**	**PWM Channel 4 Input Synchronization Latched Flag** 0 = Indicates no SYNC_IN event has occurred. 1 = Indicates an SYNC_IN event has occurred, software can write 1 to clear this bit.
[9]	**SYNCINF2**	**PWM Channel 2 Input Synchronization Latched Flag** 0 = Indicates no SYNC_IN event has occurred. 1 = Indicates an SYNC_IN event has occurred, software can write 1 to clear this bit.
[8]	**SYNCINF0**	**PWM Channel 0 Input Synchronization Latched Flag** 0 = Indicates no SYNC_IN event has occurred. 1 = Indicates an SYNC_IN event has occurred, software can write 1 to clear this bit.
[7:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5]	**CNTMAXF5**	**PWM Channel 5 Time-base Counter Equal to 0xFFFF Latched Flag** 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.
[4]	**CNTMAXF4**	**PWM Channel 4 Time-base Counter Equal to 0xFFFF Latched Flag** 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]	**CNTMAXF3**	**PWM Channel 3 Time-base Counter Equal to 0xFFFF Latched Flag** 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.
[2]	**CNTMAXF2**	**PWM Channel 2 Time-base Counter Equal to 0xFFFF Latched Flag** 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]	**CNTMAXF1**	**PWM Channel 1 Time-base Counter Equal to 0xFFFF Latched Flag** 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.
[0]	**CNTMAXF0**	**PWM Channel 0 Time-base Counter Equal to 0xFFFF Latched Flag** 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	PWM0_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. Any values read should be ignored. When writing to this field always write with reset value.
[5]	**CAPINEN5**	**PWM Channel 5 Capture Input Enable Bits** 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.
[4]	**CAPINEN4**	**PWM Channel 4 Capture Input Enable Bits** 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.
[3]	**CAPINEN3**	**PWM Channel 3 Capture Input Enable Bits** 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.
[2]	**CAPINEN2**	**PWM Channel 2 Capture Input Enable Bits** 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.
[1]	**CAPINEN1**	**PWM Channel 1 Capture Input Enable Bits** 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.
[0]	**CAPINEN0**	**PWM Channel 0 Capture Input Enable Bits** 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	PWM0_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. Any values read should be ignored. When writing to this field always write with reset value.
[29]	**FCRLDEN5**	**PWM Channel 5 Falling Capture Reload Enable Bits** 0 = Falling capture reload counter Disabled. 1 = Falling capture reload counter Enabled.
[28]	**FCRLDEN4**	**PWM Channel 4 Falling Capture Reload Enable Bits** 0 = Falling capture reload counter Disabled. 1 = Falling capture reload counter Enabled.
[27]	**FCRLDEN3**	**PWM Channel 3 Falling Capture Reload Enable Bits** 0 = Falling capture reload counter Disabled. 1 = Falling capture reload counter Enabled.
[26]	**FCRLDEN2**	**PWM Channel 2 Falling Capture Reload Enable Bits** 0 = Falling capture reload counter Disabled. 1 = Falling capture reload counter Enabled.
[25]	**FCRLDEN1**	**PWM Channel 1 Falling Capture Reload Enable Bits** 0 = Falling capture reload counter Disabled. 1 = Falling capture reload counter Enabled.
[24]	**FCRLDEN0**	**PWM Channel 0 Falling Capture Reload Enable Bits** 0 = Falling capture reload counter Disabled. 1 = Falling capture reload counter Enabled.
[23:22]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[21]	**RCRLDEN5**	**PWM Channel 5 Rising Capture Reload Enable Bits** 0 = Rising capture reload counter Disabled. 1 = Rising capture reload counter Enabled.

[20]	**RCRLDEN4**	**PWM Channel 4 Rising Capture Reload Enable Bits** 0 = Rising capture reload counter Disabled. 1 = Rising capture reload counter Enabled.
[19]	**RCRLDEN3**	**PWM Channel 3 Rising Capture Reload Enable Bits** 0 = Rising capture reload counter Disabled. 1 = Rising capture reload counter Enabled.
[18]	**RCRLDEN2**	**PWM Channel 2 Rising Capture Reload Enable Bits** 0 = Rising capture reload counter Disabled. 1 = Rising capture reload counter Enabled.
[17]	**RCRLDEN1**	**PWM Channel 1 Rising Capture Reload Enable Bits** 0 = Rising capture reload counter Disabled. 1 = Rising capture reload counter Enabled.
[16]	**RCRLDEN0**	**PWM Channel 0 Rising Capture Reload Enable Bits** 0 = Rising capture reload counter Disabled. 1 = Rising capture reload counter Enabled.
[15:14]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[13]	**CAPINV5**	**PWM Channel 5 Capture Inverter Enable Bits** 0 = Capture source inverter Disabled. 1 = Capture source inverter Enabled. Reverse the input signal from GPIO.
[12]	**CAPINV4**	**PWM Channel 4 Capture Inverter Enable Bits** 0 = Capture source inverter Disabled. 1 = Capture source inverter Enabled. Reverse the input signal from GPIO.
[11]	**CAPINV3**	**PWM Channel 3 Capture Inverter Enable Bits** 0 = Capture source inverter Disabled. 1 = Capture source inverter Enabled. Reverse the input signal from GPIO.
[10]	**CAPINV2**	**PWM Channel 2 Capture Inverter Enable Bits** 0 = Capture source inverter Disabled. 1 = Capture source inverter Enabled. Reverse the input signal from GPIO.
[9]	**CAPINV1**	**PWM Channel 1 Capture Inverter Enable Bits** 0 = Capture source inverter Disabled. 1 = Capture source inverter Enabled. Reverse the input signal from GPIO.
[8]	**CAPINV0**	**PWM Channel 0 Capture Inverter Enable Bits** 0 = Capture source inverter Disabled. 1 = Capture source inverter Enabled. Reverse the input signal from GPIO.
[7:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5]	**CAPEN5**	**PWM Channel 5 Capture Function Enable Bits** 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).
[4]	**CAPEN4**	**PWM Channel 4 Capture Function Enable Bits** 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).
[3]	**CAPEN3**	**PWM Channel 3 Capture Function Enable Bits** 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).
[2]	**CAPEN2**	**PWM Channel 2 Capture Function Enable Bits** 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).
[1]	**CAPEN1**	**PWM Channel 1 Capture Function Enable Bits** 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).
[0]	**CAPEN0**	**PWM Channel 0 Capture Function Enable Bits** 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	PWM0_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. Any values read should be ignored. When writing to this field always write with reset value.
[13]	**CFLIFOV5**	**PWM Channel 5 Capture Falling Latch Interrupt Flag Overrun Status (Read Only)** This flag indicates if falling latch happened when the corresponding CFLIF is 1. **Note:** This bit will be cleared automatically when user clear corresponding CFLIF.
[12]	**CFLIFOV4**	**PWM Channel 4 Capture Falling Latch Interrupt Flag Overrun Status (Read Only)** This flag indicates if falling latch happened when the corresponding CFLIF is 1. **Note:** This bit will be cleared automatically when user clear corresponding CFLIF.
[11]	**CFLIFOV3**	**PWM Channel 3 Capture Falling Latch Interrupt Flag Overrun Status (Read Only)** This flag indicates if falling latch happened when the corresponding CFLIF is 1. **Note:** This bit will be cleared automatically when user clear corresponding CFLIF.
[10]	**CFLIFOV2**	**PWM Channel 2 Capture Falling Latch Interrupt Flag Overrun Status (Read Only)** This flag indicates if falling latch happened when the corresponding CFLIF is 1. **Note:** This bit will be cleared automatically when user clear corresponding CFLIF.
[9]	**CFLIFOV1**	**PWM Channel 1 Capture Falling Latch Interrupt Flag Overrun Status (Read Only)** This flag indicates if falling latch happened when the corresponding CFLIF is 1. **Note:** This bit will be cleared automatically when user clear corresponding CFLIF.
[8]	**CFLIFOV0**	**PWM Channel 0 Capture Falling Latch Interrupt Flag Overrun Status (Read Only)** This flag indicates if falling latch happened when the corresponding CFLIF is 1. **Note:** This bit will be cleared automatically when user clear corresponding CFLIF.
[7:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5]	**CRLIFOV5**	**PWM Channel 5 Capture Rising Latch Interrupt Flag Overrun Status (Read Only)** This flag indicates if rising latch happened when the corresponding CRLIF is 1. **Note:** This bit will be cleared automatically when user clear corresponding CRLIF.

[4]	**CRLIFOV4**	**PWM Channel 4 Capture Rising Latch Interrupt Flag Overrun Status (Read Only)** This flag indicates if rising latch happened when the corresponding CRLIF is 1. **Note:** This bit will be cleared automatically when user clear corresponding CRLIF.
[3]	**CRLIFOV3**	**PWM Channel 3 Capture Rising Latch Interrupt Flag Overrun Status (Read Only)** This flag indicates if rising latch happened when the corresponding CRLIF is 1. **Note:** This bit will be cleared automatically when user clear corresponding CRLIF.
[2]	**CRLIFOV2**	**PWM Channel 2 Capture Rising Latch Interrupt Flag Overrun Status (Read Only)** This flag indicates if rising latch happened when the corresponding CRLIF is 1. **Note:** This bit will be cleared automatically when user clear corresponding CRLIF.
[1]	**CRLIFOV1**	**PWM Channel 1 Capture Rising Latch Interrupt Flag Overrun Status (Read Only)** This flag indicates if rising latch happened when the corresponding CRLIF is 1. **Note:** This bit will be cleared automatically when user clear corresponding CRLIF.
[0]	**CRLIFOV0**	**PWM Channel 0 Capture Rising Latch Interrupt Flag Overrun Status (Read Only)** This flag indicates if rising latch happened when the corresponding CRLIF is 1. **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	PWM0_BA+0x20C	R	PWM Rising Capture Data Register 0	0x0000_0000
PWM_RCAPDAT1	PWM0_BA+0x214	R	PWM Rising Capture Data Register 1	0x0000_0000
PWM_RCAPDAT2	PWM0_BA+0x21C	R	PWM Rising Capture Data Register 2	0x0000_0000
PWM_RCAPDAT3	PWM0_BA+0x224	R	PWM Rising Capture Data Register 3	0x0000_0000
PWM_RCAPDAT4	PWM0_BA+0x22C	R	PWM Rising Capture Data Register 4	0x0000_0000
PWM_RCAPDAT5	PWM0_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. Any values read should be ignored. When writing to this field always write with reset value.
[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	PWM0_BA+0x210	R	PWM Falling Capture Data Register 0	0x0000_0000
PWM_FCAPDAT1	PWM0_BA+0x218	R	PWM Falling Capture Data Register 1	0x0000_0000
PWM_FCAPDAT2	PWM0_BA+0x220	R	PWM Falling Capture Data Register 2	0x0000_0000
PWM_FCAPDAT3	PWM0_BA+0x228	R	PWM Falling Capture Data Register 3	0x0000_0000
PWM_FCAPDAT4	PWM0_BA+0x230	R	PWM Falling Capture Data Register 4	0x0000_0000
PWM_FCAPDAT5	PWM0_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. Any values read should be ignored. When writing to this field always write with reset value.
[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	PWM0_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. Any values read should be ignored. When writing to this field always write with reset value.
[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. Do not use. 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** 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. Any values read should be ignored. When writing to this field always write with reset value.
[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. Do not use. 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** 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. Any values read should be ignored. When writing to this field always write with reset value.
[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. Do not use. 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** 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	PWM0_BA+0x240	R	PWM Capture Channel 01 PDMA Register	0x0000_0000
PWM_PDMACAP2_3	PWM0_BA+0x244	R	PWM Capture Channel 23 PDMA Register	0x0000_0000
PWM_PDMACAP4_5	PWM0_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. Any values read should be ignored. When writing to this field always write with reset value.
[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	PWM0_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. Any values read should be ignored. When writing to this field always write with reset value.
[13]	**CAPFIEN5**	**PWM Channel 5 Capture Falling Latch Interrupt Enable Bits** 0 = Capture falling edge latch interrupt Disabled. 1 = Capture falling edge latch interrupt Enabled. **Note:** When Capture with PDMA operating, CINTENR corresponding channel CAPFIEN must be disabled.
[12]	**CAPFIEN4**	**PWM Channel 4 Capture Falling Latch Interrupt Enable Bits** 0 = Capture falling edge latch interrupt Disabled. 1 = Capture falling edge latch interrupt Enabled. **Note:** When Capture with PDMA operating, CINTENR corresponding channel CAPFIEN must be disabled.
[11]	**CAPFIEN3**	**PWM Channel 3 Capture Falling Latch Interrupt Enable Bits** 0 = Capture falling edge latch interrupt Disabled. 1 = Capture falling edge latch interrupt Enabled. **Note:** When Capture with PDMA operating, CINTENR corresponding channel CAPFIEN must be disabled.
[10]	**CAPFIEN2**	**PWM Channel 2 Capture Falling Latch Interrupt Enable Bits** 0 = Capture falling edge latch interrupt Disabled. 1 = Capture falling edge latch interrupt Enabled. **Note:** When Capture with PDMA operating, CINTENR corresponding channel CAPFIEN must be disabled.
[9]	**CAPFIEN1**	**PWM Channel 1 Capture Falling Latch Interrupt Enable Bits** 0 = Capture falling edge latch interrupt Disabled. 1 = Capture falling edge latch interrupt Enabled. **Note:** When Capture with PDMA operating, CINTENR corresponding channel CAPFIEN must be disabled.

[8]	**CAPFIEN0**	**PWM Channel 0 Capture Falling Latch Interrupt Enable Bits** 0 = Capture falling edge latch interrupt Disabled. 1 = Capture falling edge latch interrupt Enabled. **Note:** When Capture with PDMA operating, CINTENR corresponding channel CAPFIEN must be disabled.
[7:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5]	**CAPRIEN5**	**PWM Channel 5 Capture Rising Latch Interrupt Enable Bits** 0 = Capture rising edge latch interrupt Disabled. 1 = Capture rising edge latch interrupt Enabled. **Note:** When Capture with PDMA operating, CINTENR corresponding channel CAPRIEN must be disabled.
[4]	**CAPRIEN4**	**PWM Channel 4 Capture Rising Latch Interrupt Enable Bits** 0 = Capture rising edge latch interrupt Disabled. 1 = Capture rising edge latch interrupt Enabled. **Note:** When Capture with PDMA operating, CINTENR corresponding channel CAPRIEN must be disabled.
[3]	**CAPRIEN3**	**PWM Channel 3 Capture Rising Latch Interrupt Enable Bits** 0 = Capture rising edge latch interrupt Disabled. 1 = Capture rising edge latch interrupt Enabled. **Note:** When Capture with PDMA operating, CINTENR corresponding channel CAPRIEN must be disabled.
[2]	**CAPRIEN2**	**PWM Channel 2 Capture Rising Latch Interrupt Enable Bits** 0 = Capture rising edge latch interrupt Disabled. 1 = Capture rising edge latch interrupt Enabled. **Note:** When Capture with PDMA operating, CINTENR corresponding channel CAPRIEN must be disabled.
[1]	**CAPRIEN1**	**PWM Channel 1 Capture Rising Latch Interrupt Enable Bits** 0 = Capture rising edge latch interrupt Disabled. 1 = Capture rising edge latch interrupt Enabled. **Note:** When Capture with PDMA operating, CINTENR corresponding channel CAPRIEN must be disabled.
[0]	**CAPRIEN0**	**PWM Channel 0 Capture Rising Latch Interrupt Enable Bits** 0 = Capture rising edge latch interrupt Disabled. 1 = Capture rising edge latch interrupt Enabled. **Note:** When Capture with PDMA operating, CINTENR corresponding channel CAPRIEN must be disabled.

PWM Capture Interrupt Flag Register (PWM_CAPIF)

Register	Offset	R/W	Description	Reset Value
PWM_CAPIF	PWM0_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. Any values read should be ignored. When writing to this field always write with reset value.
[13]	**CFLIF5**	**PWM Channel 5 Capture Falling Latch Interrupt Flag** This bit is writing 1 to clear. 0 = No capture falling latch condition happened. 1 = Capture falling latch condition happened, this flag will be set to high. **Note:** When Capture with PDMA operating, CAPIF corresponding channel CFLIF will cleared by hardware after PDMA transfer data.
[12]	**CFLIF4**	**PWM Channel 4 Capture Falling Latch Interrupt Flag** This bit is writing 1 to clear. 0 = No capture falling latch condition happened. 1 = Capture falling latch condition happened, this flag will be set to high. **Note:** When Capture with PDMA operating, CAPIF corresponding channel CFLIF will cleared by hardware after PDMA transfer data.
[11]	**CFLIF3**	**PWM Channel 3 Capture Falling Latch Interrupt Flag** This bit is writing 1 to clear. 0 = No capture falling latch condition happened. 1 = Capture falling latch condition happened, this flag will be set to high. **Note:** When Capture with PDMA operating, CAPIF corresponding channel CFLIF will cleared by hardware after PDMA transfer data.
[10]	**CFLIF2**	**PWM Channel 2 Capture Falling Latch Interrupt Flag** This bit is writing 1 to clear. 0 = No capture falling latch condition happened. 1 = Capture falling latch condition happened, this flag will be set to high. **Note:** When Capture with PDMA operating, CAPIF corresponding channel CFLIF will cleared by hardware after PDMA transfer data.
[9]	**CFLIF1**	**PWM Channel 1 Capture Falling Latch Interrupt Flag**

		This bit is writing 1 to clear. 0 = No capture falling latch condition happened. 1 = Capture falling latch condition happened, this flag will be set to high. **Note:** When Capture with PDMA operating, CAPIF corresponding channel CFLIF will cleared by hardware after PDMA transfer data.
[8]	**CFLIF0**	**PWM Channel 0 Capture Falling Latch Interrupt Flag** This bit is writing 1 to clear. 0 = No capture falling latch condition happened. 1 = Capture falling latch condition happened, this flag will be set to high. **Note:** When Capture with PDMA operating, CAPIF corresponding channel CFLIF will cleared by hardware after PDMA transfer data.
[7:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5]	**CRLIF5**	**PWM Channel 5 Capture Rising Latch Interrupt Flag** This bit is writing 1 to clear. 0 = No capture rising latch condition happened. 1 = Capture rising latch condition happened, this flag will be set to high. **Note:** When Capture with PDMA operating, CAPIF corresponding channel CRLIF will cleared by hardware after PDMA transfer data.
[4]	**CRLIF4**	**PWM Channel 4 Capture Rising Latch Interrupt Flag** This bit is writing 1 to clear. 0 = No capture rising latch condition happened. 1 = Capture rising latch condition happened, this flag will be set to high. **Note:** When Capture with PDMA operating, CAPIF corresponding channel CRLIF will cleared by hardware after PDMA transfer data.
[3]	**CRLIF3**	**PWM Channel 3 Capture Rising Latch Interrupt Flag** This bit is writing 1 to clear. 0 = No capture rising latch condition happened. 1 = Capture rising latch condition happened, this flag will be set to high. **Note:** When Capture with PDMA operating, CAPIF corresponding channel CRLIF will cleared by hardware after PDMA transfer data.
[2]	**CRLIF2**	**PWM Channel 2 Capture Rising Latch Interrupt Flag** This bit is writing 1 to clear. 0 = No capture rising latch condition happened. 1 = Capture rising latch condition happened, this flag will be set to high. **Note:** When Capture with PDMA operating, CAPIF corresponding channel CRLIF will cleared by hardware after PDMA transfer data.
[1]	**CRLIF1**	**PWM Channel 1 Capture Rising Latch Interrupt Flag** This bit is writing 1 to clear. 0 = No capture rising latch condition happened. 1 = Capture rising latch condition happened, this flag will be set to high. **Note:** When Capture with PDMA operating, CAPIF corresponding channel CRLIF will cleared by hardware after PDMA transfer data.
[0]	**CRLIF0**	**PWM Channel 0 Capture Rising Latch Interrupt Flag** This bit is writing 1 to clear. 0 = No capture rising latch condition happened. 1 = Capture rising latch condition happened, this flag will be set to high. **Note:** When Capture with PDMA operating, CAPIF corresponding channel CRLIF will cleared by hardware after PDMA transfer data.

PWM Period Register Buffer 0~5 (PWM_PBUF0~5)

Register	Offset	R/W	Description	Reset Value
PWM_PBUF0	PWM0_BA+0x304	R	PWM PERIOD0 Buffer	0x0000_0000
PWM_PBUF1	PWM0_BA+0x308	R	PWM PERIOD1 Buffer	0x0000_0000
PWM_PBUF2	PWM0_BA+0x30C	R	PWM PERIOD2 Buffer	0x0000_0000
PWM_PBUF3	PWM0_BA+0x310	R	PWM PERIOD3 Buffer	0x0000_0000
PWM_PBUF4	PWM0_BA+0x314	R	PWM PERIOD4 Buffer	0x0000_0000
PWM_PBUF5	PWM0_BA+0x318	R	PWM PERIOD5 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. Any values read should be ignored. When writing to this field always write with reset value.
[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	PWM0_BA+0x31C	R	PWM CMPDAT0 Buffer	0x0000_0000
PWM_CMPBUF1	PWM0_BA+0x320	R	PWM CMPDAT1 Buffer	0x0000_0000
PWM_CMPBUF2	PWM0_BA+0x324	R	PWM CMPDAT2 Buffer	0x0000_0000
PWM_CMPBUF3	PWM0_BA+0x328	R	PWM CMPDAT3 Buffer	0x0000_0000
PWM_CMPBUF4	PWM0_BA+0x32C	R	PWM CMPDAT4 Buffer	0x0000_0000
PWM_CMPBUF5	PWM0_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. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**CMPBUF**	**PWM Comparator Register Buffer (Read Only)** Used as CMP active register.

PWM CLKPSC Buffer 0_1, 2_3, 4_5 (PWM_CPSCBUF0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
PWM_CPSCBUF0_1	PWM0_BA+0x334	R	PWM CLKPSC0_1 Buffer	0x0000_0000
PWM_CPSCBUF2_3	PWM0_BA+0x338	R	PWM CLKPSC2_3 Buffer	0x0000_0000
PWM_CPSCBUF4_5	PWM0_BA+0x33C	R	PWM CLKPSC4_5 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**
Reserved				**CPSCBUF**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CPSCBUF							

Bits	Description	
[31:12]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[11:0]	**CPSCBUF**	**PWM Counter Clock Pre-scale Buffer** Use as PWM counter clock pre-scare active register.

PWM_FTCMPDAT Buffer (PWM_FTCBUF0_1,2_3,4_5)

Register	Offset	R/W	Description	Reset Value
PWM_FTCBUF0_1	PWM0_BA+0x340	R	PWM FTCMPDAT0_1 Buffer	0x0000_0000
PWM_FTCBUF2_3	PWM0_BA+0x344	R	PWM FTCMPDAT2_3 Buffer	0x0000_0000
PWM_FTCBUF4_5	PWM0_BA+0x348	R	PWM FTCMPDAT4_5 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**
FTCMPBUF							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FTCMPBUF							

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**FTCMPBUF**	**PWM FTCMPDAT Buffer (Read Only)** Used as FTCMPDAT active register.

PWM FTCMPDAT Indicator Register (PWM_FTCI)

Register	Offset	R/W	Description	Reset Value
PWM_FTCI	PWM0_BA+0x34C	R/W	PWM FTCMPDAT Indicator Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**FTCMD4**	**FTCMD2**	**FTCMD0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**FTCMU4**	**FTCMU2**	**FTCMU0**

Bits	Description	
[31:11]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[10]	**FTCMD4**	**PWM Channel 4 FTCMPDAT Down Indicator** Indicator will be set to high when FTCMPDAT4_5 equal to PERIOD0 and DIRF=0, software can write 1 to clear this bit.
[9]	**FTCMD2**	**PWM Channel 2 FTCMPDAT Down Indicator** Indicator will be set to high when FTCMPDAT2_3 equal to PERIOD0 and DIRF=0, software can write 1 to clear this bit.
[8]	**FTCMD0**	**PWM Channel 0 FTCMPDAT Down Indicator** Indicator will be set to high when FTCMPDAT0_1 equal to PERIOD0 and DIRF=0, software can write 1 to clear this bit.
[7:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**FTCMU4**	**PWM Channel 4 FTCMPDAT Up Indicator** Indicator will be set to high when FTCMPDAT0_1 equal to PERIODn and DIRF=1, software can write 1 to clear this bit.
[1]	**FTCMU2**	**PWM Channel 2 FTCMPDAT Up Indicator** Indicator will be set to high when FTCMPDAT0_1 equal to PERIODn and DIRF=1, software can write 1 to clear this bit.
[0]	**FTCMU0**	**PWM Channel 0 FTCMPDAT Up Indicator** Indicator will be set to high when FTCMPDAT0_1 equal to PERIODn and DIRF=1, software can write 1 to clear this bit.

6.9 Watchdog Timer (WDT)

6.9.1 Overview

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

6.9.2 Features

- 18-bit free running up counter for WDT time-out interval
- Selectable time-out interval (2^4 ~ 2^{18}) and the time-out interval is 1.6 ms ~ 26.214 s if WDT_CLK = 10 kHz.
- System kept in reset state for a period of (1 / WDT_CLK) * 63
- Supports selectable WDT reset delay period, including 1026、130、18 or 3 WDT_CLK reset delay period
- Supports to force WDT enabled after chip power-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 10 kHz or LXT.

6.9.3 Block Diagram

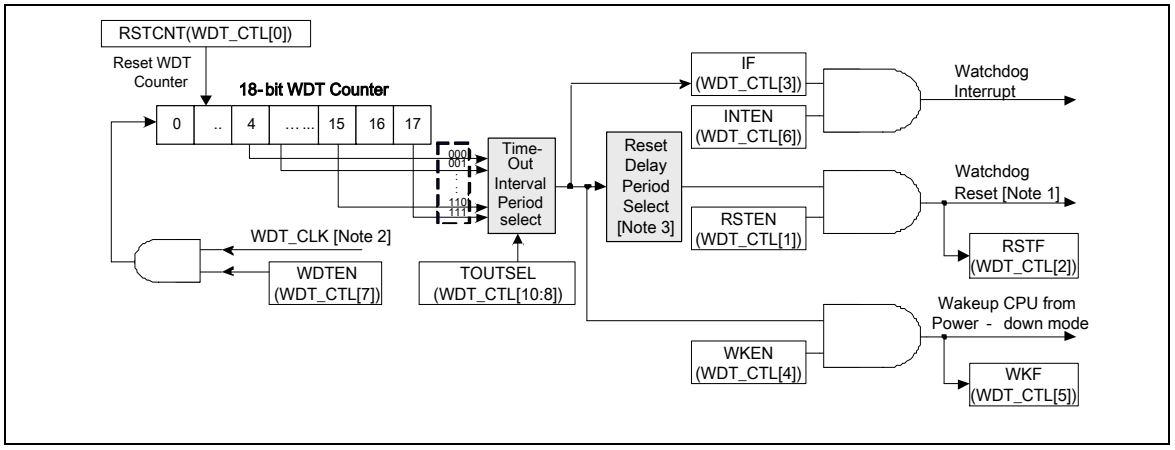


Figure 6.9-1 Watchdog Timer Block Diagram

Note1: WDT resets CPU and lasts 63 WDT_CLK.

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

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

6.9.4 Basic Configuration

- Clock source configuration
  - Select the source of WDT peripheral clock on WDTSEL (CLK_CLKSEL1[1:0])

- Enable WDT peripheral clock in WDTCKEN (CLK_APBCLK0[0]).
- Force enable WDT controller after chip power-on or reset in CWDTEN[2:0] (CWDTEN[2] is CONFIG0[31], CWDTEN[1:0] is CONFIG0[4:3])

The WDT clock control are shown in Figure 6.9-2

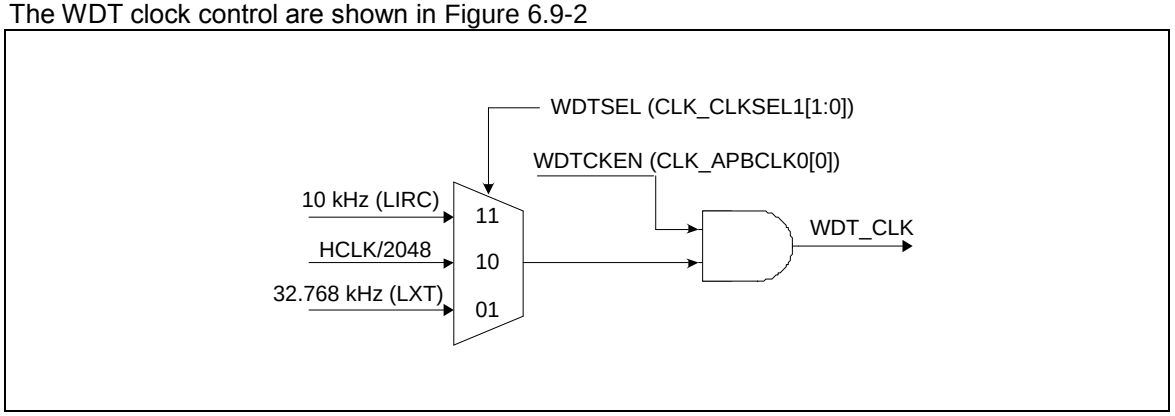


Figure 6.9-2 Watchdog Timer Clock Control

6.9.5 Functional Description

The WDT includes an 18-bit free running up counter with programmable time-out intervals. Table 6.9.5-1 shows the WDT time-out interval period selection and Figure 6.9-3 shows the WDT time-out interval and reset period timing.

6.9.5.1 WDT Time-out Interrupt

Setting WDTEN (WDTCR[7]) to 1 will enable the WDT function and the WDT counter to start counting up. There are eight time-out interval period can be selected by setting 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.

6.9.5.2 WDT Reset Delay Period and Reset System

There is a specified T_{RSTD} reset delay period follows the IF (WDT_CTL[3]) is setting 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 , 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.

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 6.9.5-1 Watchdog Timer Time-out Interval Period Selection

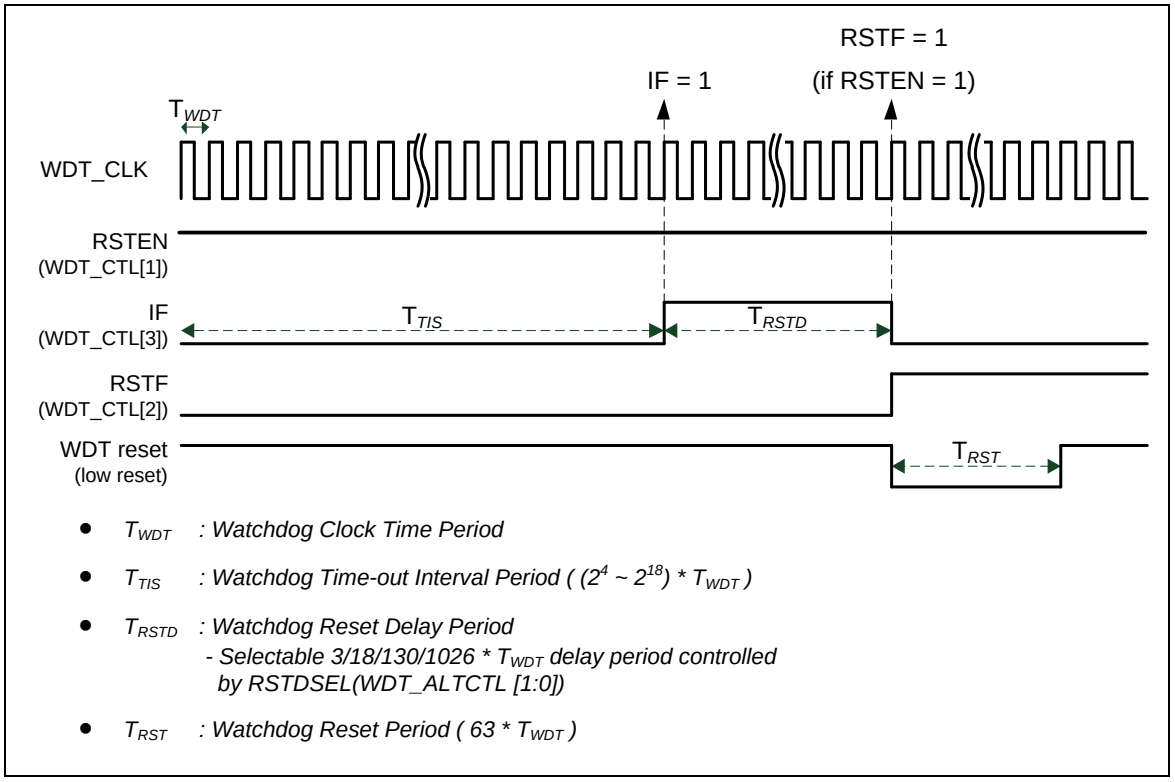


Figure 6.9-3 Watchdog Timer Time-out Interval and Reset Period Timing

6.9.5.3 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. Notice that user should set LXTEN (CLK_PWRCTL [1]) or LIRCEN (CLK_PWRCTL [3]) to select clock source before system entries Power-down mode because the system peripheral clock are disabled when system is Power-down mode. 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.

6.9.5.4 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.

6.9.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
WDT Base Address: **WDT_BA = 0x4004_0000**				
WDT_CTL	WDT_BA+0x00	R/W	WDT Control Register	0x0000_07X0
WDT_ALTCTL	WDT_BA+0x04	R/W	WDT Alternative Control Register	0x0000_0000

Note:

1. Any register not listed here is reserved and must not be written. The result of a read operation on these bits is undefined.
2. The reserved register fields that listed in register description must be written to their reset value. Writing reserved fields with other than reset values may produce undefined results.

6.9.7 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_07X0

31	30	29	28	27	26	25	24
ICEDEBUG	**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 Protected)** 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.
[30:11]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[10:8]	**TOUTSEL**	**WDT Time-out Interval Selection (Write Protected)** 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 Protected)** 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 111, this bit is forced as 1 and user cannot change this bit to 0.
[6]	**INTEN**	**WDT Time-out Interrupt Enable Control (Write Protected)**

		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. **Note1:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note2:** The reset value of this bit is 0.
[5]	**WKF**	**WDT Time-out Wake-up Flag (Write Protected)** 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. **Note3:** The reset value of this bit is 0.
[4]	**WKEN**	**WDT Time-out Wake-up Function Control (Write Protected)** 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 internal low speed RC oscillator (LIRC) or LXT. **Note3:** The reset value of this bit is 0.
[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 Protected)** 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 Protected)** 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. Any values read should be ignored. When writing to this field always write with reset value.
[1:0]	**RSTDSEL**	**WDT Reset Delay Selection (Write Protected)** 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.

6.10 Window Watchdog Timer (WWDT)

6.10.1 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.

6.10.2 Features

- 6-bit down counter value (CNTDAT, WWDT_CNT[5:0]) and 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

6.10.3 Block Diagram

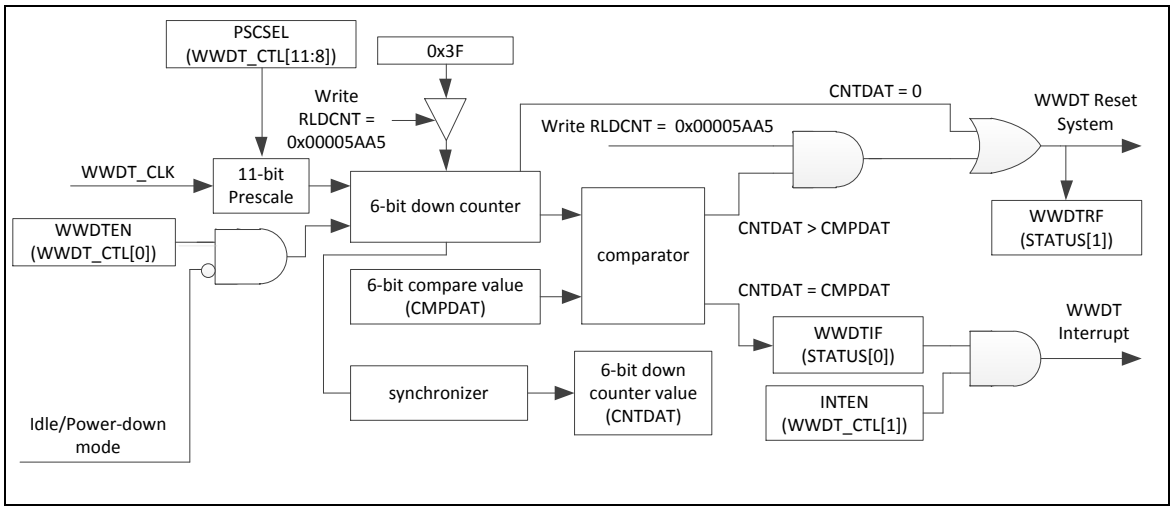


Figure 6.10-1 WWDT Block Diagram

6.10.4 Basic Configuration

- Clock source configuration
  - Select the source of WWDT peripheral clock in WWDTSEL (CLK_CLKSEL1[31:30])
  - Enable WWDT peripheral clock in WDTCKEN (CLK_APBCLK0[0]).

The WWDT clock control are shown in Figure 6.10-2.

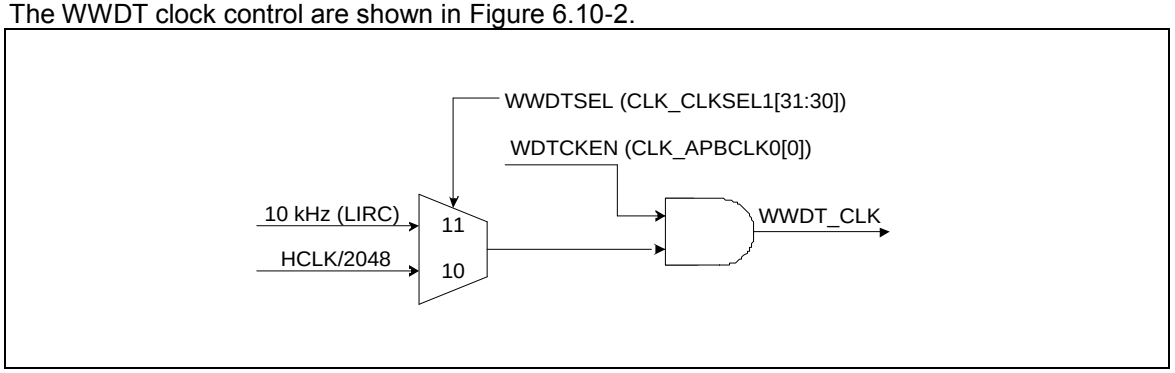


Figure 6.10-2 WWDT Clock Control

6.10.5 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 divide 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 the Table 6.10.5-1.

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 6.10.5-1 WWDT Prescaler Value Selection

6.10.5.1 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 the system is reset 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 start down counting.

6.10.5.2 WWDT Compare Match Interrupt

During down counting by the WWDT counter, the WWDTIF (WWDT_STATUS[0]) is set to 1 while the WWDT counter value (CNTDAT) is equal to window compare value (CMPDAT) and WWDTIF can be cleared by user; if INTEN (WWDT_CTL[1]) is also set to 1 by user, the WWDT compare match interrupt signal is generated also while WWDTIF is set to 1 by hardware.

6.10.5.3 WWDT Reset System

Figure 6.10-3 shows three cases of WWDT reset and reload behavior.

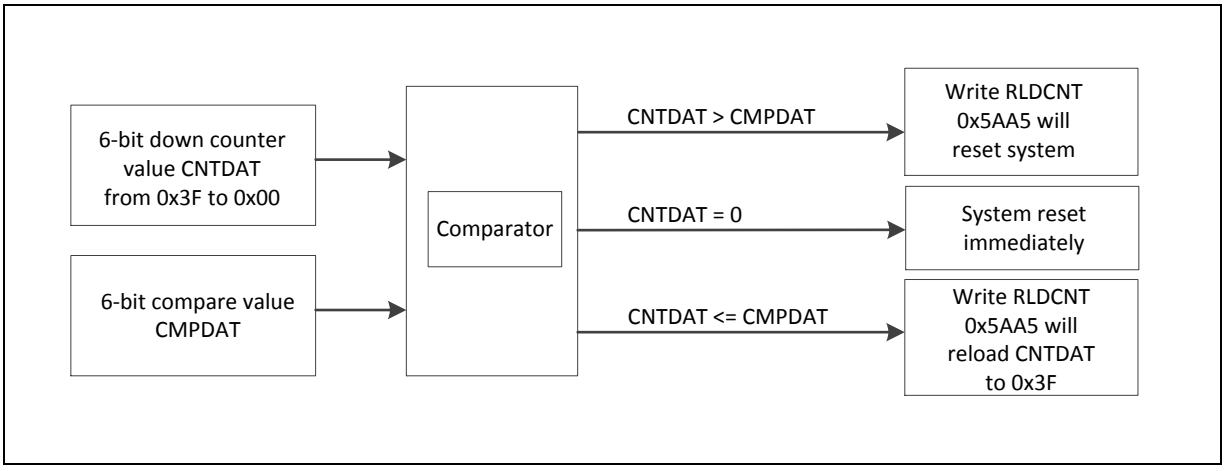


Figure 6.10-3 WWDT Reset and Reload Behavior

If current CNTDAT (WWDT_CNT[5:0]) 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. The waveform of WWDT reload counter when

CNTDAT > CMPDAT is shown in Figure 6.10-4.

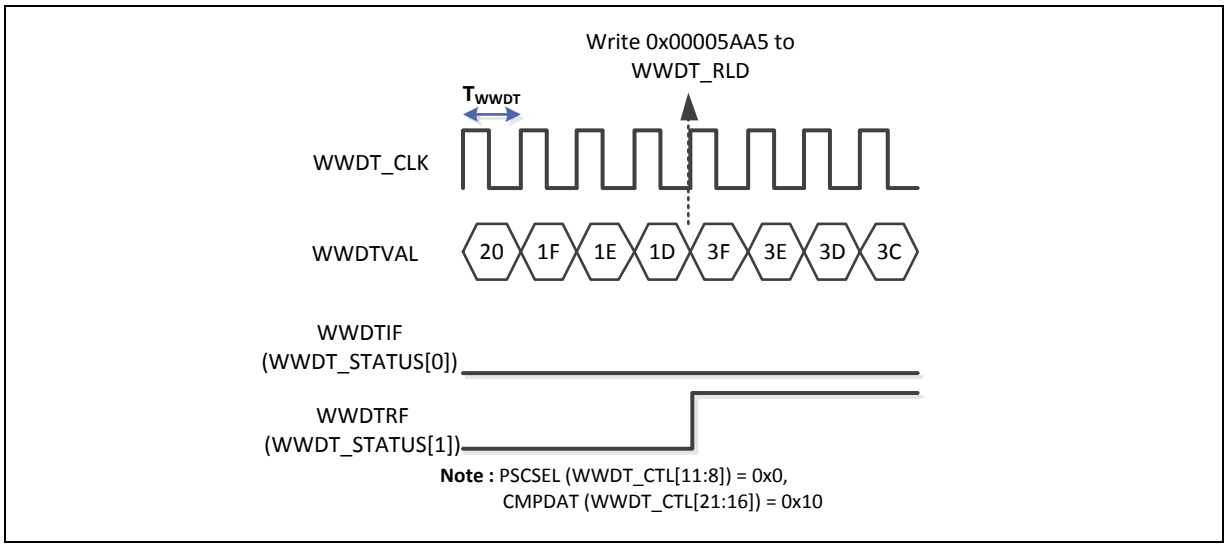


Figure 6.10-4 WWDT Reload Counter When CNTDAT > CMPDAT

When WWDTIF (WWDT_STATUS[0]) is generated, user must reload WWDT counter value to 0x3F by writing 0x00005AA5 to WWDT_RLDCNT register, and also to prevent WWDT counter value reached to 0 and generate WWDT reset system signal to info system reset. Figure 6.10-5 shows the waveform of WWDT reload counter when CNTDAT < CMPDAT and Figure 6.10-6 shows WWDT generate reset system signal (WWDTRF) if user doesn't write 0x00005AA5 to WWDT_RLD before WWDT counter value reach to 0.

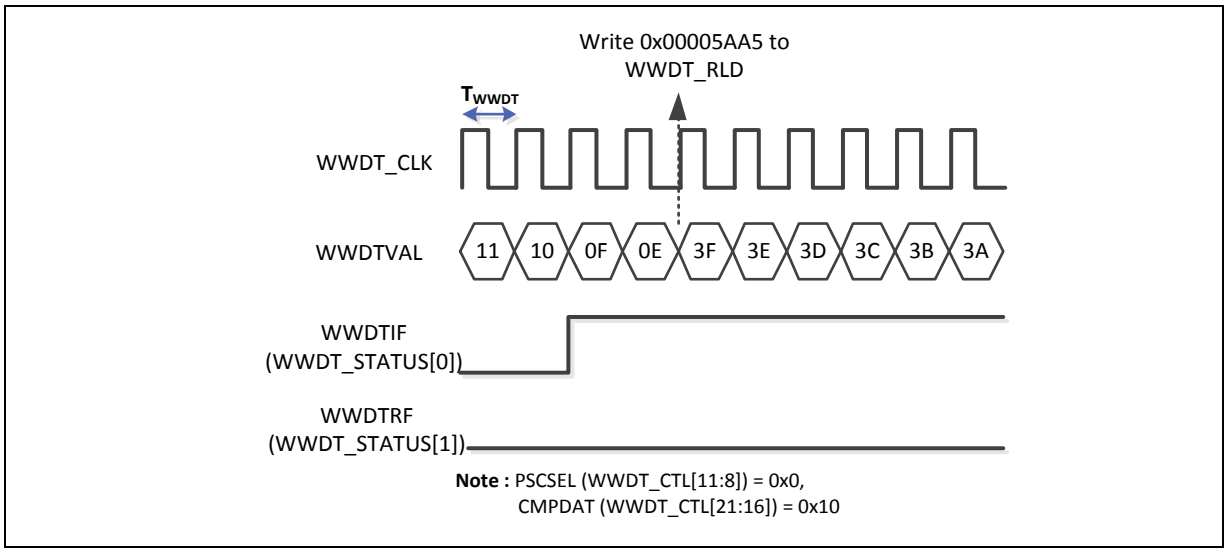


Figure 6.10-5 WWDT Reload Counter When WWDT_CNT < WINCMP

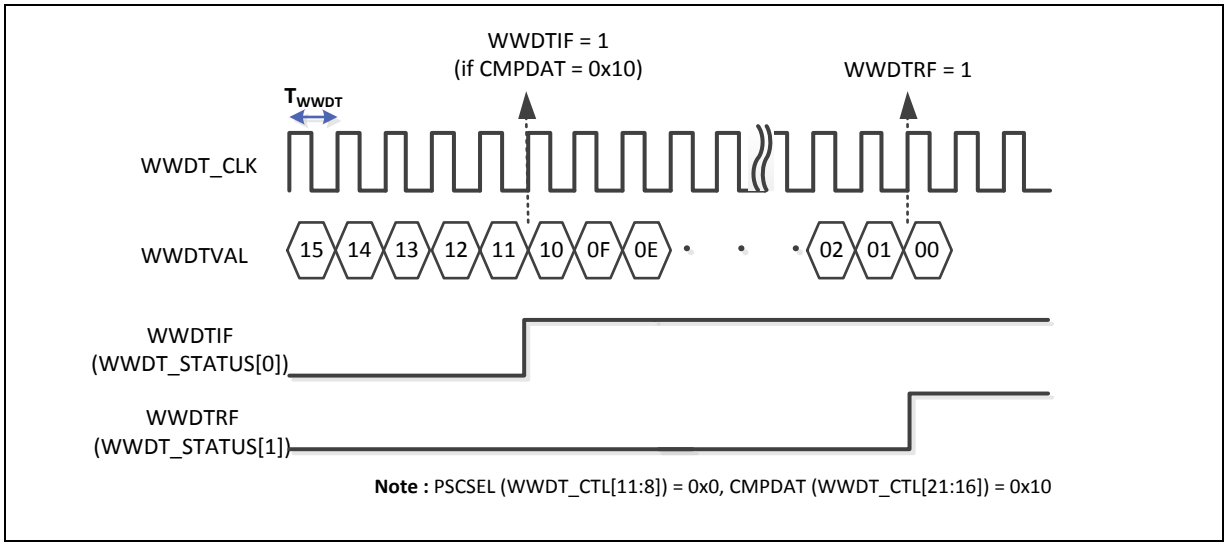


Figure 6.10-6 WWDT Interrupt and Reset Signals

6.10.5.4 WWDT Window Setting Limitation

When user writes 0x00005AA5 to WWDT_RLDCNT register to reload WWDT counter value to 0x3F, it needs 3 WWDT clocks to sync the reload command to actually perform reload action. Notice that if user set 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. The WWDT CMPDAT setting limitation is shown in Table 6.10.5-2.

PSCSEL	Prescale Value	Valid CMPDAT Value
0000	1	0x3 ~ 0x3F
0001	2	0x2 ~ 0x3F
Others	Others	0x0 ~ 0x3F

Table 6.10.5-2 CMPDAT Setting Limitation

6.10.5.5 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.

6.10.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
WWDT Base Address: **WWDT_BA = 0x4004_0100**				
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

Note:

1. Any register not listed here is reserved and must not be written. The result of a read operation on these bits is undefined.
2. The reserved register fields that listed in register description must be written to their reset value. Writing reserved fields with other than reset values may produce undefined results.

6.10.7 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. **Note:** User can only write WWDT_RLDCNT register to reload WWDT counter value when current WWDT counter value between 0 and CMPDAT (WWDT_CTL[21:16]). If user writes WWDT_RLDCNT when current WWDT counter value is larger than CMPDAT , WWDT reset signal will generate immediately.

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. Any values read should be ignored. When writing to this field always write with reset value.
[21:16]	**CMPDAT**	**WWDT Window Compare Register** Set this register to adjust the valid reload window. **Note:** User can only write WWDT_RLDCNT register to reload WWDT counter value when current WWDT counter value between 0 and CMPDAT. If user writes WWDT_RLDCNT register when current WWDT counter value larger than CMPDAT, WWDT reset signal will generate immediately.
[15:12]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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. 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. Any values read should be ignored. When writing to this field always write with reset value.
[1]	**INTEN**	**WWDT Interrupt Enable Control Bit** If this bit is enabled, the WWDT counter compare match interrupt signal is generated and inform to CPU. 0 = WWDT counter compare match interrupt Disabled. 1 = WWDT counter compare match interrupt Enabled.
[0]	**WWDTEN**	**WWDT Enable Control 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. Any values read should be ignored. When writing to this field always write with reset value.
[1]	**WWDTRF**	**WWDT Timer-out Reset Flag** This bit indicates the system has been reset by WWDT time-out reset or not. 0 = WWDT time-out reset did not occur. 1 = WWDT time-out reset occurred. **Note:** This bit is cleared by writing 1 to it.
[0]	**WWDTIF**	**WWDT Compare Match Interrupt Flag** This bit indicates the interrupt flag status of WWDT while WWDT counter value matches 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. Any values read should be ignored. When writing to this field always write with reset value.
[5:0]	**CNTDAT**	**WWDT Counter Value** CNTDAT will be updated continuously to monitor 6-bit WWDT down counter value.

6.11 Real Time Clock (RTC)

6.11.1 Overview

The Real Time Clock (RTC) controller provides the real time and calendar message. The RTC offers programmable time tick and alarm match interrupts. The data format of time and calendar message are expressed in BCD format. A digital frequency compensation feature is available to compensate external crystal oscillator frequency accuracy.

6.11.2 Features

- Supports real time counter in RTC_TIME (hour, minute, second) and calendar counter in RTC_CAL (year, month, day) for RTC time and calendar check
- Supports alarm time (hour, minute, second) and calendar (year, month, day) setting in RTC_TALM and RTC_CALM
- Supports alarm time (hour, minute, second) and calendar (year, month, day) mask enable in RTC_TAMSK and RTC_CAMSK
- Selectable 12-hour or 24-hour time scale in RTC_CLKFMT register
- Supports Leap Year indication in RTC_LEAPYEAR register
- Supports Day of the Week counter in RTC_WEEKDAY register
- Frequency of RTC clock source compensate by RTC_FREQADJ register
- All time and calendar message expressed in BCD format
- Supports periodic RTC Time Tick interrupt with 8 period interval options 1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2 and 1 second
- Supports RTC Time Tick and Alarm Match interrupt
- Supports chip wake-up from Idle or Power-down mode while a RTC interrupt signal is generated

6.11.3 Block Diagram

The RTC block diagram is shown below.

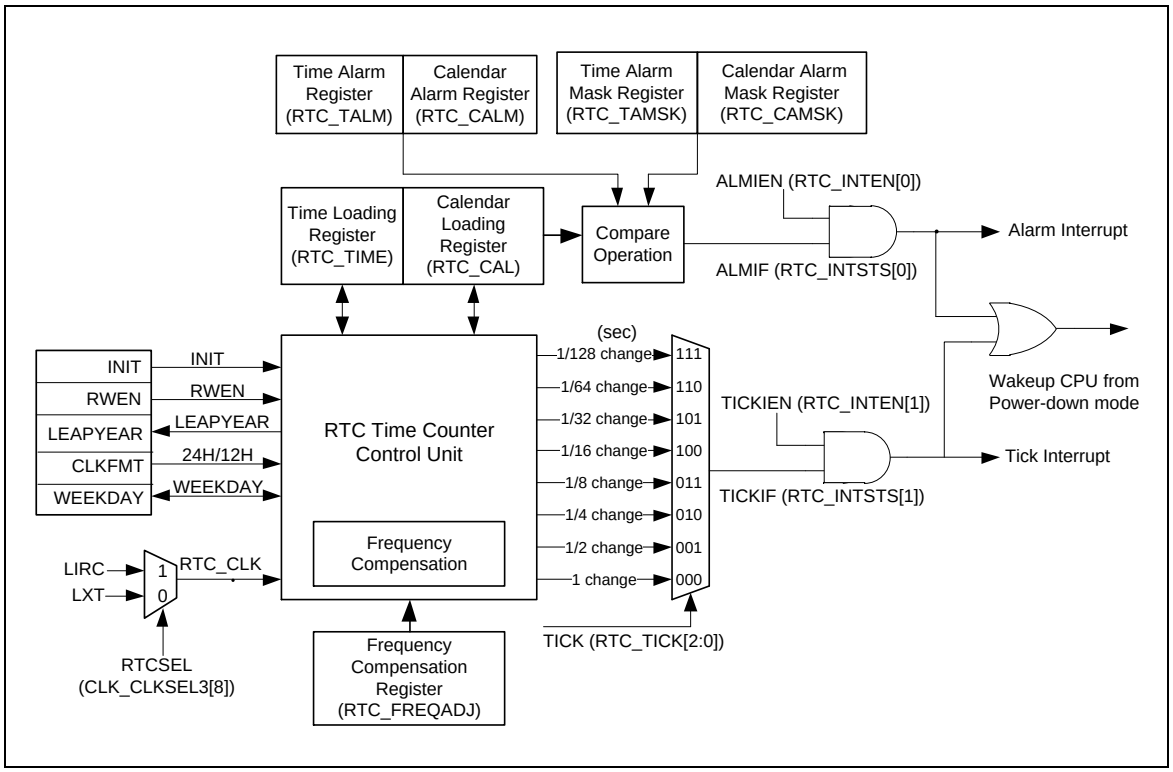


Figure 6.11-1 RTC Block Diagram

6.11.4 Basic Configuration

- Clock source configuration
  - The RTC controller clock source is enabled by RTCCKEN (APBCLK0[1]) and RTC Time Counter source is selected by CLK_CLKSEL3[8], the clock source can be LXT or LIRC.
- Pin configuration

Group	Pin Name	GPIO	MFP
X32	X32_OUT	PC.0	MFP2
	X32_IN	PC.1	MFP2

6.11.5 Functional Description

6.11.5.1 RTC Initiation

When a RTC block is power-on, RTC is at reset state. User has to write a number 0xa5eb1357 to RTC initial register INIT(RTC_INIT[31:0]) to make RTC leaving reset state. Once the INIT(RTC_INIT[31:0]) register is written as 0xa5eb1357, the RTC will be in normal active state permanently. User can read INIT[0](RTC_INIT[0]) to check the RTC is at normal active state or reset state.

6.11.5.2 RTC Read/Write Enable

If RWENF(RTC_RWEN[16]) bit is read as 1, it means the RTC registers are read/write

accessible. When executing write RTC register command exceed 6 times within 1120 PCLK cycles, the RTCBUSY(RTC_RWEN[24]) flag will be set 1 and RWENF(RTC_RWEN[16]) will be clear to 0. The RTC control registers access attribute when RWENF is 1 and 0 are shown in Table 6.11.5-1.

Register	INIT[0] = 0	RWENF = 1	RWENF = 0 Or RTCBUSY=1
RTC_INIT	available	R/W	R/W
RTC_RWEN	available	R/W	R/W
RTC_FREQADJ	available	R/W	R
RTC_TIME	Not available	R/W	R
RTC_CAL	Not available	R/W	R
RTC_CLKFMT	Not available	R/W	R
RTC_WEEKDAY	Not available	R/W	R
RTC_TALM	Not available	R/W	R
RTC_CALM	Not available	R/W	R
RTC_LEAPYEAR	Not available	R	R
RTC_INTEN	available	R/W	R/W
RTC_INTSTS	available	R/W	R/W
RTC_TICK	Not available	R/W	R
RTC_TAMSK	Not available	R/W	R
RTC_CAMSK	Not available	R/W	R
RTC_LXTCTL	available	R/W	R/W

Table 6.11.5-1 RTC Read/Write Enable

6.11.5.3 Frequency Compensation

The RTC_FREQADJ register allows user to make digital compensation to a clock input. Please follow the example and formula below to write the actual frequency of 32k crystal to RTC_FREQADJ register. Following are the compensation examples for higher or lower than 32768 Hz.

Example 1:

Frequency counter measurement : 32773.65 Hz (> 32768 Hz)

FREQADJ = (32768 * 0x200000) / 32773.65 = 0x1FFE96

Example 2:

Frequency counter measurement : 32763.25 Hz (< 32768 Hz)

FREQADJ = (32768 * 0x200000) / 32763.25 = 0x200130

Note: The value of RTC_FREQADJ register will be the default value (0x0020_0000) while the compensation is not executed. User can utilize a frequency counter to measure RTC clock source via clock output function in manufacturing. In the meanwhile, user can use clock output function to check the result of RTC frequency compensation.

6.11.5.4 Time and Calendar counter

RTC_TIME and RTC_CAL are used to load the real time and calendar. RTC_TALM and RTC_CALM are used for setup alarm time and calendar.

6.11.5.5 12/24 hour Time Scale Selection

The 12/24 hour time scale selection depends on 24HEN (RTC_CLKFMT[0]).

When RTC runs as 12-hour time scale mode, RTC_TIME[21] (the high bit of TENHR[1:0]) means AM/PM indication, if RTC_TIME[21] is 1, it indicates PM time message and RTC_TIME[21] is 0 indicates AM time message. Table 6.11.5-2 shows RTC_TIME mapping table of 12/24 hour time scale selection.

Note: The Hour Value Write Into RTC_TIME[21:16], Messages Are Expressed In BCD Format.			
24-Hour Time Scale (24HEN = 1)		12-Hour Time Scale (PM Time + 20) (24HEN = 0) (PM Time + 20)	
0x00 (AM12)	0x12 (PM12)	0x12 (AM12)	0x32 (PM12)
0x01 (AM01)	0x13 (PM01)	0x01 (AM01)	0x21 (PM01)
0x02 (AM02)	0x14 (PM02)	0x02 (AM02)	0x22 (PM02)
0x03 (AM03)	0x15 (PM03)	0x03 (AM03)	0x23 (PM03)
0x04 (AM04)	0x16 (PM04)	0x04 (AM04)	0x24 (PM04)
0x05 (AM05)	0x17 (PM05)	0x05 (AM05)	0x25 (PM05)
0x06 (AM06)	0x18 (PM06)	0x06 (AM06)	0x26 (PM06)
0x07 (AM07)	0x19 (PM07)	0x07 (AM07)	0x27 (PM07)
0x08 (AM08)	0x20 (PM08)	0x08 (AM08)	0x28 (PM08)
0x09 (AM09)	0x21 (PM09)	0x09 (AM09)	0x29 (PM09)
0x10 (AM10)	0x22 (PM10)	0x10 (AM10)	0x30 (PM10)
0x11 (AM11)	0x23 (PM11)	0x11 (AM11)	0x31 (PM11)

Table 6.11.5-2 12/24 hour Time Scale Selection

6.11.5.6 Day of the Week Counter

The RTC controller provides day of week in WEEKDAY bits (RTC_WEEKDAY[2:0]). The value is defined from 0 to 6 to represent Sunday to Saturday respectively.

6.11.5.7 Periodic Time Tick Interrupt

The periodic time tick interrupt has 8 period interval options 1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2 and 1 second that are selected by TICK bits (RTC_TICK[2:0]). When periodic time tick interrupt is enabled by setting TICKIEN (RTC_INTEN[1]) to 1, the periodic time tick interrupt is requested periodically in the period selected by RTC_TICK[2:0] settings.

6.11.5.8 Alarm Interrupt

When the real time and calendar message in RTC_TIME and RTC_CAL registers are equal to alarm time and calendar values in RTC_TALM and RTC_CALM registers, the RTC alarm interrupt flag ALMIF (RTC_INTSTS[0]) is set to 1 and the RTC alarm interrupt signal assert if the alarm interrupt enable ALMIEN (RTC_INTEN[0]) is enabled.

The RTC controller provides time alarm mask register (RTC_TAMSK register) and calendar alarm mask register (RTC_CAMSK register) to mask the specified digit and generate periodic interrupt without changing the alarm match condition in RTC_TALM and RTC_CALM registers in each alarm interrupt service routine.

6.11.5.9 Application Note

1. All data in RTC_TALM, RTC_CALM, RTC_TIME and RTC_CAL registers are all expressed in BCD format.
2. User has to make sure that the loaded values are reasonable. For example, Load RTC_CAL as 201a (year), 13 (month), 00 (day), or RTC_CAL does not match with RTC_WEEKDAY, etc.
3. In RTC_CAL and RTC_CALM, only 2 BCD digits are used to express “year”. The 2 BCD digits of xy means 20xy, rather than 19xy or 21xy.
4. Example of 12-Hour Time Setting

   If current RTC time is PM12:59:30 in 12-Hour Time Scale mode, the RTC_TIME setting as:

   HOUR:
   RTC_TIME[21:16]: 0x32 (0x12+0x20) combined by TENHR (RTC_TIME[21:20]) is 0x3, HR (RTC_TIME[19:16]) is 0x2.

   MIN:
   RTC_TIME[14:8]: 0x59 combined by TENMIN (RTC_TIME[14:12]) is 0x5, MIN (RTC_TIME[11:8]) is 0x9.

   SEC:
   RTC_TIME[6:0]: 0x30 combined by TENSEC (RTC_TIME[6:4]) is 0x3, SEC (RTC_TIME[3:0]) is 0x0.
5. Registers value after powered on:

Register	Reset State
RTC_INIT	0
RTC_RWEN	0
RTC_CAL	15/8/8 (year/month/day)
RTC_TIME	00:00:00 (hour : minute : second)
RTC_CALM	00/00/00 (year/month/day)
RTC_TALM	00:00:00 (hour : minute : second)
RTC_CLKFMT	1 (24-hour mode)
RTC_WEEKDAY	6 (Saturday)
RTC_INTEN	0
RTC_INTSTS	0
RTC_LEAPYEAR	0
RTC_TICK	0

Table 6.11.5-3 Registers value after power-on

6.11.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
RTC Base Address: **RTC_BA = 0x4004_1000**				
RTC_INIT	RTC_BA+0x00	R/W	RTC Initiation Register	0x0000_0000
RTC_RWEN	RTC_BA+0x04	R/W	RTC Access Enable Register	0x0000_0000
RTC_FREQADJ	RTC_BA+0x08	R/W	RTC Frequency Compensation Register	0x0020_0000
RTC_TIME	RTC_BA+0x0C	R/W	RTC Time Loading Register	0x0000_0000
RTC_CAL	RTC_BA+0x10	R/W	RTC Calendar Loading Register	0x0015_0808
RTC_CLKFMT	RTC_BA+0x14	R/W	RTC Time Scale Selection Register	0x0000_0001
RTC_WEEKDAY	RTC_BA+0x18	R/W	RTC Day of the Week Register	0x0000_0006
RTC_TALM	RTC_BA+0x1C	R/W	RTC Time Alarm Register	0x0000_0000
RTC_CALM	RTC_BA+0x20	R/W	RTC Calendar Alarm Register	0x0000_0000
RTC_LEAPYEAR	RTC_BA+0x24	R	RTC Leap Year Indicator Register	0x0000_0000
RTC_INTEN	RTC_BA+0x28	R/W	RTC Interrupt Enable Register	0x0000_0000
RTC_INTSTS	RTC_BA+0x2C	R/W	RTC Interrupt Status Register	0x0000_0000
RTC_TICK	RTC_BA+0x30	R/W	RTC Time Tick Register	0x0000_0000
RTC_TAMSK	RTC_BA+0x34	R/W	RTC Time Alarm Mask Register	0x0000_0000
RTC_CAMSK	RTC_BA+0x38	R/W	RTC Calendar Alarm Mask Register	0x0000_0000
RTC_LXTCTL	RTC_BA+0x100	R/W	RTC 32.768 kHz Oscillator Control Register	0x0000_000E

Note:

1. Any register not listed here is reserved and must not be written. The result of a read operation on these bits is undefined.
2. The reserved register fields that listed in register description must be written to their reset value. Writing reserved fields with other than reset values may produce undefined results.

6.11.7 Register Description

RTC Initiation Register (RTC_INIT)

Register	Offset	R/W	Description	Reset Value
RTC_INIT	RTC_BA+0x00	R/W	RTC Initiation Register	0x0000_0000

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

Bits	Description	
[31:1]	**INIT[31:1]**	**RTC Initiation (Write Only)** When RTC block is powered on, RTC is at reset state. User has to write a number (0x a5eb1357) to INIT to make RTC leaving reset state. Once the INIT is written as 0xa5eb1357, the RTC will be in un-reset state permanently. The INIT is a write-only field and read value will be always 0.
[0]	**INIT[0]/ACTIVE**	**RTC Active Status (Read Only)** 0 = RTC is at reset state. 1 = RTC is at normal active state.

RTC Access Enable Register (RTC_RWEN)

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

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

Bits	Description	
[31:25]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[24]	**RTCBUSY**	**RTC Write Busy Flag** This bit indicates RTC registers are writable or not. 0: RTC registers are writable. 1: RTC registers can't write, RTC under Busy Status. **Note:** RTCBUSY flag will be set when execute write RTC register command exceed 6 times within 1120 PCLK cycles.
[23:17]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[16]	**RWENF**	**RTC Register Access Enable Flag (Read Only)** 0 = RTC register read/write Disabled. 1 = RTC register read/write Enabled. **Note:** RWENF will be mask to 0 during RTCBUSY is 1, and first turn on RTCCKEN (CLK_APBCLK[1]) also.
[15:0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

RTC Frequency Compensation Register (RTC_FREQADJ)

Register	Offset	R/W	Description	Reset Value
RTC_FREQADJ	RTC_BA+0x08	R/W	RTC Frequency Compensation Register	0x0020_0000

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

Bits	Description	
[31:22]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[21:0]	**FREQADJ**	**Frequency Compensation Register** User must to get actual LXT frequency for RTC application. FREQADJ = 0x200000 * (32768 / LXT frequency). **Note:** This formula is suitable only when RTC clock source is from LXT, RTCSEL (CLK_CLKSEL3[8]) is 0.

RTC Time Loading Register (RTC_TIME)

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

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

Bits	Description	
[31:22]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[21:20]	**TENHR**	10-Hour Time Digit (0~2) When RTC runs as 12-hour time scale mode, RTC_TIME[21] (the high bit of TENHR[1:0]) means AM/PM indication (If RTC_TIME[21] is 1, it indicates PM time message.)
[19:16]	**HR**	1-Hour Time Digit (0~9)
[15]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[14:12]	**TENMIN**	10-Min Time Digit (0~5)
[11:8]	**MIN**	1-Min Time Digit (0~9)
[7]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[6:4]	**TENSEC**	10-Sec Time Digit (0~5)
[3:0]	**SEC**	1-Sec Time Digit (0~9)

Note:

1. RTC_TIME is a BCD digit counter and RTC will not check loaded data.
2. The reasonable value range is listed in the parenthesis.

RTC Calendar Loading Register (RTC_CAL)

Register	Offset	R/W	Description	Reset Value
RTC_CAL	RTC_BA+0x10	R/W	RTC Calendar Loading Register	0x0015_0808

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

Bits	Description	
[31:24]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[23:20]	**TENYEAR**	10-Year Calendar Digit (0~9)
[19:16]	**YEAR**	1-Year Calendar Digit (0~9)
[15:13]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[12]	**TENMON**	10-Month Calendar Digit (0~1)
[11:8]	**MON**	1-Month Calendar Digit (0~9)
[7:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5:4]	**TENDAY**	10-Day Calendar Digit (0~3)
[3:0]	**DAY**	1-Day Calendar Digit (0~9)

Note:

1. RTC_CAL is a BCD digit counter and RTC will not check loaded data.
2. The reasonable value range is listed in the parenthesis.

RTC Time Scale Selection Register (RTC_CLKFMT)

Register	Offset	R/W	Description	Reset Value
RTC_CLKFMT	RTC_BA+0x14	R/W	RTC Time Scale Selection Register	0x0000_0001

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

Bits	Description	
[31:1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**24HEN**	**24-hour / 12-hour Time Scale Selection** Indicates that RTC_TIME and RTC_TALM are in 24-hour time scale or 12-hour time scale 0 = 12-hour time scale with AM and PM indication selected. 1 = 24-hour time scale selected.

RTC Day of the Week Register (RTC_WEEKDAY)

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

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**WEEKDAY**	**Day of the Week Register** 000 = Sunday. 001 = Monday. 010 = Tuesday. 011 = Wednesday. 100 = Thursday. 101 = Friday. 110 = Saturday. 111 = Reserved. Do not use.

RTC Time Alarm Register (RTC_TALM)

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

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

Bits	Description	
[31:22]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[21:20]	**TENHR**	10-Hour Time Digit of Alarm Setting (0~2) When RTC runs as 12-hour time scale mode, RTC_TIME[21] (the high bit of TENHR[1:0]) means AM/PM indication (If RTC_TIME[21] is 1, it indicates PM time message.)
[19:16]	**HR**	1-Hour Time Digit of Alarm Setting (0~9)
[15]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[14:12]	**TENMIN**	10-Min Time Digit of Alarm Setting (0~5)
[11:8]	**MIN**	1-Min Time Digit of Alarm Setting (0~9)
[7]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[6:4]	**TENSEC**	10-Sec Time Digit of Alarm Setting (0~5)
[3:0]	**SEC**	1-Sec Time Digit of Alarm Setting (0~9)

Note:

1. RTC_TALM is a BCD digit counter.
2. The reasonable value range is listed in the parenthesis.
3. This register can be read back and written after the RTC register access enable bit RWENF (RTC_RWEN[16]) is active.

RTC Calendar Alarm Register (RTC_CALM)

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

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

Bits	Description	
[31:24]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[23:20]	**TENYEAR**	10-Year Calendar Digit of Alarm Setting (0~9)
[19:16]	**YEAR**	1-Year Calendar Digit of Alarm Setting (0~9)
[15:13]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[12]	**TENMON**	10-Month Calendar Digit of Alarm Setting (0~1)
[11:8]	**MON**	1-Month Calendar Digit of Alarm Setting (0~9)
[7:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5:4]	**TENDAY**	10-Day Calendar Digit of Alarm Setting (0~3)
[3:0]	**DAY**	1-Day Calendar Digit of Alarm Setting (0~9)

Note:

1. RTC_CALM is a BCD digit counter.
2. The reasonable value range is listed in the parenthesis.
3. This register can be read back and written after the RTC register access enable bit RWENF (RTC_RWEN[16]) is active.

RTC Leap Year Indication Register (RTC_LEAPYEAR)

Register	Offset	R/W	Description	Reset Value
RTC_LEAPYEAR	RTC_BA+0x24	R	RTC Leap Year Indicator Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**LEAPYEAR**	**Leap Year Indication Register (Read Only)** 0 = This year is not a leap year. 1 = This year is leap year.

RTC Interrupt Enable Register (RTC_INTEN)

Register	Offset	R/W	Description	Reset Value
RTC_INTEN	RTC_BA+0x28	R/W	RTC Interrupt Enable Register	0x0000_0000

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

Bits	Description	
[31:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1]	**TICKIEN**	**Time Tick Interrupt Enable Bit** Set TICKIEN to 1 can also enable chip wake-up function when RTC tick interrupt event is generated. 0 = RTC Time Tick interrupt Disabled. 1 = RTC Time Tick interrupt Enabled.
[0]	**ALMIEN**	**Alarm Interrupt Enable Bit** Set ALMIEN to 1 can also enable chip wake-up function when RTC alarm interrupt event is generated. 0 = RTC Alarm interrupt Disabled. 1 = RTC Alarm interrupt Enabled.

RTC Interrupt Status Register (RTC_INTSTS)

Register	Offset	R/W	Description	Reset Value
RTC_INTSTS	RTC_BA+0x2C	R/W	RTC 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						**TICKIF**	**ALMIF**

Bits	Description	
[31:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1]	**TICKIF**	**RTC Time Tick Interrupt Flag** 0 = Tick condition does not occur. 1 = Tick condition occur. **Note:** Write 1 to clear this bit.
[0]	**ALMIF**	**RTC Alarm Interrupt Flag** 0 = Alarm condition is not matched. 1 = Alarm condition is matched. **Note:** Write 1 to clear this bit.

RTC Time Tick Register (RTC_TICK)

Register	Offset	R/W	Description	Reset Value
RTC_TICK	RTC_BA+0x30	R/W	RTC Time Tick Register	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:0]	**TICK**	**Time Tick Register** These bits are used to select RTC time tick period for Periodic Time Tick Interrupt request. 000 = Time tick is 1 second. 001 = Time tick is 1/2 second. 010 = Time tick is 1/4 second. 011 = Time tick is 1/8 second. 100 = Time tick is 1/16 second. 101 = Time tick is 1/32 second. 110 = Time tick is 1/64 second. 111 = Time tick is 1/128 second. **Note:** This register can be read back and written after the RTC register access enable bit RWENF (RTC_RWEN[16]) is active.

Note: This register can be read back and written after the RTC register access enable bit RWENF (RTC_RWEN[16]) is active.

RTC Time Alarm MASK Register (RTC_TAMSK)

Register	Offset	R/W	Description	Reset Value
RTC_TAMSK	RTC_BA+0x34	R/W	RTC Time Alarm Mask Register	0x0000_0000

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

Bits	Description	
[31:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5]	**MTENHR**	Mask 10-Hour Time Digit of Alarm Setting (0~2)
[4]	**MHR**	Mask 1-Hour Time Digit of Alarm Setting (0~9)
[3]	**MTENMIN**	Mask 10-Min Time Digit of Alarm Setting (0~5)
[2]	**MMIN**	Mask 1-Min Time Digit of Alarm Setting (0~9)
[1]	**MTENSEC**	Mask 10-Sec Time Digit of Alarm Setting (0~5)
[0]	**MSEC**	Mask 1-Sec Time Digit of Alarm Setting (0~9)

Note:

1. RTC_TALM is a BCD digit counter and RTC will not check loaded data.
2. The reasonable value range is listed in the parenthesis.
3. MTENHR/MHR base on 24 hour Time Scale.

RTC Calendar Alarm MASK Register (RTC_CAMSK)

Register	Offset	R/W	Description	Reset Value
RTC_CAMSK	RTC_BA+0x38	R/W	RTC Calendar Alarm Mask Register	0x0000_0000

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

Bits	Description	
[31:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5]	**MTENYEAR**	Mask 10-Year Calendar Digit of Alarm Setting (0~9)
[4]	**MYEAR**	Mask 1-Year Calendar Digit of Alarm Setting (0~9)
[3]	**MTENMON**	Mask 10-Month Calendar Digit of Alarm Setting (0~1)
[2]	**MMON**	Mask 1-Month Calendar Digit of Alarm Setting (0~9)
[1]	**MTENDAY**	Mask 10-Day Calendar Digit of Alarm Setting (0~3)
[0]	**MDAY**	Mask 1-Day Calendar Digit of Alarm Setting (0~9)

Note:

1. RTC_CALM is a BCD digit counter and RTC will not check loaded data.
2. The reasonable value range is listed in the parenthesis.

RTC 32K Oscillator Control Register (RTC_LXTCTL)

Register	Offset	R/W	Description	Reset Value
RTC_LXTCTL	RTC_BA+0x100	R/W	RTC 32.768 kHz Oscillator Control Register	0x0000_000E

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

Bits	Description	
[31:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2:1]	**GAIN**	**Oscillator Gain Option** User can select oscillator gain according to crystal external loading and operating temperature range. The larger gain value corresponding to stronger driving capability and higher power consumption. 00 = L0 mode. 01 = L1 mode. 10 = L2 mode. 11 = L3 mode.
[0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

6.12 UART Interface Controller (UART)

6.12.1 Overview

The NPCA121 series is equipped with one Universal Asynchronous Receiver/Transmitters (UART) port, which offers a mean of full-duplex asynchronous communication with external device.

The NPCA121 series UART controller also supports RS-485 standard.

6.12.2 Features

- One UART port: UART0.
- Programmable baud-rate generator
- Separate receive (RX) and transmit (TX) FIFOs with 16 bytes each to reduce CPU interrupt service loading
- RX FIFO trigger level of 1/16, 4/16, 8/16 and 14/16.
- Supports hardware auto-flow control
- Supports wake-up function which can be triggered by nCTS, incoming data, RX FIFO reached threshold or RS-485 Address Match (AAD mode).
- Supports 8-bit RX FIFO 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
  - 5, 6, 7, or 8 data bits
  - even, odd, stick or no-parity generation/detection
  - 1, 1.5, or 2 stop bit generation
- Support PDMA transfer function
- Supports RS-485 function mode
  - RS-485 9-bit mode
  - hardware or software managing nRTS pin to control RS-485 transmission direction

UART Feature	UART0
FIFO	16 Bytes
Auto Flow Control (CTS/RTS)	√
RS-485 Function Mode	√
nCTS Wake-up	√

Incoming Data Wake-up	√
RX FIFO reached threshold Wake-up	√
RS-485 Address Match (AAD mode) Wake-up	√
Auto-Baud Rate Measurement	√
STOP Bit Length	1, 1.5, 2 bit
Word Length	5, 6, 7, 8 bits
Even / Odd Parity	√
Stick Bit	√
√= Supported	

Table 6.12.2-1 UART Feature

6.12.3 Block Diagram

The UART clock control and block diagram are shown in Figure 6.12-1 and Figure 6.12-2 respectively.

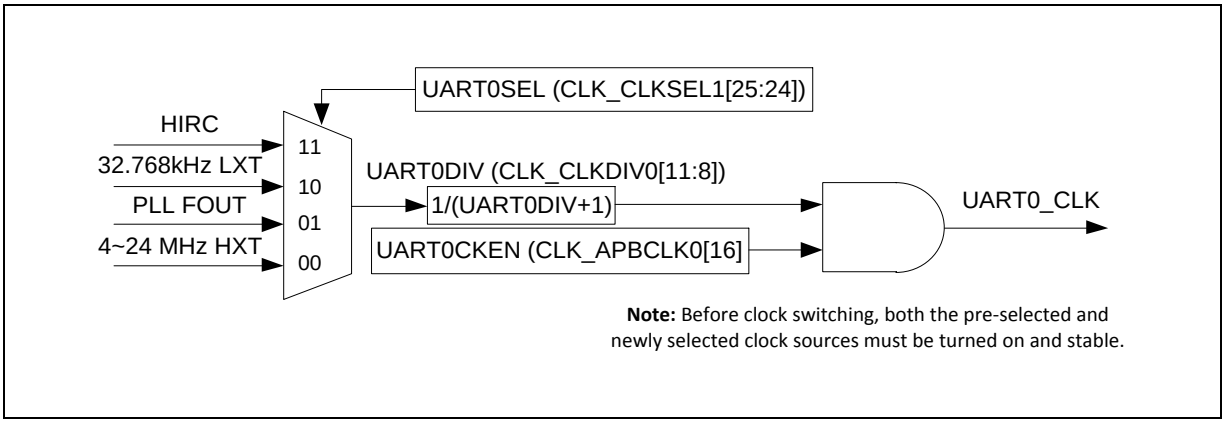


Figure 6.12-1 UART Clock Control Diagram

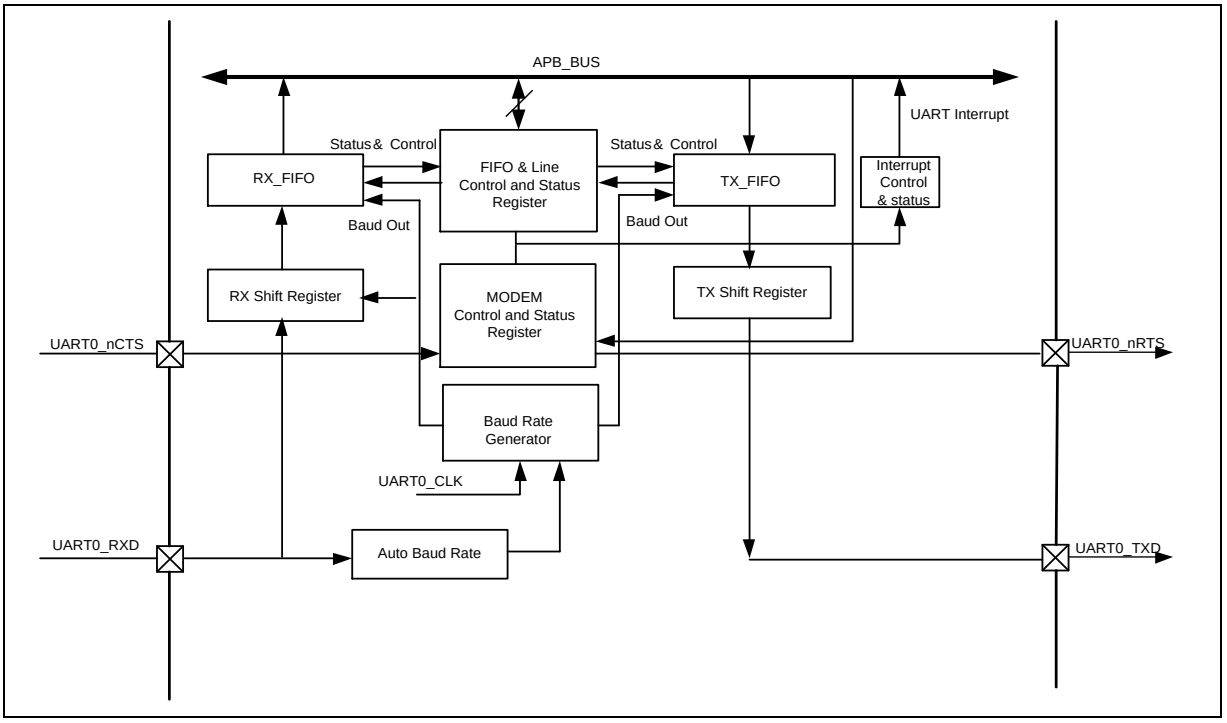


Figure 6.12-2 UART Block Diagram

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.

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, shown in Table 6.12.3-1. Write into UART Interrupt Enable Register (UART_INTEN) to enable/disable interrupt(s), and check UART Interrupt Status Register (UART_INTSTS) to identify the source of a UART 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.
WKINT	Wake-up Interrupt.
ABRINT	Auto-Baud Rate Interrupt.

Table 6.12.3-1 UART Interrupt

6.12.4 Basic Configuration

- Clock source configuration
  - Select the source of UART0 peripheral clock on UART0SEL (CLK_CLKSEL1[25:24]).
  - Select the clock divider number of UART0 peripheral clock on UART0DIV (CLK_CLKDIV0[11:8]).
  - Enable UART0 peripheral clock in UART0CKEN (CLK_APBCLK1[16]).
- Reset configuration
  - Reset UART0 controller in UART0RST (SYS_IPRST1[16]).
- Pin configuration

Group	Pin Name	GPIO	MFP
UART0	UART0_RXD	PA.8	MFP1
		PB.3	MFP4
		PB.9	MFP1
		PD.12	MFP1
		PD.15	MFP4
	UART0_TXD	PA.7	MFP1
		PB.4	MFP4
		PB.8	MFP1
		PD.11	MFP1
		PD.14	MFP4
	UART0_nCTS	PB.4	MFP1
		PD.14	MFP1
	UART0_nRTS	PB.7	MFP1
		PD.15	MFP1

UART Interface Controller Pin description is shown in Table 6.12.4-1:

Pin	Type	Description
UART0_TXD	Output	UART0 transmit
UART0_RXD	Input	UART0 receive

UART0_nCTS	Input	UART0 modem clear to send
UART0_nRTS	Output	UART0 modem request to send

Table 6.12.4-1 UART Interface Controller Pin

6.12.5 Functional Description

The NPCA121 series UART controller supports four function modes including UART and RS-485. The current function mode is selected by UART_FUNCSEL register.

6.12.5.1 UART Controller Baud Rate Generator

The NPCA121 series device integrates a programmable baud rate generator.

Table 6.12.5-1 describes the baud-rate calculation. Table 6.12.5-2 and Table 6.12.5-3 gives a quick reference for available baud rates under different mode and BRD setting. In these table:

- There are three modes for baud rate setting. Baud rate mode is defined by BAUDM0 and BAUDM1 bits (UART_BAUD[29:28]).
- BRD value in the formula is defined by BRD bits in register UART_BAUD[15:0].

Mode	BAUDM0	BAUDM1	Baud Rate Equation
Mode 0	0	0	UART0_CLK / [16 * (BRD+2)].
Mode 1	0	1	UART0_CLK / [(EDIVM1+1) * (BRD+2)], EDIVM1 must >= 8.
Mode 2	1	1	UART0_CLK / (BRD+2) If UART0_CLK <= 3*HCLK, BRD must >= 9. If UART0_CLK > 3*HCLK, BRD must >= 3*N – 1. N is the smallest integer larger than or equal to the ratio of UART0_CLK /HCLK. For example, if 3*HCLK < UART0_CLK =< 4*HCLK, BRD must >=11. if 4*HCLK < UART0_CLK =< 5*HCLK, BRD must >=14. (If the UART0_CLK is selected from LXT, BRD can be greater than or equal to 1)

Table 6.12.5-1 UART controller Baud Rate Equation Table

UART Peripheral Clock = 12 MHz			
Baud Rate	**Mode 0**	**Mode 1**	**Mode 2**
921600	Not support	Not recommended	BRD=11
460800	Not recommended	BRD=0, EDIVM1 =13	BRD=24
230400	Not recommended	BRD =2, EDIVM1 =13	BRD =50
115200	Not recommended	BRD =6, EDIVM1 =13	BRD =102

57600	BRD =11	BRD =14, EDIVM1 =13	BRD =206
38400	BRD =18	BRD =22, EDIVM1 =13	BRD =311
19200	BRD =37	BRD =123, EDIVM1 =5	BRD =623
9600	BRD =76	BRD =123, EDIVM1 =10	BRD =1248
4800	BRD =154	BRD =248, EDIVM1 =10	BRD =2498

Table 6.12.5-2 UART controller Baud Rate Parameter Setting Example Table

UART Peripheral Clock = 12 MHz			
Baud Rate	**UART_BAUD Value**		
	Mode 0	**Mode 1**	**Mode 2**
921600	Not support	Not recommended	0x3000_000B
460800	Not recommended	0x2D00_0000	0x3000_0018
230400	Not recommended	0x2D00_0002	0x3000_0032
115200	Not recommended	0x2D00_0006	0x3000_0066
57600	0x0000_000B	0x2D00_000E	0x3000_00CE
38400	0x0000_0012	0x2D00_0016	0x3000_0137
19200	0x0000_0025	0x2500_007B	0x3000_026F
9600	0x0000_004C	0x2A00_007B	0x3000_04E0
4800	0x0000_009A	0x2A00_00F8	0x3000_09C2

Table 6.12.5-3 UART controller Baud Rate Register Setting Example Table

6.12.5.2 UART Controller Baud Rate Compensation

The NPCA121 series UART controller supports baud rate compensation function. It is used to optimize the precision for each bit. Two examples are given below to explain the compensation mechanism.

- BRCOMPDEC bit (UART_BRCOMP[31])
  - = 0: positive compensation, increase one UART clock cycle,
  - = 1: negative compensation, decrease one UART clock cycle.
- BRCOMP bits (UART_BRCOMP[8:0]: for each bit in BRCOMP, '1' means to do compensation, while '0' means not to do compensation.

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
	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 6.12.5-4 Baud Rate Compensation Example Table 1

So that 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
	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 6.12.5-5 Baud Rate Compensation Example Table 2

So that the BRCOMP (UART_BRCOMP[8:0]) can be set as 9'b010000010 = 0x82.

6.12.5.3 UART Controller Auto-Baud Rate Function Mode

The NPCA121 Auto-Baud Rate function measures the baud rate of in-coming data from UART RX pin automatically. When the Auto-Baud Rate measurement is finished, the measuring baud rate is loaded into BRD (UART_BAUD[15:0]).

Both 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 1st 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.

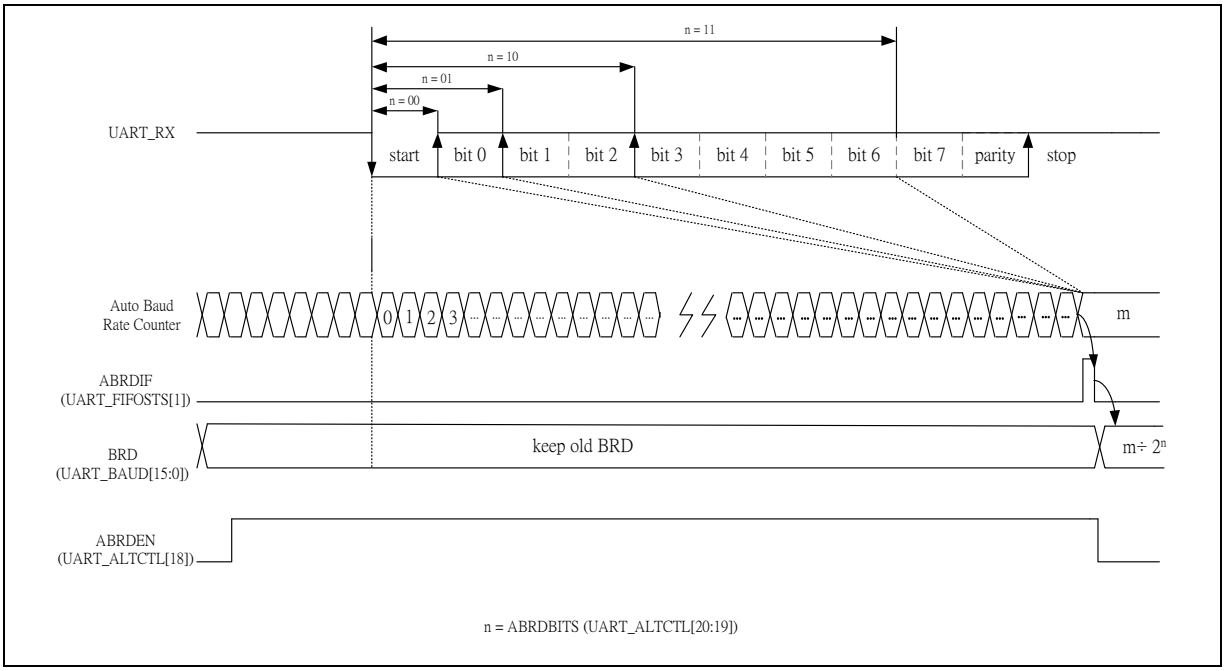


Figure 6.12-3 Auto-Baud Rate Measurement

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.

6.12.5.4 UART Controller Transmit Delay Time Value

By configuring DLY (UART_TOUT [15:8]), transfer delay time can be added between the last stop bit and next start bit in transmission, shown in Figure 6.12-4. The unit is baud.

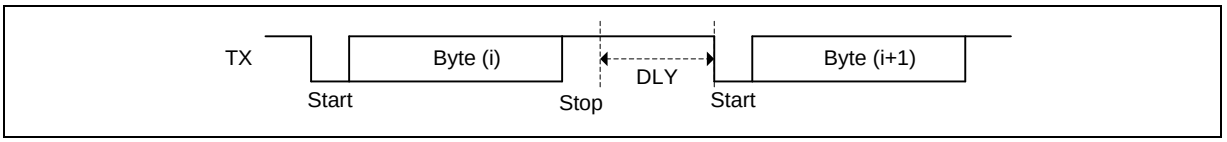


Figure 6.12-4 Transmit Delay Time Operation

6.12.5.5 UART Controller FIFO Control and Status

The UART controller has built-in 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 FIFO any time during operation. The information presented in UART_FIFOSTS register includes RX/TX FIFO status, pointer location, error status, etc. UART, and RS-485 mode all support FIFO control and status functions.

6.12.5.6 UART Controller Wake-up Function

The NPCA121 series UART controller supports wake-up function. The wake-up source be

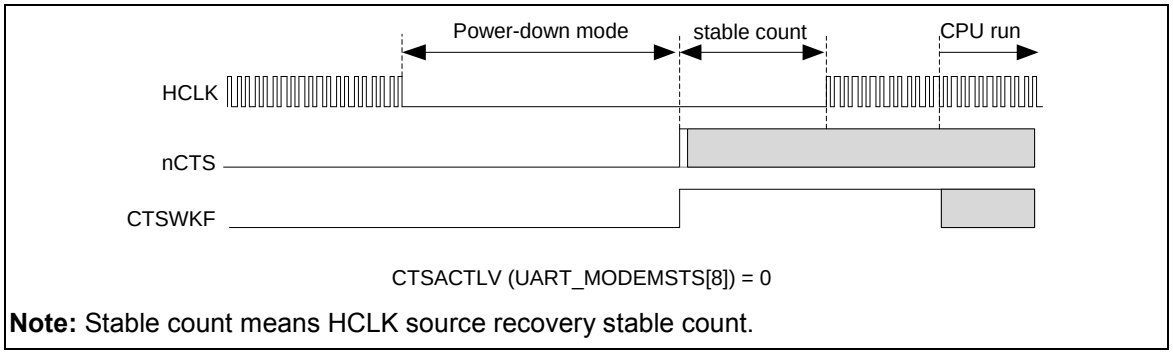
- nCTS pin
- incoming data
- RX FIFO reaching threshold
- RS-485 address matching
- RX FIFO threshold time-out

The UART wake-up source can be identified by checking UART_WKSTS register. If wake-up interrupt enable bit WKIEN (UART_INTEN[6]) is enable, UART wake-up interrupt will be generated and WKIF bit(UART_INTSTS[6]) will be set.

nCTS pin wake-up :

When system is in power down mode and WKCTSEN (UART_WKCTL[0]) is set, toggling on UART0_nCTS pin wakes up the system, and CTSWKF (UART_WKSTS[0]) will be set to 1.

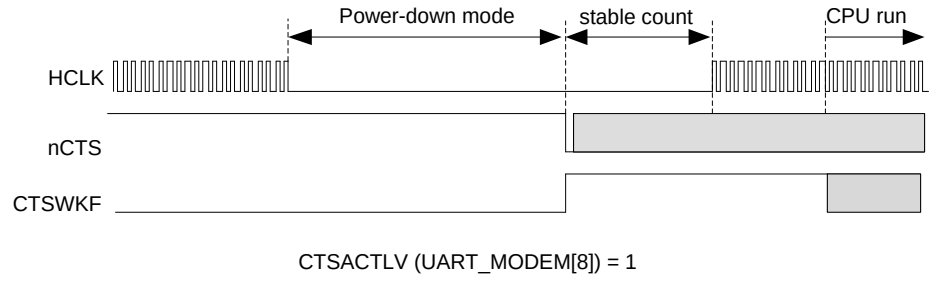
nCTS Wake-up Case 1 (nCTS transition from low to high)



Note: Stable count means HCLK source recovery stable count.

Figure 6.12-5 UART nCTS Wake-up Case1

nCTS Wake-up Case 2 (nCTS transition from high to low)



Note: Stable count means HCLK source recovery stable count.

Figure 6.12-6 UART nCTS Wake-up Case2

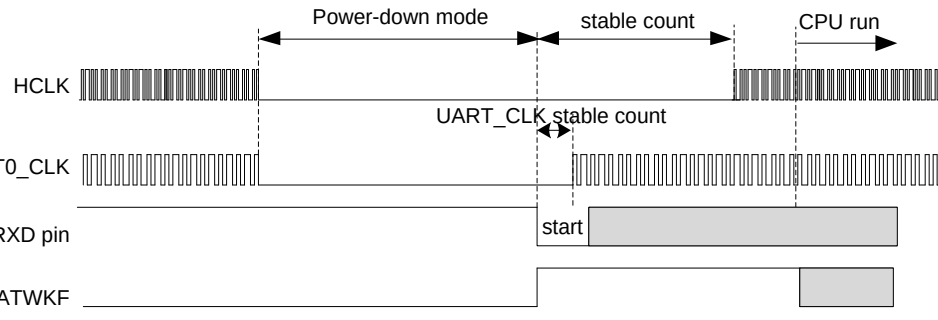
Incoming data wake-up:

When system is in Power-down mode and the WKDATEN (UART_WKCTL [1]) is set, toggling on incoming data (UART0_RXD) pin wakes up the system, and DATWKF (UART_WKSTS[1]) will be set to 1.

In order to correctly receive the incoming data after system wake-up, the Start bit Compensation Value STCOMP (UART_DWKCOMP[15:0]) shall be set in advance. STCOMP bits indicate how many UART0_CLK cycles the UART controller can have to become stable and be ready to receive 1st bit (start bit) after wakeup.

Note1: The UART controller clock source should be selected from HIRC and the compensation time for start bit is about 15.68us. It means that the value of STCOMP (UART_DWKCOMP[15:0]) can be set as 347.

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: UART0_CLK stable count means UART clock source recovery stable count.

Figure 6.12-7 UART Data Wake-up

RX FIFO reaching threshold wake-up:

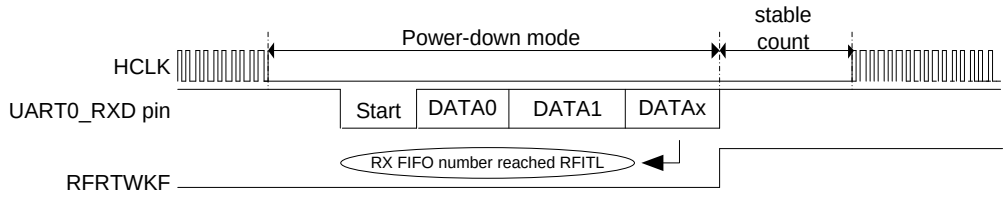
To setup the RX FIFO Reached Threshold Wake-up function, configure the following bits:

- WKRFRTEN (UART_WKCTL[2]): RX FIFO Reached Threshold Wake-up Enable bit
- RFITL (UART_FIFO[7:4]): RX FIFO Interrupt Trigger level

In power down mode, an event that the number of received data in RX FIFO reaches the threshold value RFITL (UART_FIFO[7:4]) can wakeup the system, and flag RFRTWKF

(UART_WKSTS[2]) will be set.

Note: The UART controller clock source should be choose LXT in power down mode for data receiving.



Note: Stable count means HCLK source recovery stable count.

Figure 6.12-8 UART RX FIFO reached threshold wake-up

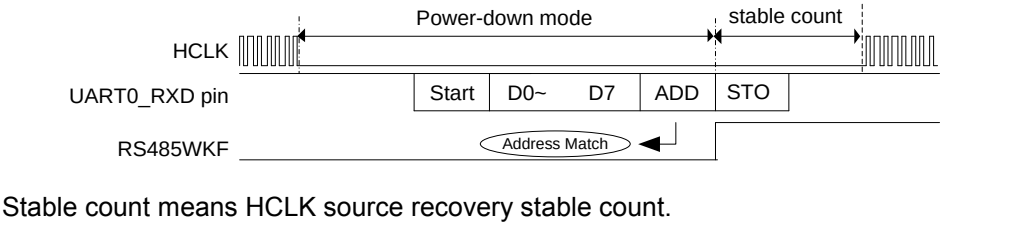
RS-485 Address Matching (AAD mode) wake-up:

Enable the following bits to setup RS-485 address matching wake-up function:

- WKRS485EN (UART_WKCTL[3]): RS-485 Address Match Wake-up Enable bit
- ADDRDEN (UART_ALTCTL[15]): Rs-485 Address Detection Enable bit

In Power-down mode, when an address byte is detected matching ADDRMV (UART_ALTCTL[31:24]), flag RS485WKF (UART_WKSTS[3]) will be raised and thus wakes up the system.

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 6.12-9 UART RS-485 AAD Mode Address Match Wake-up

RX FIFO threshold time-out wake-up:

RX FIFO Threshold Time-out Wake-up function can wake up the system if the device has not received certain amount of data within specified time.

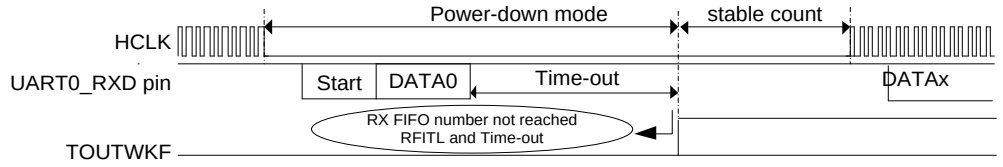
To enable RX FIFO Time-out Wake-up function:

- Enable WKRFRTEN (UART_WKCTL[2]): RX FIFO Reached Threshold Wake-up Enable bit.
- Enable WKTOUTEN (UART_WKCTL[4]): RX FIFO Time-out Wake-up Enable bit
- Enable TOCNTEN (UART_INTEN[11]): RX FIFO Time-out Counter Enable bit
- Configure RFITL (UART_FIFO[7:4]): RX FIFO Interrupt Trigger level
- Configure TOIC (UART_TOUT[7:0]): Time-out Interrupt Comparator

Under this configuration, the time-out counter resets and starts counting incrementally whenever RX FIFO receives new data. The counting clock frequency equals to the baud rate. When time-out counter value equals to time-out value defined in TOIC, and the RX FIFO level does not reach the threshold value defined in RFITL, it wakes up the system from power down mode, and flag

TOUTWKF(UART _WKSTS[4]) will be set.

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 6.12-10 UART RX FIFO threshold time-out wake-up

6.12.5.7 UART Controller Interrupt and Status

The NPCA121 series UART controller supports ten kinds of interrupts listed below:

- 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)
- Wake-up Interrupt (WKINT)
  - nCTS Wake-up Flag (CTSWKF)
  - Incoming Data Wake-up Flag (DATWKF)
  - RX FIFO Reached Threshold Wake-up Flag (RFRTWKF)
  - RS-485 Address Match (AAD mode) Wake-up Flag (RS485WKF)
  - RX 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 6.12.5-6 describes the interrupt sources and flags. When an interrupt occurred, its corresponding flag will be raised (set). Software should clear the interrupt flag after the interrupt is generated. Writing 1 (to flag) clears the interrupt flag.

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
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 6.12.5-6 UART controller Interrupt Source and Flag List

6.12.5.8 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 921 Kbps.

The UART provides full-duplex and asynchronous communications. The transmitter and receiver each has 16-byte FIFO. 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 via UART_LINE register. Table 6.12.5-7 and Table 6.12.5-8 provide a quick reference for UART port format setting.

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 6.12.5-7 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 Space Parity	1	1	0	1	Parity bit always logic 0. Parity bit on the serial byte is set to “0” regardless of total number of “1’s” (even or odd counts).

Table 6.12.5-8 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 an 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. The auto flow control block diagram shows in Figure 6.12-11.

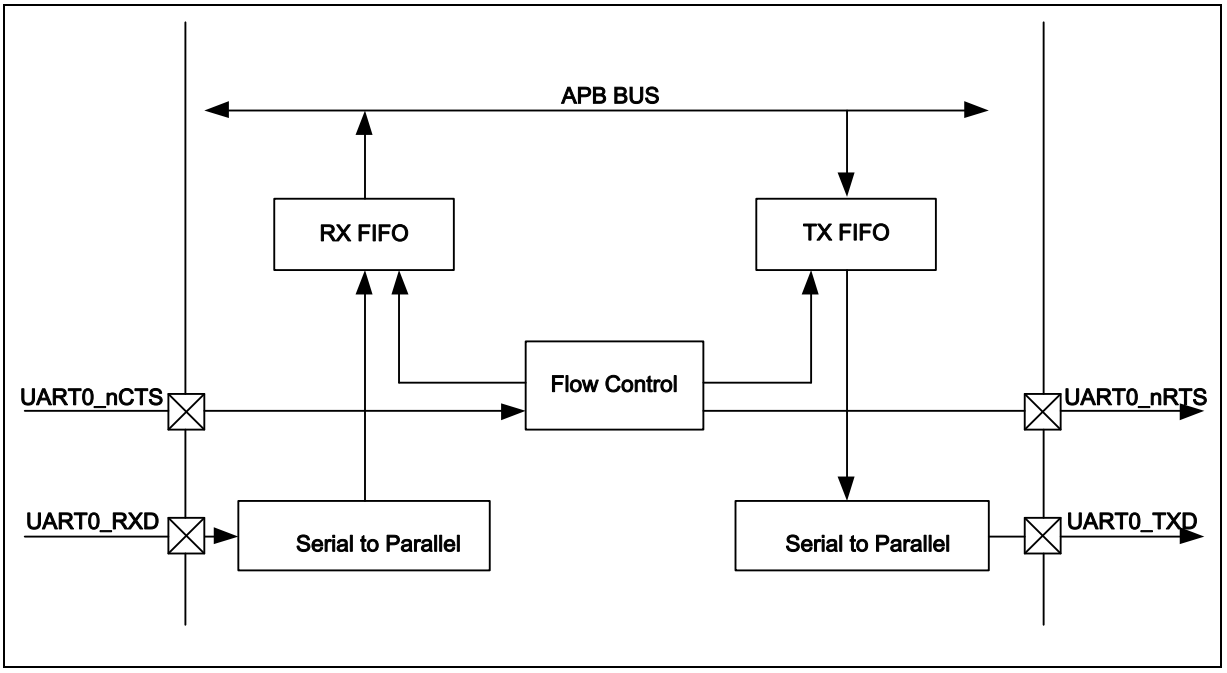


Figure 6.12-11 Auto-Flow Control Block Diagram

Figure 6.12-12 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 UART0_nCTS pin input has occurred, and then TX data will be automatically transmitted from TX FIFO.

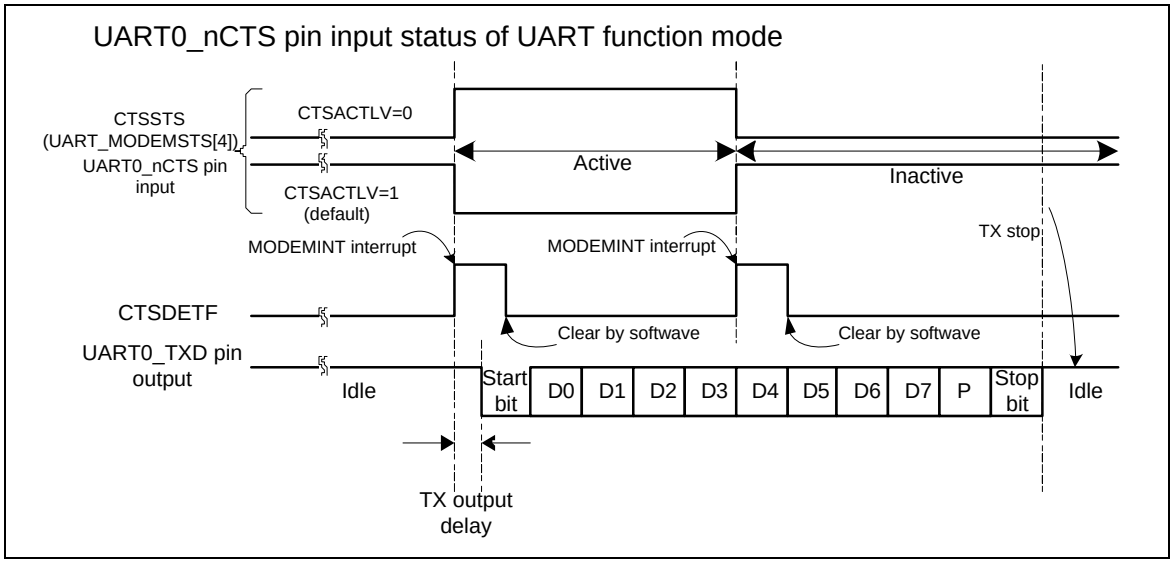


Figure 6.12-12 UART nCTS Auto-Flow Control Enabled

As shown in Figure 6.12-13, 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.

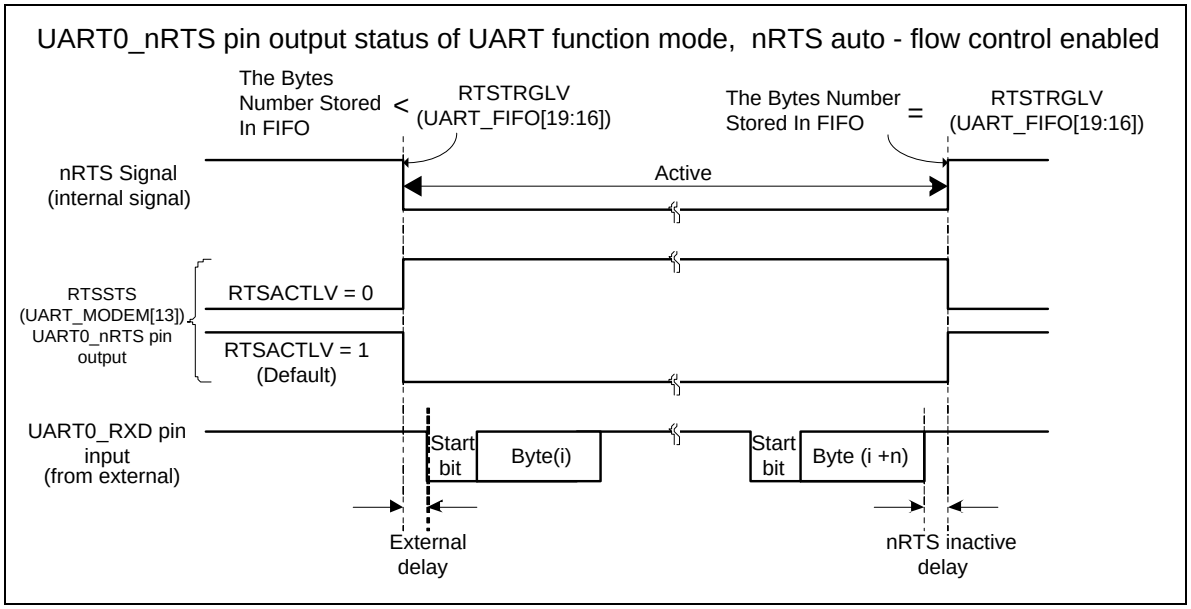


Figure 6.12-13 UART nRTS Auto-Flow Control Enabled

As shown in Figure 6.12-14, 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 UART0_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.

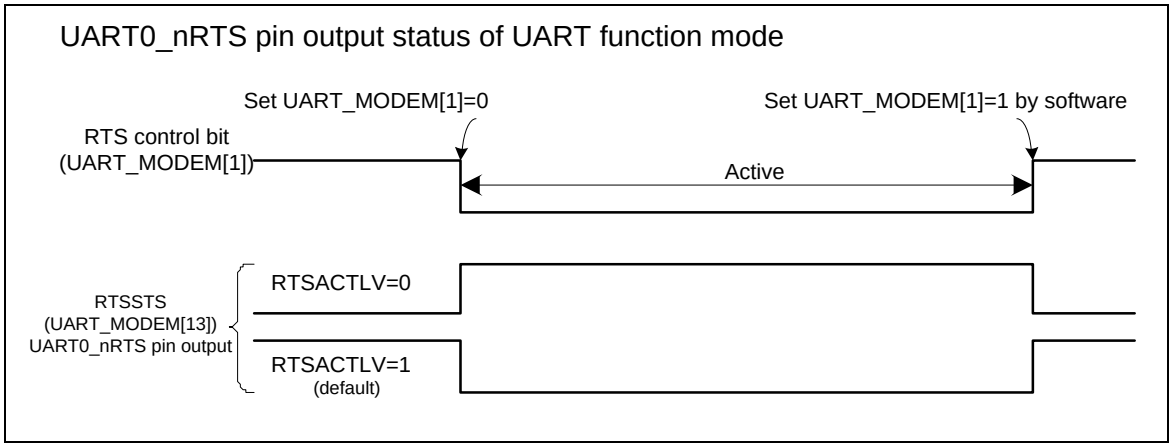


Figure 6.12-14 UART nRTS Auto-Flow with Software Control

6.12.5.9 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.

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 a next 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.

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.

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 6.12-15 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.

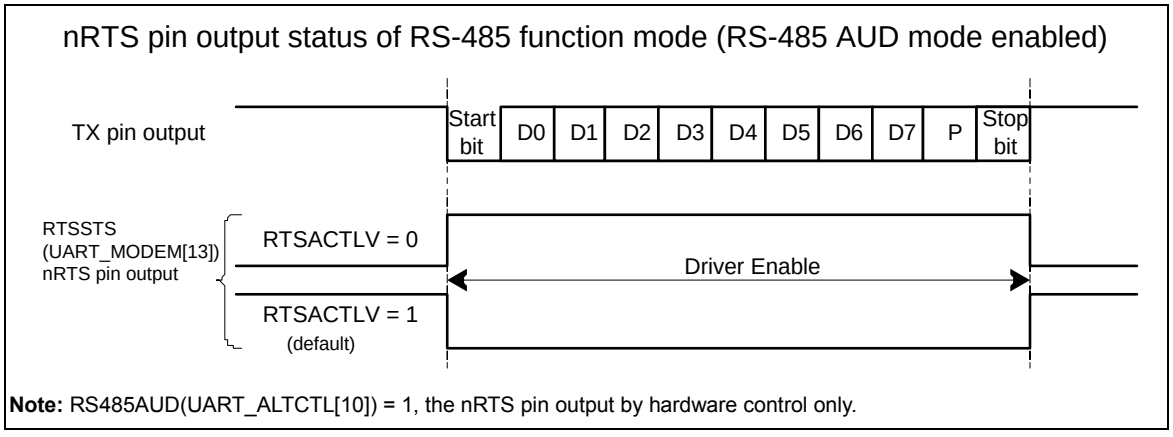


Figure 6.12-15 RS-485 nRTS Driving Level in Auto Direction Mode

Figure 6.12-16 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.

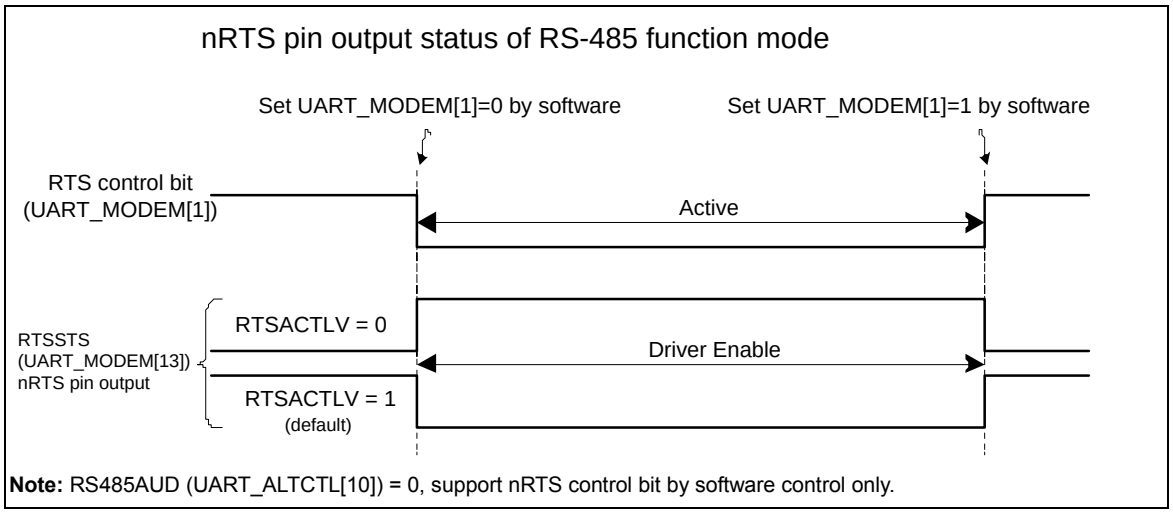


Figure 6.12-16 RS-485 nRTS Driving Level with Software Control

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]).

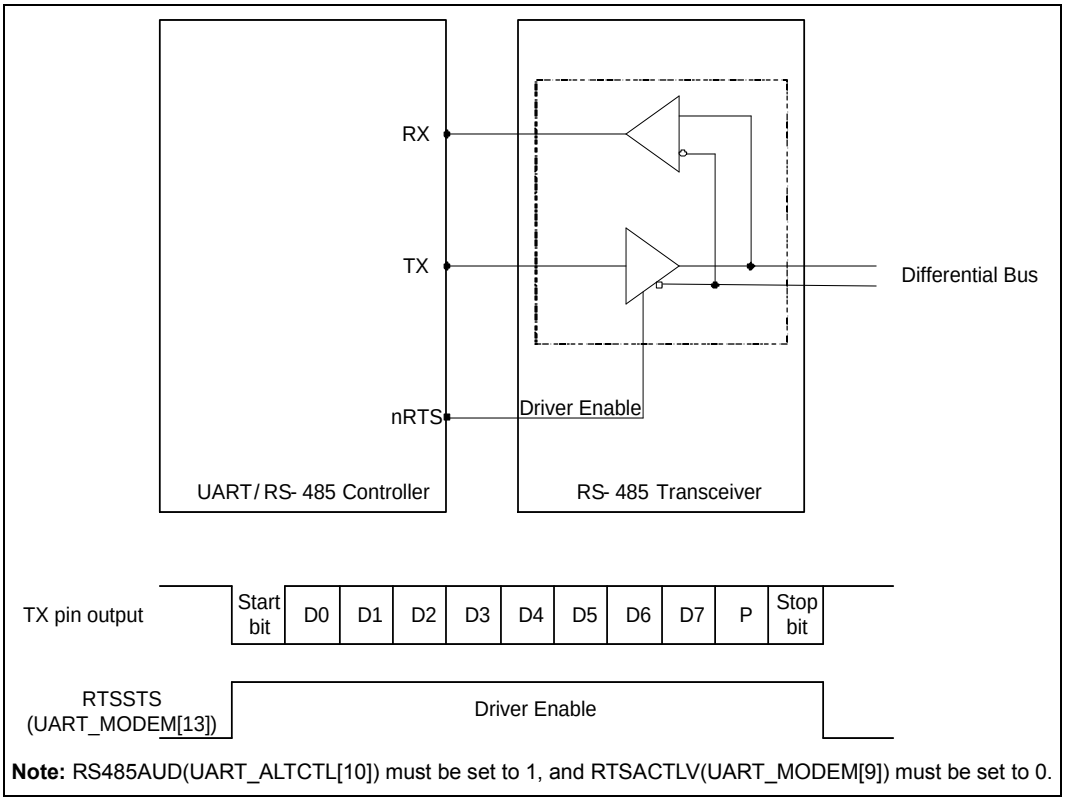


Figure 6.12-17 Structure of RS-485 Frame

6.12.5.10 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.

6.12.6 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 = 0x4007_0000**				
UART_DAT	UART0_BA+0x00	R/W	UART Receive/Transmit Buffer Register	0x0000_0000
UART_INTEN	UART0_BA+0x04	R/W	UART Interrupt Enable Register	0x0000_0000
UART_FIFO	UART0_BA+0x08	R/W	UART FIFO Control Register	0x0000_0101
UART_LINE	UART0_BA+0x0C	R/W	UART Line Control Register	0x0000_0000
UART_MODEM	UART0_BA+0x10	R/W	UART Modem Control Register	0x0000_0200
UART_MODEMSTS	UART0_BA+0x14	R/W	UART Modem Status Register	0x0000_0110
UART_FIFOSTS	UART0_BA+0x18	R/W	UART FIFO Status Register	0xB040_4000
UART_INTSTS	UART0_BA+0x1C	R/W	UART Interrupt Status Register	0x0040_0002
UART_TOUT	UART0_BA+0x20	R/W	UART Time-out Register	0x0000_0000
UART_BAUD	UART0_BA+0x24	R/W	UART Baud Rate Divider Register	0x0F00_0000
UART_ALTCTL	UART0_BA+0x2C	R/W	UART Alternate Control/Status Register	0x0000_000C
UART_FUNCSEL	UART0_BA+0x30	R/W	UART Function Select Register	0x0000_0000
UART_BRCOMP	UART0_BA+0x3C	R/W	UART Baud Rate Compensation Register	0x0000_0000
UART_WKCTL	UART0_BA+0x40	R/W	UART Wake-up Control Register	0x0000_0000
UART_WKSTS	UART0_BA+0x44	R/W	UART Wake-up Status Register	0x0000_0000
UART_DWKCOMP	UART0_BA+0x48	R/W	UART Incoming Data Wake-up Compensation Register	0x0000_0000

Note:

1. Any register not listed here is reserved and must not be written. The result of a read operation on these bits is undefined.
2. The reserved register fields that listed in register description must be written to their reset value. Writing reserved fields with other than reset values may produce undefined results.

6.12.7 Register Description

UART Receive/Transmit Buffer Register (UART_DAT)

Register	Offset	R/W	Description	Reset Value
UART_DAT	UART0_BA+0x00	R/W	UART Receive/Transmit Buffer Register	0x0000_0000

31	30	29	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. Any values read should be ignored. When writing to this field always write with reset value.
[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 UART0_TXD pin. 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 UART0_TXD pin. 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	UART0_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		
7	6	5	4	3	2	1	0
Reserved	WKIEN	BUFERRIEN	RXTOIEN	MODEMIEN	RLSIEN	THREIEN	RDAIEN

Bits	Description	
[31:23]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.
[18]	**ABRIEN**	**Auto-baud Rate Interrupt Enable Bit** 0 = Auto-baud rate interrupt Disabled. 1 = Auto-baud rate interrupt Enabled.
[17:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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 transmit request operation is stop. Clear Break Error Flag BIF or Frame Error Flag FEF or Parity Error Flag PEF by writing "1" to corresponding BIF, FEF and PEF to make UART PDMA transmit request operation continue.
[13]	**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.
[10:7]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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	UART0_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. Any values read should be ignored. When writing to this field always write with reset value.
[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. Do not use. **Note:** This field is used for auto nRTS flow control.
[15:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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. Do not use.
[3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[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. Any values read should be ignored. When writing to this field always write with reset value.

UART Line Control Register (UART_LINE)

Register	Offset	R/W	Description	Reset Value
UART_LINE	UART0_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. Any values read should be ignored. When writing to this field always write with reset value.
[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 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	UART0_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. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.
[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 6.12-13 and Figure 6.12-14 for UART function mode. **Note2:** Refer to Figure 6.12-15 and Figure 6.12-16 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. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.

UART Modem Status Register (UART_MODEMSTS)

Register	Offset	R/W	Description	Reset Value
UART_MODEMSTS	UART0_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. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.
[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	UART0_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	Reserved		TXPTR			
15	14	13	12	11	10	9	8
RXFULL	RXEMPTY	Reserved		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 cannot transmit or receive data at this moment. Otherwise this bit is set.
[30]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.
[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).
[21:20]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[19:16]	**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.
[13:12]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[11:8]	**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. Any values read should be ignored. When writing to this field always write with reset value.
[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	UART0_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
Reserved	WKINT	BUFERRINT	RXTOINT	MODEMINT	RLSINT	THREINT	RDAINT
7	6	5	4	3	2	1	0
Reserved	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 in PDMA 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. Any values read should be ignored. When writing to this field always write with reset value.
[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.
[17:15]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with

		reset value.
[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]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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 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. 1 = RX time-out interrupt flag is generated. **Note:** This bit is read only and user can read UART_DAT (RX is in active) to clear it.
[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** 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** 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	UART0_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. Any values read should be ignored. When writing to this field always write with reset value.
[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. So, for example, if TOIC is set with 40, the time-out interrupt is generated after four characters are not received when 1 stop bit and no parity check is set for UART transfer.

UART Baud Rate Divider Register (UART_BAUD)

Register	Offset	R/W	Description	Reset Value
UART_BAUD	UART0_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. Any values read should be ignored. When writing to this field always write with reset value.
[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 6.12.5-1.
[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 6.12.5-1.
[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 6.12.5-1.
[23:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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 6.12.5-1.

UART Alternate Control/Status Register (UART_ALTCTL)

Register	Offset	R/W	Description	Reset Value
UART_ALTCTL	UART0_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
Reserved							

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. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.
[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.
[7:0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

UART Function Select Register (UART_FUNCSEL)

Register	Offset	R/W	Description	Reset Value
UART_FUNCSEL	UART0_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. Any values read should be ignored. When writing to this field always write with reset value.
[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 completed current task before disable TX and RX. When TX and RX disable, the TXRXACT (UART_FIFOSTS[31]) is cleared.
[2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1:0]	**FUNCSEL**	**Function Select** 00 = UART function. 11 = RS-485 function. Others = Reserved. Do not use.

UART Baud Rate Compensation Register (UART_BRCOMP)

Register	Offset	R/W	Description	Reset Value
UART_BRCOMP	UART0_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. Any values read should be ignored. When writing to this field always write with reset value.
[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	UART0_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. Any values read should be ignored. When writing to this field always write with reset value.
[4]	**WKTOUTEN**	**RX FIFO Time-out Wake-up Enable Bit** 0 = RX FIFO time-out wake-up system function Disabled. 1 = RX FIFO time-out wake-up system function Enabled: a time-out event for RX FIFO not reaching threshold wakes up the 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**	**RX FIFO Reached Threshold Wake-up Enable Bit** 0 = RX FIFO reached threshold wake-up system function Disabled. 1 = RX FIFO reached threshold wake-up system function Enabled: RX FIFO reaching threshold wakes up the 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	UART0_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. Any values read should be ignored. When writing to this field always write with reset value.
[4]	**TOUTWKF**	**RX FIFO Threshold Time-out Wake-up Flag** This bit is set indicating system wake-up from a RX FIFO Threshold Time-out event 0 = Chip stays in power-down state. 1 = Chip wake-up from RX FIFO Threshold Time-out event. wake-up. **Note1:** If WKTOUTEN (UART_WKCTL[4]) is enabled, the RX FIFO threshold time-out wake-up will set TOUTWKF bit 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**	This bit is set if chip wake-up from power-down state by RX FIFO reached threshold wake-up . 0 = Chip stays in power-down state. 1 = Chip wake-up from power-down state by RX FIFO Reached Threshold wake-up. **Note1:** If WKRFRTEN (UART_WKCTL[2]) is enabled, the RX FIFO Reached Threshold wake-up cause this bit is set to '1'.
[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 Incoming Data Wake-up Compensation Register (UART_DWKCOMP)

Register	Offset	R/W	Description	Reset Value
UART_DWKCOMP	UART0_BA+0x48	R/W	UART Incoming 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. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**STCOMP**	**Start Bit Compensation Value** These bits field indicate how many clock cycle selected by UART0_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.

6.13 I²C Serial Interface Controller (I²C)

6.13.1 Overview

I²C is a two-wire, bi-directional serial bus that provides a simple and efficient method of data exchange between devices. The NPCA121 series device provides two sets of I²C controller which can function as either master or slave, provide multi-master capability, support up to 1Mbs transfer rate.

The NPCA121 I2C module can also be used for a variety of purposes, including CRC verification, SMBus (System Management Bus) and PMBus (Power Management Bus).

6.13.2 Features

The NPCA121 series I2C module supports the following features:

- Two I²C ports
- Master, Salve and Multi-master mode operation
- Support High speed mode 3.4Mbps
- Supports Standard mode (100 kbps), Fast mode (400 kbps) and Fast mode plus (1 Mbps)
- 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 I²C interrupt if the I²C bus hangs up and timer-out counter overflows
- Programmable clocks allow for versatile rate control
- 7-bit and 10-bit addressing mode
- Multiple address recognition (four slave address with mask option)
- Power-down wake-up function
- Programmable setup/hold time
- Bus Management (SM/PM compatible) function

6.13.3 Block Diagram

The block diagram of I²C controller is shown below.

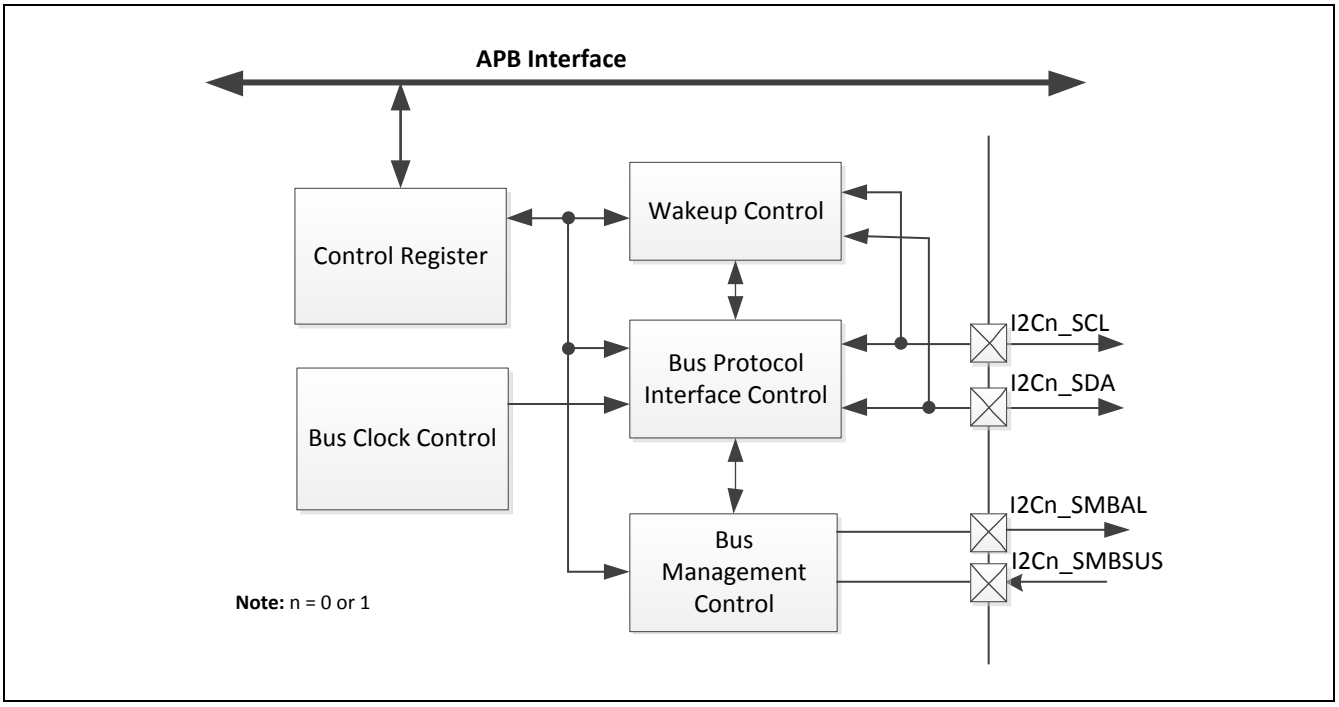


Figure 6.13-1 I^{2}C Controller Block Diagram

6.13.4 Basic Configuration

6.13.4.1 I2C0 Basic Configurations

- Clock source configuration
  - Enable I2C0 peripheral clock in I2C0CKEN (CLK_APBCLK0[8]).
- Reset configuration
  - Reset I2C0 controller in I2C0RST (SYS_IPRST1[8]).
- Pin configuration

Group	Pin Name	GPIO	MFP
I2C0	I2C0_SCL	PA.9	MFP1
		PB.0	MFP2
		PB.6	MFP3
		PC.13	MFP2
		PD.0	MFP3
		PD.8	MFP4
		PD.14	MFP3
	I2C0_SDA	PA.10	MFP1
		PB.1	MFP2
		PB.5	MFP3

		PC.14	MFP2
		PD.1	MFP3
		PD.9	MFP4
		PD.15	MFP3
	I2C0_SMBAL	PA.12	MFP1
	I2C0_SMBSUS	PA.11	MFP1

6.13.4.2 I2C1 Basic Configurations

- Clock source configuration
  - Enable I2C1 peripheral clock in I2C1CKEN (CLK_APBCLK0[9]).
- Reset configuration
  - Reset I2C1 controller in I2C1RST (SYS_IPRST1[9]).
- Pin configuration

Group	Pin Name	GPIO	MFP
I2C1	I2C1_SCL	PA.13	MFP4
		PB.6	MFP4
		PC.0	MFP1
		PD.0	MFP2
		PD.2	MFP4
		PD.8	MFP3
		PD.14	MFP5
	I2C1_SDA	PA.14	MFP4
		PB.5	MFP4
		PC.1	MFP1
		PD.1	MFP2
		PD.9	MFP3
		PD.15	MFP5
	I2C1_SMBAL	PC.3	MFP1
	I2C1_SMBSUS	PC.2	MFP1

6.13.5 Functional Description

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

(START or STOP). Please refer to the Figure 6.13-2 for more detailed I²C BUS Timing.

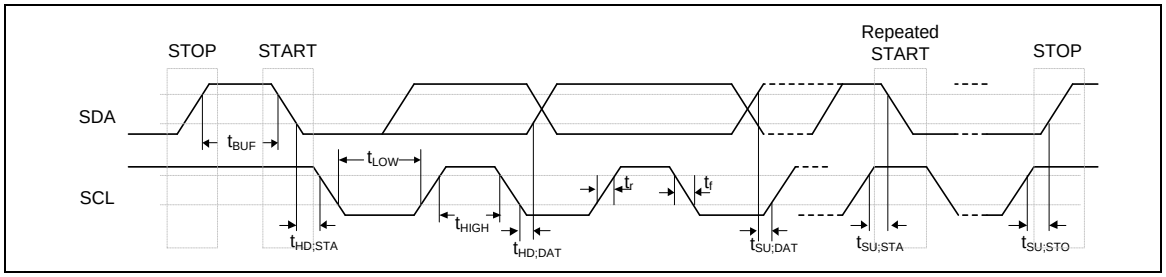


Figure 6.13-2 I2C Bus Timing

The device's on-chip I²C provides the serial interface that meets the I²C bus standard mode specification. The I²C port handles byte transfers autonomously. To enable this port, the bit I2CEN in I2C_CTL should be set to '1'. The I²C hardware interfaces to the I²C bus via two pins: SDA and SCL. When I/O pins are used as I²C ports, user must set the pins function to I²C in advance.

Note: Pull-up resistor is needed for I²C operation as the SDA and SCL are open-drain pins.

6.13.5.1 I²C Protocol

Figure 6.13-3 shows the typical I²C signal waveform. Normally, a standard I2C communication session consists of four stages:

- START or Repeated START condition
- Slave address and R/W bit transfer
- Data transfer
- STOP condition

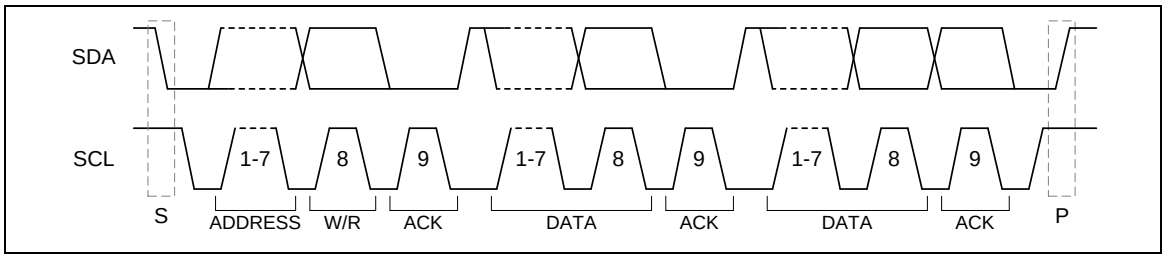


Figure 6.13-3 I²C Protocol

6.13.5.1.1 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 new I2C session by generating a START condition. A START condition, 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 condition denotes the beginning of a new data transmission.

After a successful data transmitting /receiving session, a master can generate a STOP condition to release the I2C bus, or generate a Repeated START (Sr) condition for continuing data transfer. The START (S) and Repeated START (Sr) conditions are functionally identical. By generating

Repeated START (Sr) condition, a master can transmit and/or receive data from the same slave or different slaves without releasing the bus.

6.13.5.1.2 STOP signal

I2C data flow follows the direction indicated by the R/W bit in addressing byte. The receiver (master or slave) should acknowledge each received byte on the 9th SCL clock cycle.

- If the slave signals a Not Acknowledge (NACK), the master can generate a STOP condition to abort the data transfer or generate a Repeated START condition and start a new transmission.
- 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 condition.

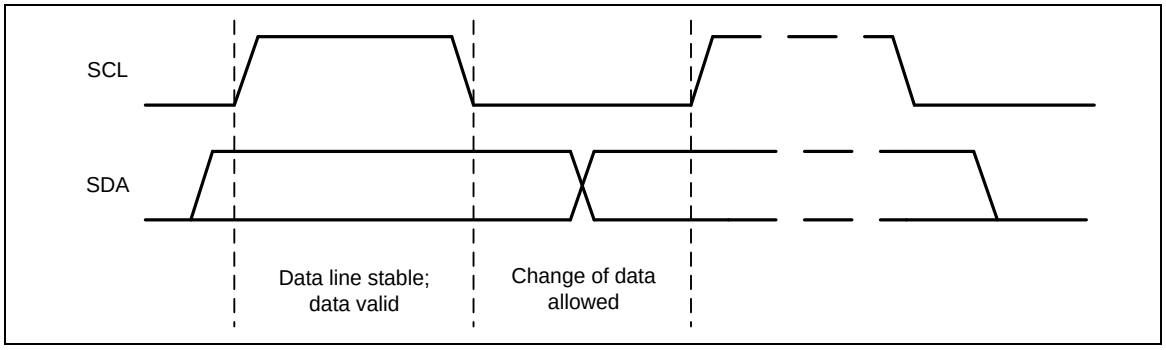


Figure 6.13-4 Bit Transfer on the I^2C Bus

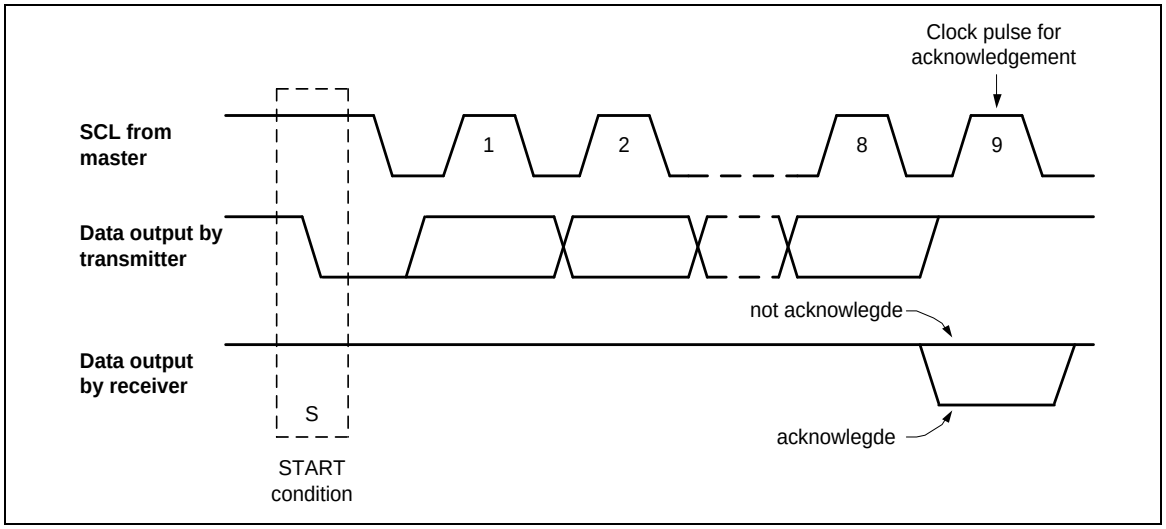


Figure 6.13-5 Acknowledge on the I^2C Bus

6.13.5.1.3 Slave Address Transmission

After a START (or Repeated START) condition, the master sends slave address to the SDA line.

- In 7-bit address mode, one address byte is to be sent.
  - To enter Transmitter mode, a master sends the slave address with LSB reset (i.e. as 0).
  - To enter Receiver mode, a master sends the slave address with LSB set (i.e. as 1).
- In 10-bit address mode,
  - To enter Transmitter mode, a master sends the 1st header byte (11110xx0) followed by the 2nd address byte which contains the slave's lower 8 bit address, (where xx denotes the two most significant bits of the address).
  - To enter Receiver mode, a master sends the 1st header byte (11110xx0) followed by the 2nd address byte. Then the master should send a Repeated START condition followed by header (11110xx1), (where xx denotes the two most significant bits of the address).

In either mode, the slave with matching address should send acknowledge bit after the addressing byte(s), so the communication can continue.

6.13.5.1.4 Data Transfer

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

If the master, as 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. The Figure 6.13-6 and Figure 6.13-7 shows the waveform of bit transfer and acknowledge.

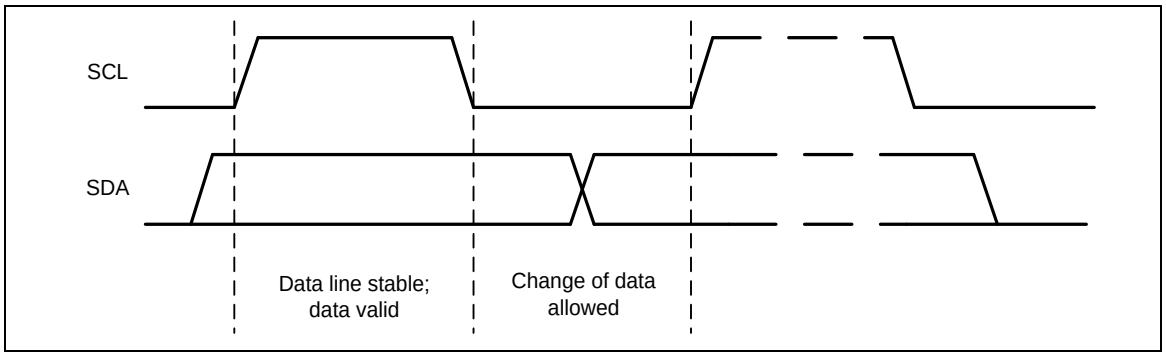


Figure 6.13-6 Bit Transfer on the I2C Bus

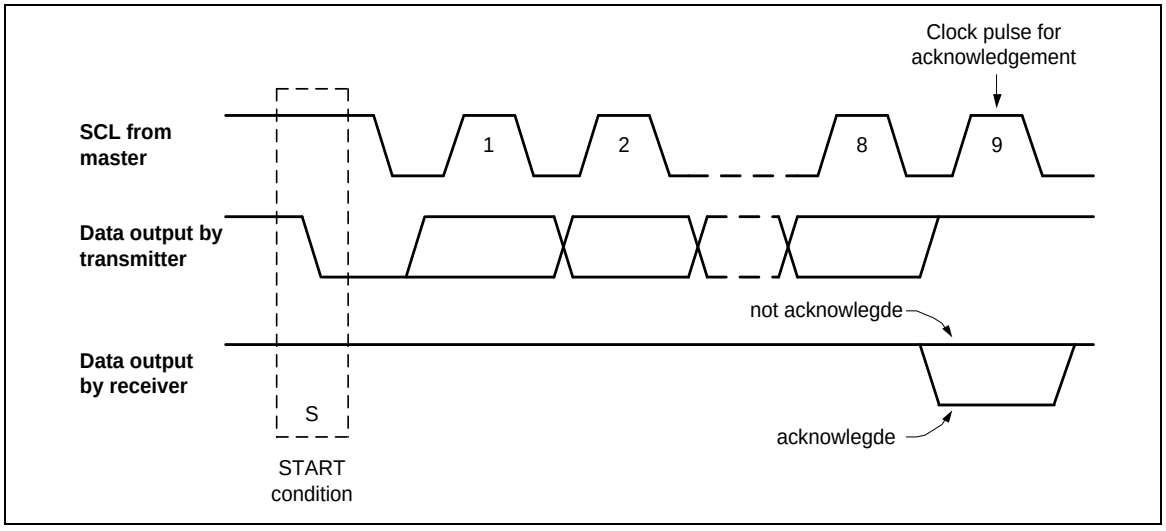


Figure 6.13-7 Acknowledge on the I^2C Bus

6.13.5.1.5 Complete I2C Communication Flow

The following figures illustrate how an I2C master initiates and completes a read/write operation with a 7-bit or 10-bit slave.

- Master transmits data to a slave with 7-bit address

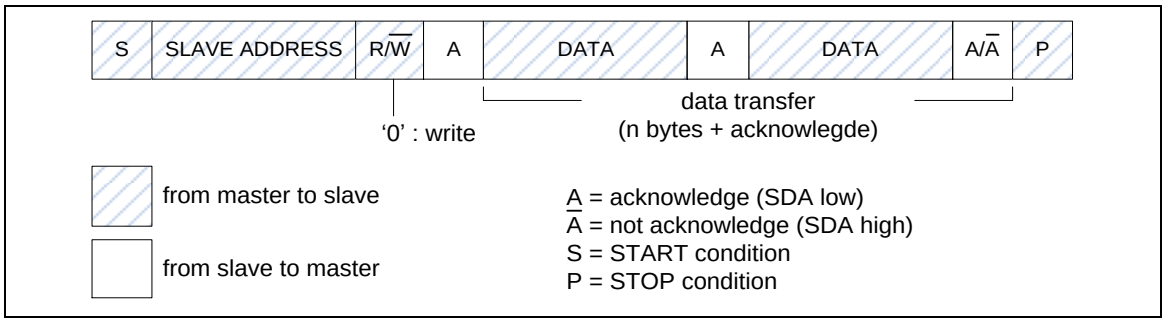


Figure 6.13-8 Master Transmits Data to Slave by 7-bit

- Master reads data from a slave with 7-bit address

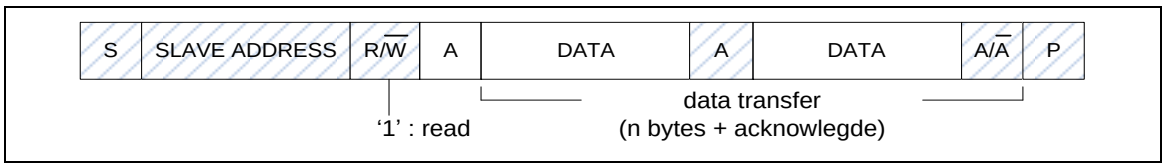


Figure 6.13-9 Master Reads Data from Slave by 7-bit

- Master transmits data to a slave with 10-bit address

The header byte contains 10-bit address indicator (5'b11110), two most significant address bits and one Read/Write bit – in this case it is '0' indicating a write operation. The second address

byte contains the lower 8-bit address. The master keeps sending data after addressing byte acknowledged. The 7-bit and 10-bit address devices can work on the same bus.

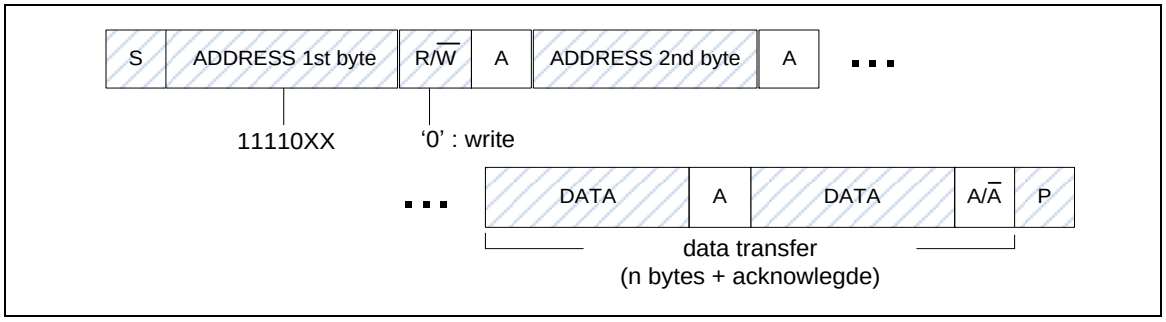


Figure 6.13-10 Master Transmits Data to Slave by 10-bit

- Master reads data from a slave with 10-bit address

Figure 6.13-11 shows a master reading data from a slave with 10-bit address. First the master needs to send 10-bit addressing bits with R/W bit as 0 in the first header byte; then the master needs to send the second header byte only with R/W bit as 1 to indicate this is a master read operation.

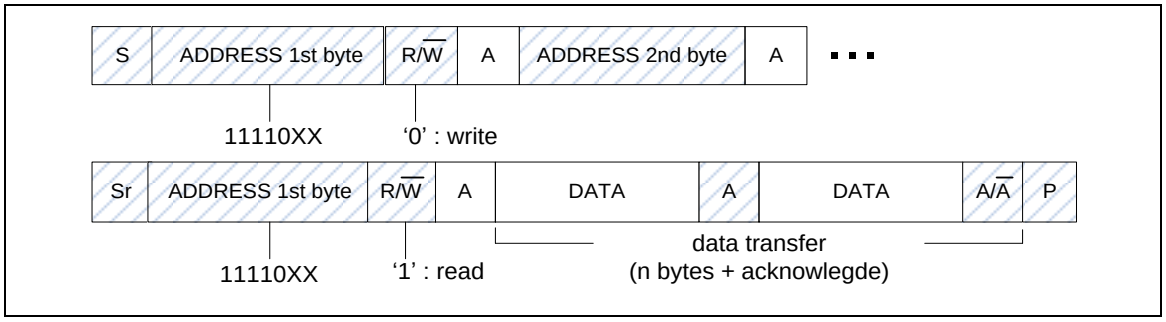


Figure 6.13-11 Master Reads Data from Slave by 10-bit

6.13.5.2 Operational Description

6.13.5.2.1 I2C Initialization

An interrupt driven process is suitable for handling the NPCA121 series I2C communication. When working as a master, the NPCA121 series device generates a START condition, then responds to the following interrupts; in slave mode, after I2C initialization the device simply waits for I2C interrupts.

Typically the NPCA121 series I2C initialization does the following:

- Enable I2C module clock
- Configure I2C clock speed, enable I2C module
- Enable I2C interrupt, and initialize I2C NVIC vector
- Configure its own the slave address register.

The code snippet below shows the I2C module initialization procedure:

```
FMC->ISPCMD = FMC_ISPCMD_READ;          // op code for 32-bit read is 0x00
CLK_EnableModuleClock(I2C0_MODULE);
I2C_Open(I2C0, 100000);                 // set clock to 100K, and enable I2C_CTL_I2CEN_Msk in I2C_CTL

I2C_SetSlaveAddr(I2C0, 0, 0x15, I2C_GCMODE_DISABLE);   /* Slave Address : 0x15 */
…
I2C_EnableInt(I2C0);
NVIC_EnableIRQ(I2C0_IRQn);
```

Writing 1 to INTEN bit in I2C_CTL register enables the I2C interrupt. I2C interrupt is generated at the 9th clock, i.e. after master/slave received ACK/NACK. The interrupt service routine (ISR) should check the I2C_STATUS value, and handle the I2C transaction via I2C_CTL control bits and I2C_DAT data register.

Four control bits in I2C_CTL register are frequently used:

- STA: writing 1 to STA bit generates a START (or repeated START) condition. This bit will be automatically cleared by hardware.
- STO: writing 1 to STO bit generates a STOP condition. This bit will be automatically cleared by hardware.
- SI: Interrupt flag is set by hardware, software must write 1 to SI bit to clear the I2C interrupt flag.
- AA: 1- return acknowledge after a byte is received (matched address or data). 0 – return Not Acknowledge (NACK).

For more details of I2C registers, refer to section “I2C Protocol Registers” and “Register Description”.

Figure 6.13-12 shows an example of master side flow control: software reads I2C_Status, if it reads value 0x08 (START condition sent), then set I2C_DATA=SLA+W and (STA,STO,SI,AA) = (0,0,1,x) to send the address to I2C bus. An ACK is expected from the slave with the matching address. After ACK is received, the I2C_STATUS will be updated by status code 0x18.

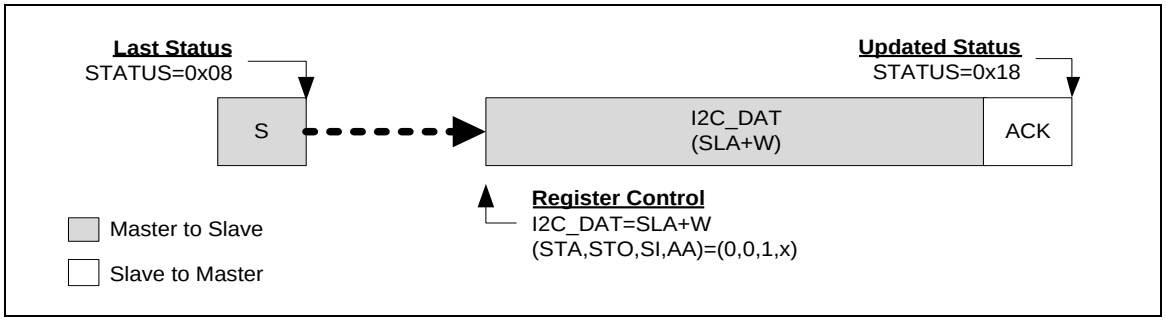


Figure 6.13-12 Control I2C Bus according to the current I2C Status

Sample code for Figure 6.13-12

```
if (u32Status == 0x08) {                    /* START has been transmitted and prepare SLA+W */
   I2C_SET_DATA(I2C0, (g_u8DeviceAddr << 1));  /* Write SLA+W to Register I2CDAT */
   I2C_SET_CONTROL_REG(I2C0, I2C_SI);       /* STA =0, STO=0; SI = 1 to clear the interrupt flag */

} else if (u32Status == 0x18) {             /* SLA+W has been transmitted and ACK has been received */
   …
}
```

6.13.5.2.2 Master Mode

Figure 6.13-13 and Figure 6.13-14 illustrates the work flow as a Master Transmitter and a Master Receiver, respectively.

Writing 1 to STA bit (I2C_CTL[5]) enters Master mode. This operation generates a START condition and therefore a new I2C session is started. An interrupt will be generated if I2C interrupt is enabled. In ISR, software always checks the I2C_STATUS value first, then proceeds by manipulating I2C_CTL control bits and I2C_DAT data registers, etc.

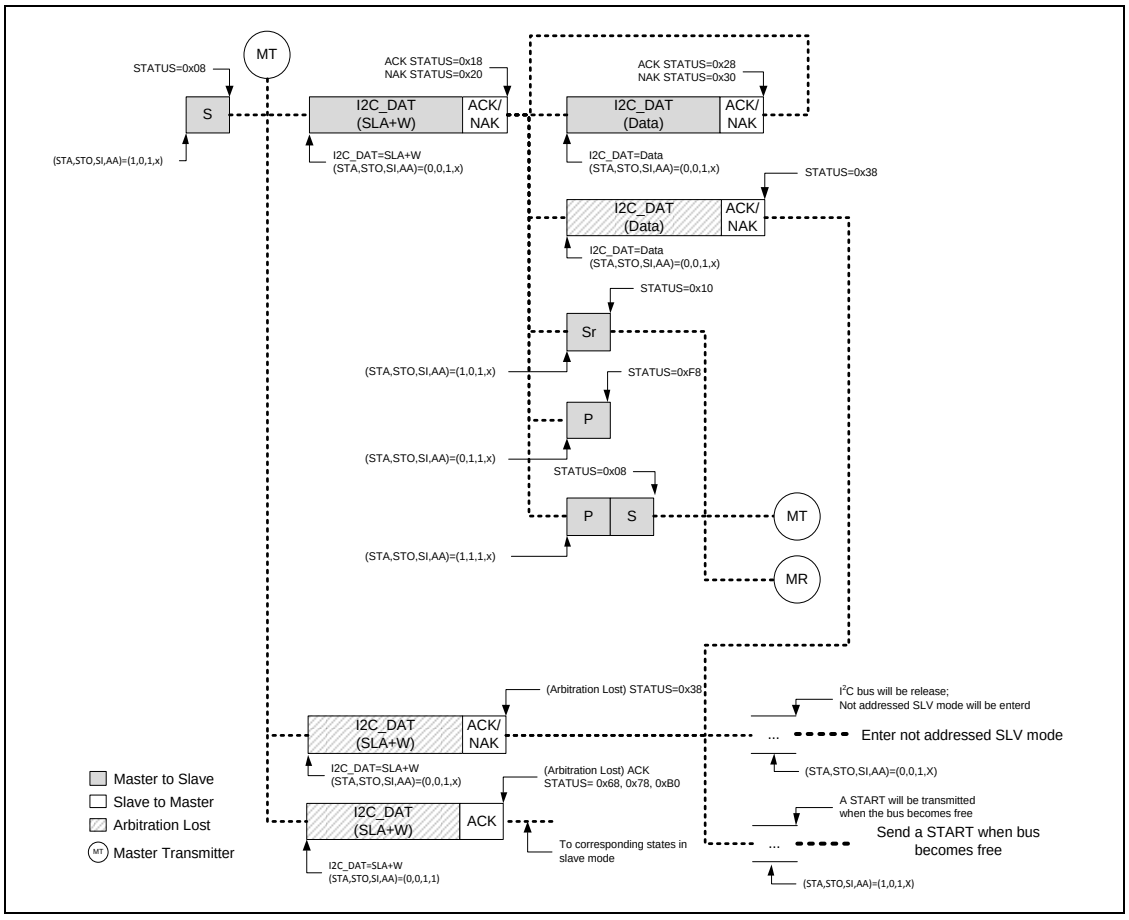


Figure 6.13-13 Master Transmitter Mode Control Flow

Sample code for Figure 6.13-13, master mode transmitter:

```
/*  ------  in main routine ------ */
…
/* I2C interface initialization */
…
s_I2C0HandlerFn = (I2C_FUNC)I2C_MasterTx;          /* Setup I2C ISR callback function */
I2C_SET_CONTROL_REG(I2C0, I2C_STA);                /* As an I2C master,  generates START condition */
…

/* ------- I2C interrupt service routine: I2C_MasterTx  ------ */
u32Status = I2C_GET_STATUS(I2C0);
if (u32Status == 0x08) {                           /* START has been transmitted */
```

```
        I2C_SET_DATA(I2C0, g_u8DeviceAddr << 1);            /* Write SLA+W to Register I2CDAT */
        I2C_SET_CONTROL_REG(I2C0, I2C_SI);                  /* Write 1 to SI to clear the I2C interrupt flag */
    }
    else if (u32Status == 0x18) {                           /* SLA+W has been transmitted and ACK has been received */
        I2C_SET_DATA(I2C0, g_au8TxData[g_u8DataLen++]);     /* Write data into I2C_DAT register, */
                                                            /* and the I2C module will start the transmission */
        I2C_SET_CONTROL_REG(I2C0, I2C_SI);
    }
    else if (u32Status == 0x20) {                /* SLA+W has been transmitted and NACK has been received */
        I2C_SET_CONTROL_REG(I2C0, I2C_STA | I2C_STO | I2C_SI);
    }
    else if (u32Status == 0x28) {           /* DATA has been transmitted and ACK has been received */
        if (g_u8DataLen != 3) {
            I2C_SET_DATA(I2C0, g_au8TxData[g_u8DataLen++]);          /* continue writing the rest of the data */
            I2C_SET_CONTROL_REG(I2C0, I2C_SI);
        }
        else {
            I2C_SET_CONTROL_REG(I2C0, I2C_STO | I2C_SI);      /* data writing completed, issue STOP condition */
            …
        }
    }
    else {
    /* To be added: handling for other status code, such Arbitration lost 0x38, etc.    */
    }
```

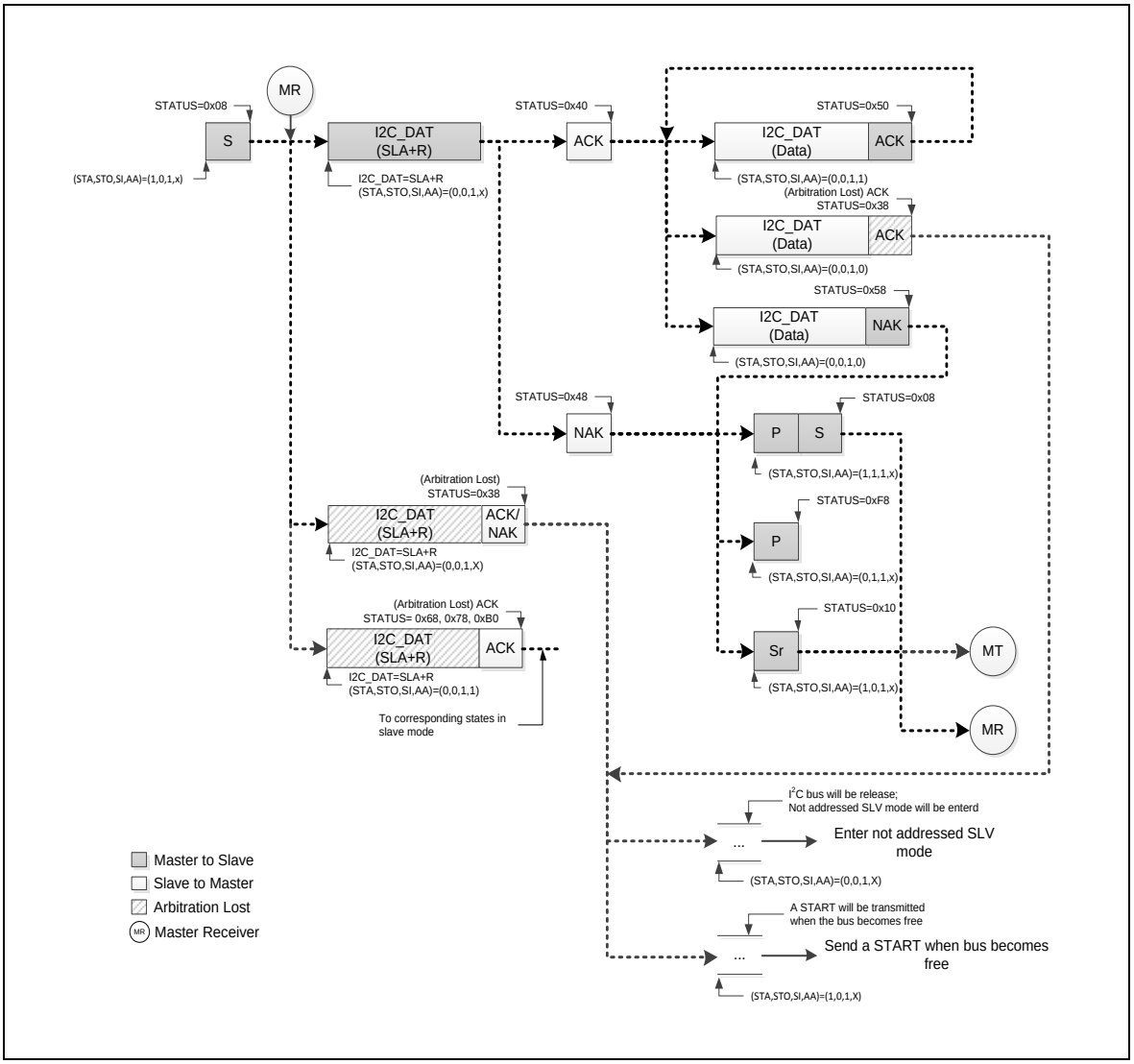


Figure 6.13-14 Master Receiver Mode Control Flow

Sample code for Figure 6.13-14, master mode receiver:

```
/*  ------  in main routine ------ */
…
/* I2C interface initialization */
…
s_I2C0HandlerFn = (I2C_FUNC)I2C_MasterTx;          /* Setup I2C ISR callback function: master write */
I2C_SET_CONTROL_REG(I2C0, I2C_STA);                /* As an I2C master,  generates START condition */

    /* Wait I2C Tx Finish */
    while (g_u8EndFlag == 0);
    g_u8EndFlag = 0;

s_I2C0HandlerFn = (I2C_FUNC)I2C_MasterRx;          /* I2C function to read data from slave: master read */
g_u8DataLen = 0;
g_u8DeviceAddr = slvaddr;
```

```
I2C_SET_CONTROL_REG(I2C0, I2C_STA);                    /* again, generates START condition */

…
/* ------- I2C interrupt service routine: I2C_MasterTx  ------ */
/* see sample code for Figure 6.13-13 */

/* ------- I2C interrupt service routine: I2C_MasterRx  ------ */
if (u32Status == 0x08) {                                /* START has been transmitted and prepare SLA+W */
    I2C_SET_DATA(I2C0, (g_u8DeviceAddr << 1));          /* Write SLA+W to Register I2CDAT */
    I2C_SET_CONTROL_REG(I2C0, I2C_SI);                  /* write 1 to SI to clear the I2C interrupt flag */
}
else if (u32Status == 0x18) {                           /* SLA+W has been transmitted and ACK has been received */
    I2C_SET_DATA(I2C0, g_au8TxData[g_u8DataLen++]);
    I2C_SET_CONTROL_REG(I2C0, I2C_SI);
}
else if (u32Status == 0x20) {                           /* SLA+W has been transmitted and NACK has been received */
    I2C_SET_CONTROL_REG(I2C0, I2C_STA | I2C_STO | I2C_SI);
}
else if (u32Status == 0x28) {                           /* DATA has been transmitted and ACK has been received */
    if (g_u8DataLen != 2) {
        I2C_SET_DATA(I2C0, g_au8TxData[g_u8DataLen++]);
        I2C_SET_CONTROL_REG(I2C0, I2C_SI);
    } else {
        I2C_SET_CONTROL_REG(I2C0, I2C_STA | I2C_SI);
    }
}
else if (u32Status == 0x10) {          /* Repeat START has been transmitted and prepare SLA+R */
    I2C_SET_DATA(I2C0, (g_u8DeviceAddr << 1) | 0x01);  /* Write SLA+R to Register I2CDAT */
    I2C_SET_CONTROL_REG(I2C0, I2C_SI);
}
else if (u32Status == 0x40) {          /* SLA+R has been transmitted and ACK has been received */
    I2C_SET_CONTROL_REG(I2C0, I2C_SI);
}
else if (u32Status == 0x58) {          /* DATA has been received and NACK has been returned */
    g_u8RxData = I2C_GET_DATA(I2C0);
    I2C_SET_CONTROL_REG(I2C0, I2C_STO | I2C_SI);
    g_u8EndFlag = 1;
}
else {
    /* To be added: handling for other status such as Arbitration Lost code 0x38, etc. */
}
```

In master mode if the device gets arbitration lost after START condition (status code 0x38), software may set (STA, STO, SI, AA) = (1, 0, 1, X) to send repeated START so the master operation can resume when bus is free, or set (STA, STO, SI, AA) = (0, 0, 1, X) to release I^2C bus and enter not addressed Slave mode.

6.13.5.2.3 Slave Mode

After reset by default the NPCA121 device I²C is not addressed and will not acknowledge the address on I²C bus. Once the I2C address registers I2C_ADDRn (n=0~3) are written, and (STA, STO, SI, AA) = (0, 0, 1, 1), the NPCA121 I2C will acknowledge the matching address presented on I2C bus. Figure 6.13-15 shows all the possible control flow in I2C slave mode.

If bus arbitration is lost in Master mode, I²C port switches to Slave mode immediately and can detect its own slave address in the same serial transfer. If the detected address is SLA+W (Master want to write data to Slave) after arbitration lost, the status code is 0x68. If the detected address is SLA+R (Master want to read data from Slave) after arbitration lost, the status code is 0xB0.

Note: During I²C communication, the SCL clock will be released when writing '1' to clear SI flag in slave mode.

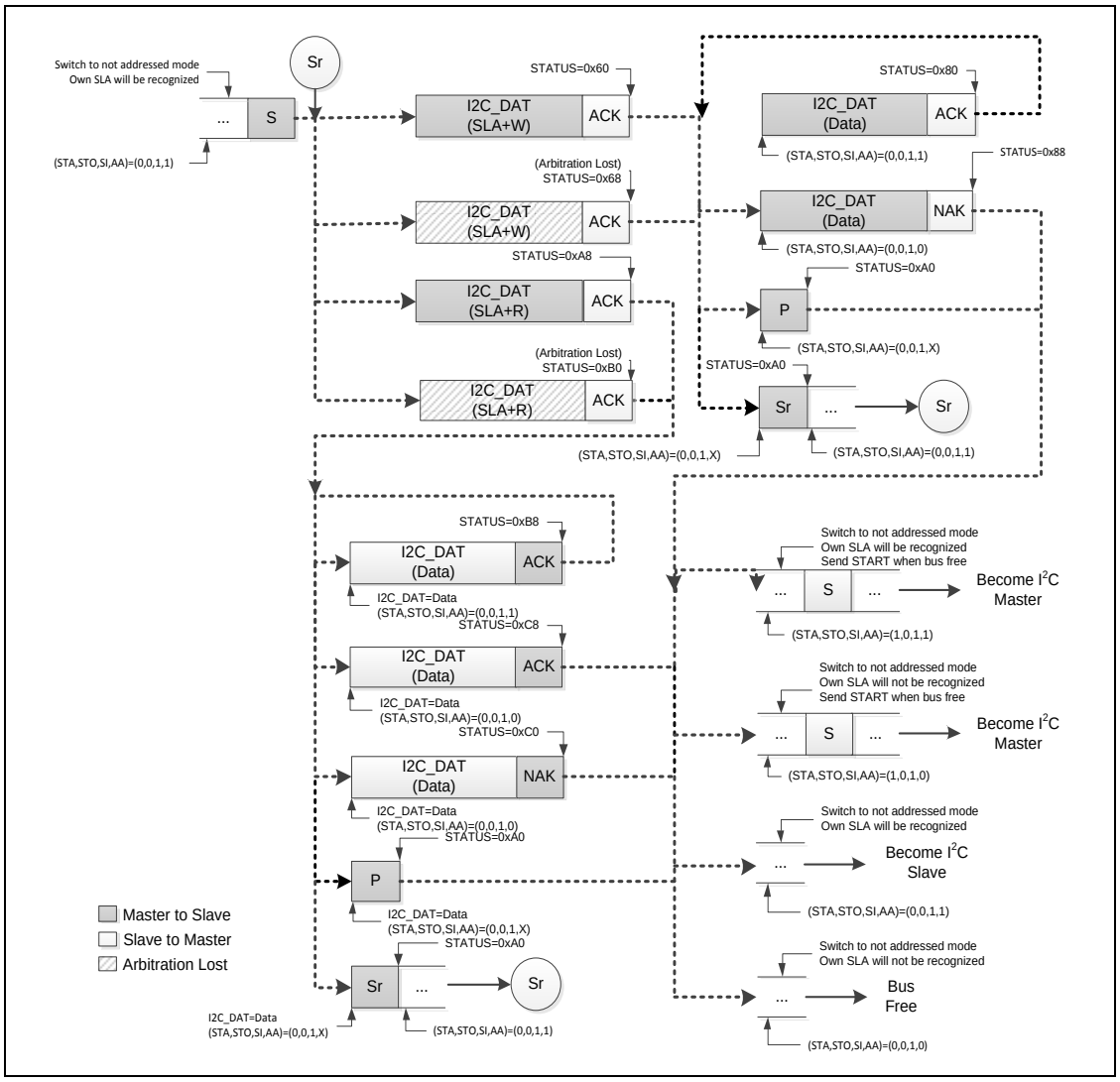


Figure 6.13-15 Slave Mode Control Flow

If I²C is still receiving data in addressed Slave mode but got a STOP or Repeat START, the status

code will be 0xA0. User could follow the action for status code 0x88 as shown in the above figure when getting 0xA0 status.

If I^2C is still transmitting data in addressed Slave mode but got a STOP or Repeat START, the status code will be 0xA0. User could follow the action for status code 0xC8 as shown in the above figure when getting 0xA0 status.

Note: After slave gets status of 0x88, 0xC8, 0xC0 and 0xA0, slave can switch to not address mode and own SLA will not be recognized. If entering this status, slave will not receive any I^2C signal or address from master. At this status, I^2C should enter into idle mode.

Sample code for Figure 6.13-15, I2C slave mode control flow:

```
/*   ------  in main routine, slave mode initialization ------ */
…
I2C_Open(I2C0, 100000);                 /* Open I2C0 and set clock to 100k */

 /* Set I2C0 4 Slave Addresses */
I2C_SetSlaveAddr(I2C0, 0, 0x15, I2C_GCMODE_DISABLE);   /* Slave Address : 0x15 */
I2C_SetSlaveAddr(I2C0, 1, 0x35, I2C_GCMODE_DISABLE);   /* Slave Address : 0x35 */
I2C_SetSlaveAddr(I2C0, 2, 0x55, I2C_GCMODE_DISABLE);   /* Slave Address : 0x55 */
I2C_SetSlaveAddr(I2C0, 3, 0x75, I2C_GCMODE_DISABLE);   /* Slave Address : 0x75 */

I2C_EnableInt(I2C0);
NVIC_EnableIRQ(I2C0_IRQn);

I2C_SET_CONTROL_REG(I2C0, I2C_SI | I2C_AA);          /* I2C enters addressed slave mode */
s_I2C0HandlerFn=I2C_SlaveTRx;                        /* entry point for slave mode ISR function */

... /* now waits for the I2C interrupt */
…

/* ------- I2C slave mode interrupt service routine: I2C_SlaveTRx ------ */
u32Status = I2C_GET_STATUS(I2C0);

if (u32Status == 0x60) {                /* detected matching address, and sent ACK to master */
      …
      I2C_SET_CONTROL_REG(I2C0, I2C_SI | I2C_AA);     /* clear interrupt and prepare for next ACK */
 }
else if (u32Status == 0x80) {           /* got data from master, and returned ACK to master */
      g_au8RxData[g_u8DataLen] = I2C_GET_DATA(I2C0);  /* read data from I2C_DAT register*/
      g_u8DataLen++;
      …
      I2C_SET_CONTROL_REG(I2C0, I2C_SI | I2C_AA);
}
else if(u32Status == 0xA8) {            /* sent address to master, and returned ACK to master */
      I2C_SET_DATA(I2C0, g_u8SlvData[slave_buff_addr]);  /* continue sending next address byte */
      slave_buff_addr++;
      I2C_SET_CONTROL_REG(I2C0, I2C_SI | I2C_AA);
 }
else if (u32Status == 0xC0)   {         /* Data byte or last data in I2CDAT has been transmitted,
                                           and received NACK */
      I2C_SET_CONTROL_REG(I2C0, I2C_SI | I2C_AA);
}
```

```
else if (u32Status == 0x88) {          /* Previously addressed with own SLA address; NOT ACK has
                                       been returned */
    g_u8DataLen = 0;
    I2C_SET_CONTROL_REG(I2C0, I2C_SI | I2C_AA);
} else if (u32Status == 0xA0) {        /* A STOP or repeated START has been received while still
                                       addressed as Slave/Receiver*/
    g_u8DataLen = 0;
    I2C_SET_CONTROL_REG(I2C0, I2C_SI | I2C_AA);
}
else {
    /* TO DO */
}
```

6.13.5.2.4 General Call (GC) Mode

If the GC bit (I2C_ADDRn [0]) is set, the I²C 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 I²C in Slave mode, it can receive the general call address by 0x00 after master send general call address to I²C bus, then it will follow status of GC mode.

The GC mode can wake up when address matched. Notice that the default address is 0x00, but user must set an address except for 0x00.

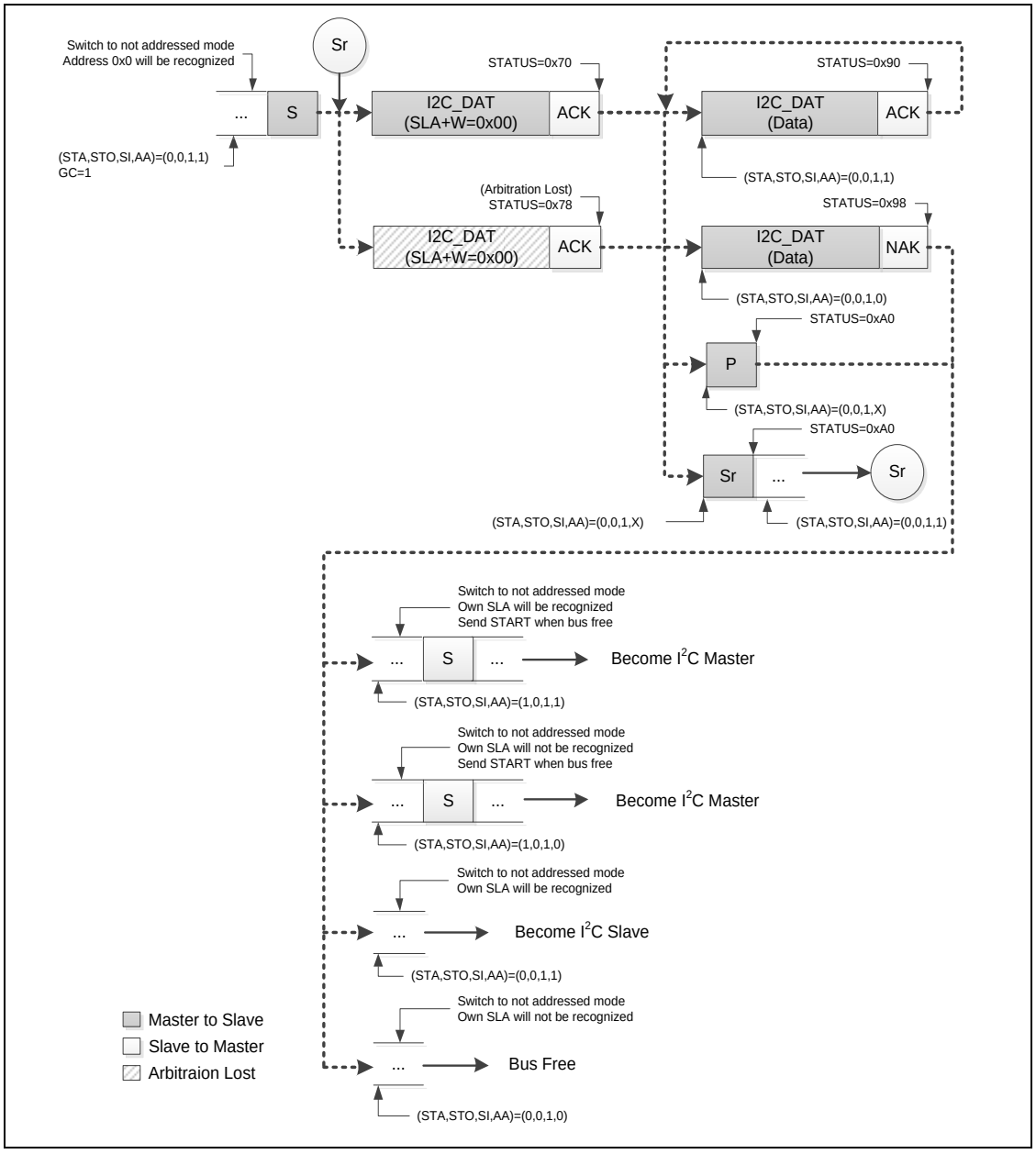


Figure 6.13-16 GC Mode

If I²C is still receiving data in GC mode but got a STOP or Repeat START, the status code will be 0xA0. User could follow the action for status code 0x98 in above figure when getting 0xA0 status.

Note: After slave gets status of 0x98 and 0xA0, slave can switch to not address mode and own

SLA will not be recognized. If entering this status, slave will not receive any I^{2}C signal or address from master. At this time, I^{2}C controller should enter into idle mode.

6.13.5.2.5 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.

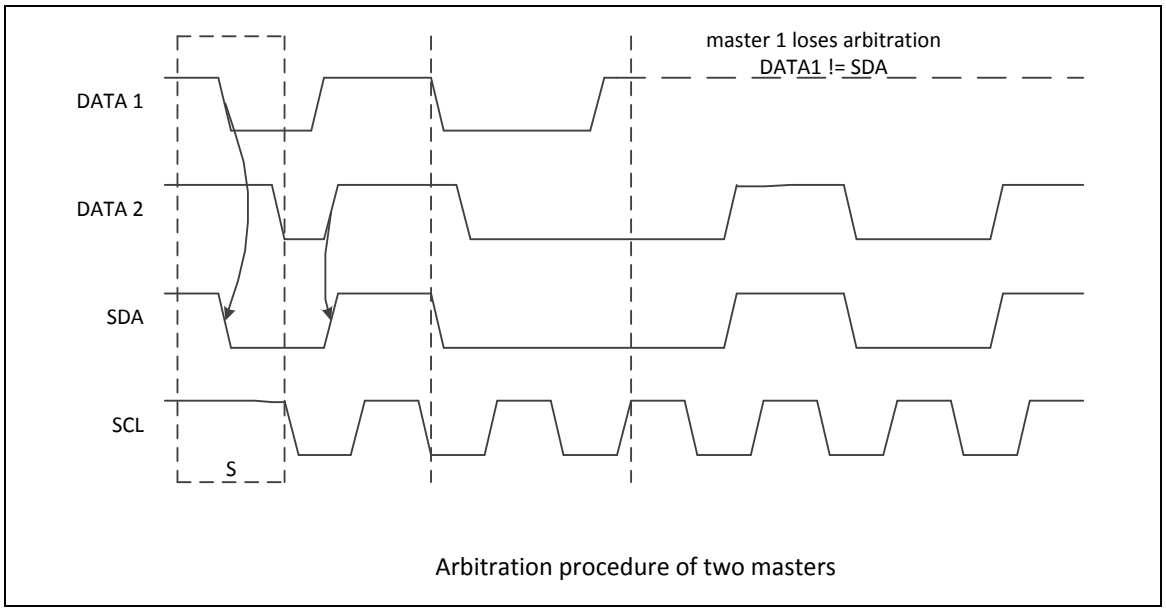


Figure 6.13-17 Arbitration Lost

- 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 error, STO should be set and SI should be cleared, and then STO is cleared to release bus.
  - Set (STA, STO, SI, AA) = (0, 1, 1, X) to stop current transfer
  - Set (STA, STO, SI, AA) = (0, 0, 1, X) to release bus

6.13.5.2.6 Bus Management (SMBus/PMBus Compatible)

This section is relevant only when Bus Management feature is supported.

Introduction

The Bus Management is an I^2C interface through which various devices can communicate with each other and with the rest of the system. It is based on I^2C principles of operation. The Bus Management provides a control bus for system and power management related tasks.

This peripheral is compatible with the SMBUS specification rev 2.0 (*http://smbus.org/specs/*) and PMBUS specification rev 1.2 (http://pmbus.org/).

The System Management Bus Specification refers to three types of devices.

- A slave is a device that receives or responds to a command.
- A master is a device that issues commands, generates the clocks and terminates the transfer.
- A host is a specialized master that provides the main interface to the system's CPU. A host must be a master-slave and must support the SMBus host notify protocol. Only one host is allowed in a system.

This Bus Management peripheral is based on I^2C specification rev 2.1.

Device Identification – slave address

Any device that exists on the Bus Management as a slave has a unique address called the Slave Address. For reference, the following addresses are reserved and must not be used by or assign to any Bus Management device. (Refer to SMBus specification for detail information)

Slave Address Bits 7-1	R/W Bit Bit 0	Comment
0000 000	0	General Call Address
0000 000	1	START byte
0000 001	X	CBUS address
0000 010	X	Address reserved for different bus format
0000 011	X	Reserved for future use
0000 1XX	X	Reserved for future use
0101 000	X	Reserved for ACCESS.bus host
0110 111	X	Reserved for ACCESS.bus default address
1111 0XX	X	10-bit slave addressing
1111 1XX	X	Reserved for future use
0001 000	X	SMBus Host
0001 100	X	SMBus Alert Response Address
1100 001	X	SMBus Device Default Address

Table 6.13.5-1 Reserved SMBus Address

Bus protocols

There are eleven possible command protocols for any given device. A device may use any or all of the eleven protocols to communicate. The protocols are **Quick Command**, **Send Byte**, **Receive Byte**, **Write Byte**, **Write Word**, **Read Byte**, **Read Word**, **Process Call**, **Block Read**, **Block Write** and **Block Write-Block Read Process Call**. These protocols should be implemented by the user software. (**For more details of these protocols, refer to SMBus specification ver. 2.0**)

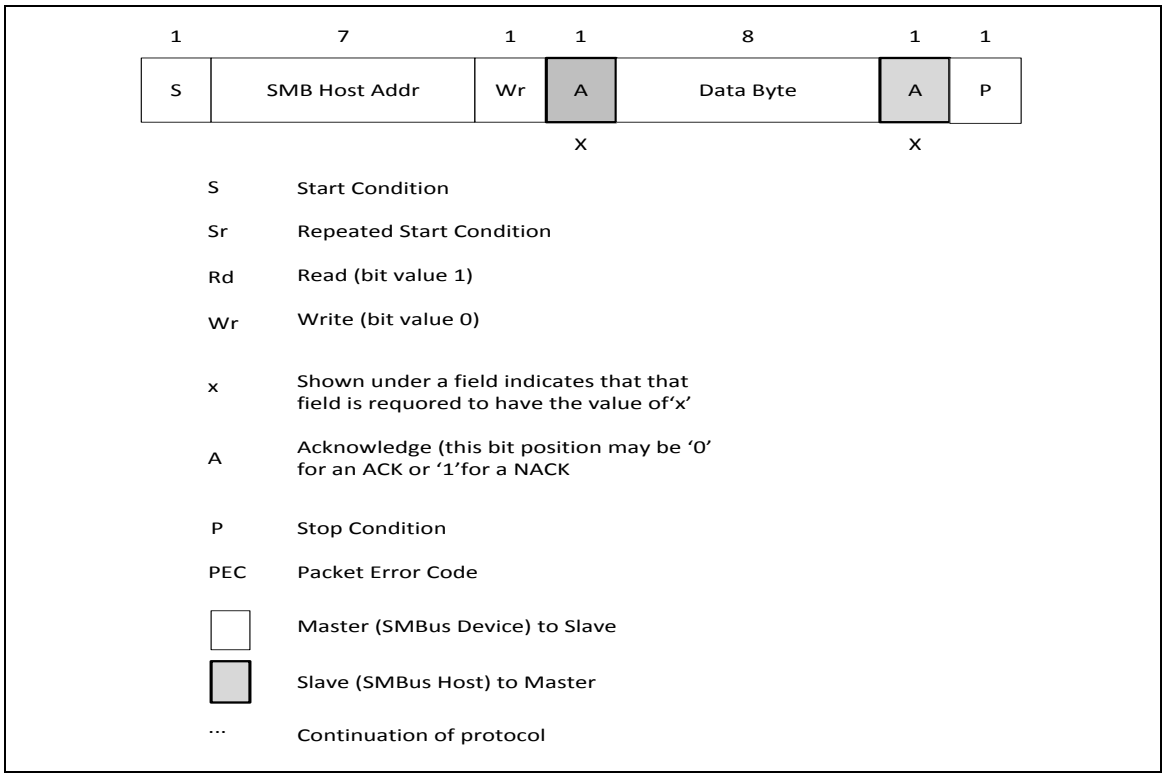


Figure 6.13-18 Bus Management Packet Protocol Diagram Element Key

Address resolution protocol (ARP)

Bus Management slave address conflicts can be resolved by dynamically assigning a new unique address to each slave device. In order to provide a mechanism to isolate each device for the purpose of address assignment each device must implement a unique device identifier (UDID). This 128-bit number is implemented by software.

This peripheral supports the Address Resolution Protocol (ARP). The Bus Management Device Default Address (0b1100 001) is enabled by setting BUSEN (I2C_BUSCTL[7]), BMDEN (I2C_BUSCTL[2]) and ALERTEN (I2C_BUSCTL[4]) bits. The ARP commands should be implemented by the user software. Arbitration is also performed in slave mode for ARP support.

Received Command and Data acknowledge control

A Bus Management receiver must be able to NACK each received command or data. In order to allow the ACK control in slave mode, the Slave Byte Control mode must be enabled by setting ACKMEN bit (I2C_BUSCTL[0]).

Host Notify protocol

To prevent message coming to the Bus Management host controller from unknown devices in unknown formats only one method of communication is allowed, a modified form of the Write Word protocol. The standard Write Word protocol is modified by replacing the command code with the alerting device's address.

This peripheral supports the Host Notify protocol by setting the BUSEN (I2C_BUSCTL[7]), BMHEN (I2C_BUSCTL[3]) and ALERTEN (I2C_BUSCTL[4]). In this case the host will acknowledge the Bus Management Host address (0001 000b). This protocol is used when the device acts as a master and the host as a slave.

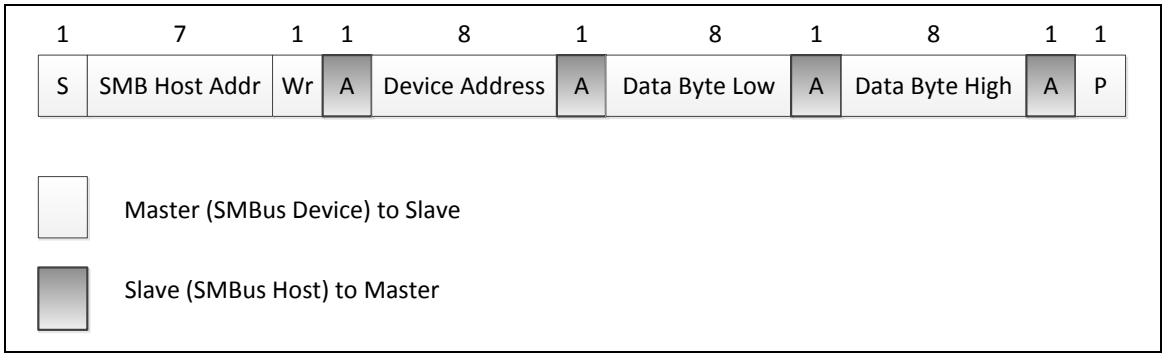


Figure 6.13-19 7-Bit Addressable Device to Host Communication

Bus Management Alert

The Bus Management ALERT optional signal is supported. A slave-only device can signal the host through the Bus Management ALERT (I2Cn_SMBAL, n=0 or 1) pin that it wants to talk. The host processes the interrupt and simultaneously accesses all Bus Management ALERT pin's devices through the Alert Response Address (0001 100b). Only the device(s) which pulled Bus Management ALERT pin low will acknowledge the Alert Response Address.

When configured as a slave device(BMHEN=0), the Bus Management ALERT pin is pulled low by setting the ALERTEN bit (I2C_BUSCTL[4]). The Alert Response Address (ARA) is enabled at the same time.

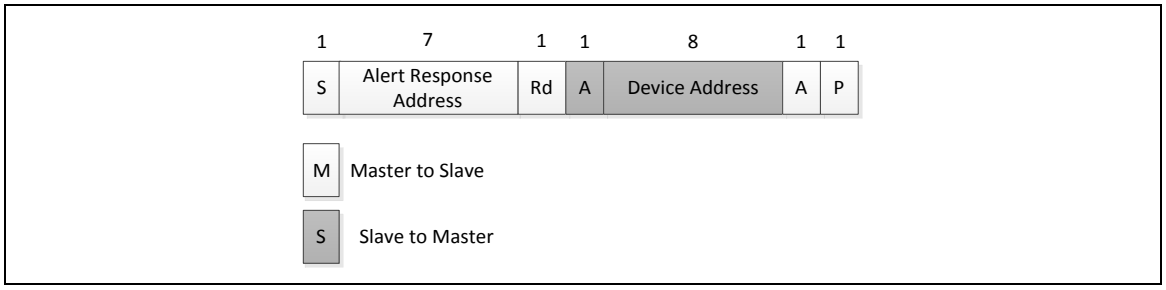


Figure 6.13-20 7-Bit Addressable Device Responds to an ARA

When configured as a host (BMHEN=1), the ALERT flag (I2C_BUSSTS[3]) is set when a falling edge is detected on the Bus Management ALERT pin and ALERTEN=1. When ALERTEN=0, the ALERT line is considered high even if the external Bus Management ALERT pin is low. If the Bus Management ALERT pin is not needed, the Bus Management ALERT pin can be used as a standard GPIO if ALERTEN = 0;

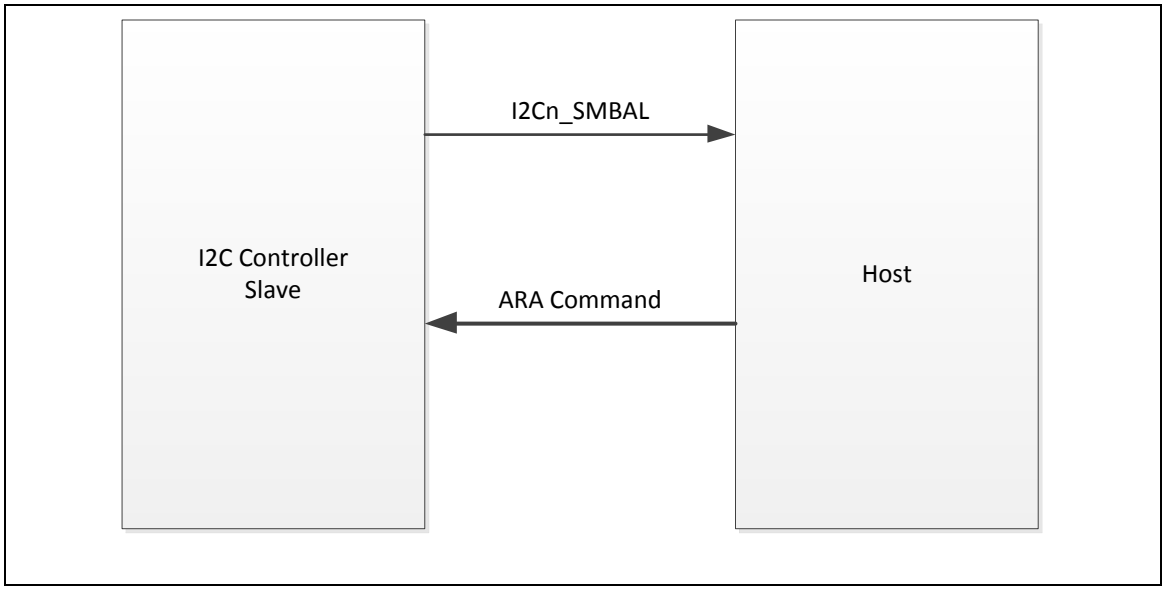


Figure 6.13-21 Bus Management ALERT function

Packet error checking

A packet error checking mechanism has been introduced in the SMBus specification to improve reliability and communication robustness. Packet Error Checking is implemented by appending a Packet Error Code (PEC) at the end of each message transfer. The PEC is calculated by using the $C(x) = x^8 + x^2 + x + 1$ CRC-8 polynomial on all the message bytes (including addresses and read/write bits).

The peripheral embeds a hardware PEC calculator when the PECEN bit (I2C_BUSCTL[1]) is set and allows to send a Not Acknowledge automatically when the received byte does not match with the hardware calculated PEC. The calculated value of PEC also can be read back on I2C_PKTCRC.

Time-out

This peripheral embeds hardware timers in order to be compliant with the 3 time-outs defined in SMBus specification ver. 2.0.

Bus management time-out:

1. The SCLK low time-out condition when bus no IDLE

$T_{Time\text{-}out}$ = BUSTO(I2C_BUSTOUT[7:0]) x 16x1024 (14-bit) x T_{PCLK} (if TOCDIV4 = 0).

= BUSTO(I2C_BUSTOUT[7:0] x 16x1024 (14-bit) x 4 x T_{PCLK} (if TOCDIV4 = 1)

2. The bus idle condition (both SCLK and SDA high) when bus IDLE

$T_{Time\text{-}out}$ = BUSTO(I2C_BUSTOUT[7:0]) x 4 x T_{PCLK}.

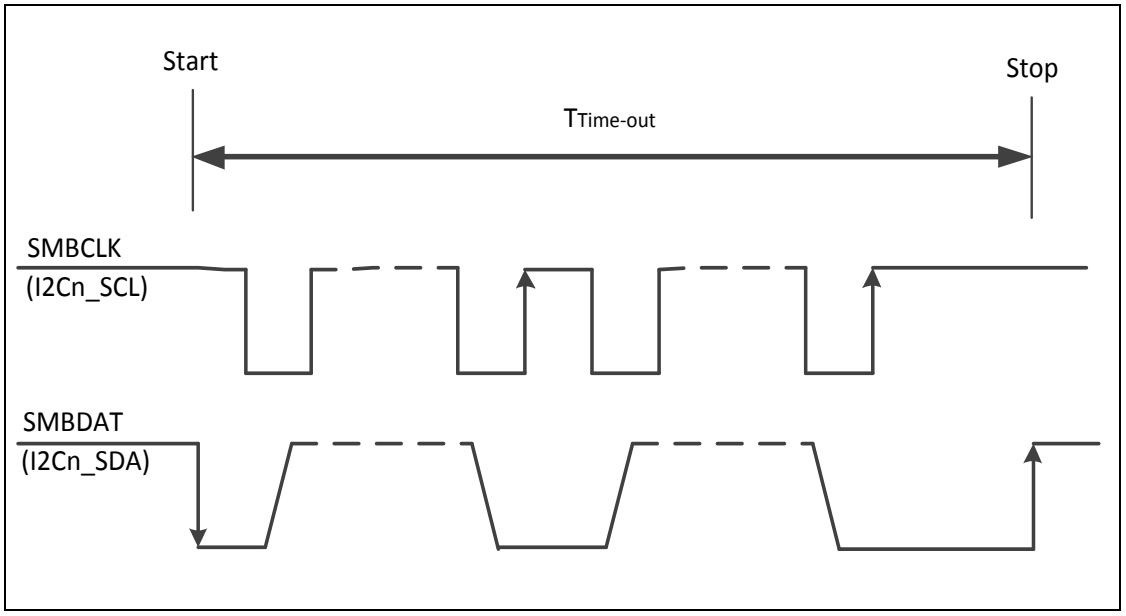


Figure 6.13-22 Bus Management Time Out Timing

Bus clock low time-out:

In Master mode, the Master cumulative clock low extend time ($T_{LOW:MEXT}$) is detected

In Slave mode, the slave cumulative clock low extend time ($T_{LOW:SEXT}$) is detected

$T_{TLOW:EXT}$ = CLKTO (I2C_CLKTOUT[7:0]) x 16x1024 (14-bit) x T_{PCLK} (if TOCDIV4= 0).

= CLKTO (I2C_CLKTOUT[7:0]) x 16x1024 (14-bit) x 4 x T_{PCLK} (if TOCDIV4= 1)

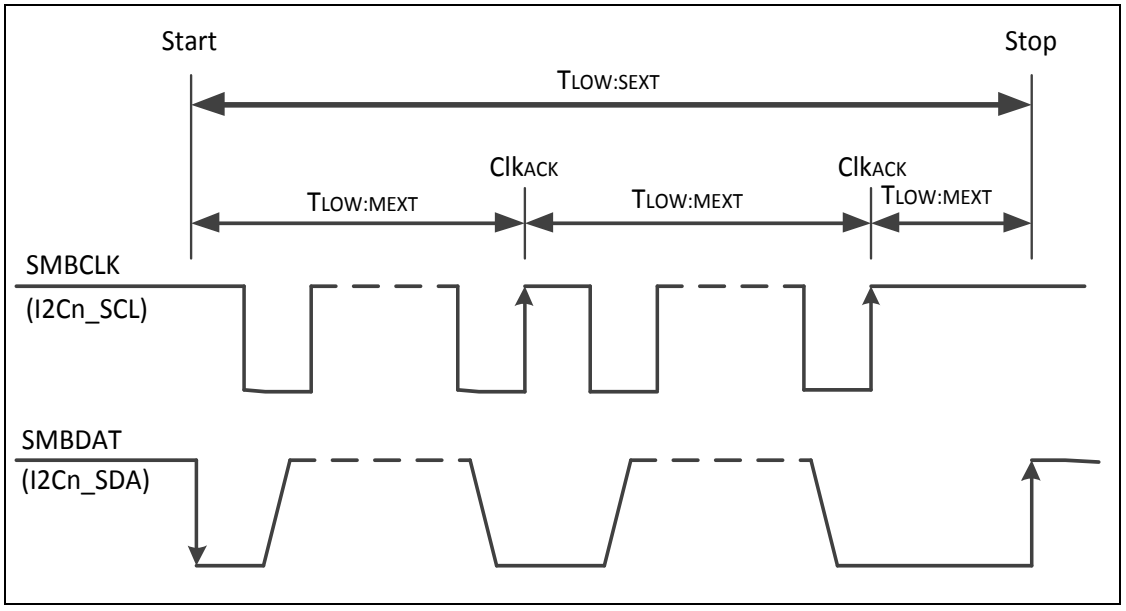


Figure 6.13-23 Bus Clock Low Time Out Timing

Bus idle detection

A master can assume that the bus is free if it detects that the clock and data signals have been high for T_{IDLE} greater than $T_{HIGH,MAX}$.

This timing parameter covers the condition where a master has been dynamically added to the

bus and may not have detected a state transition on the SMBCLK or SMBDAT lines. In this case, the master must wait long enough to ensure that a transfer is not currently in progress. The peripheral supports a hardware bus idle detection.

6.13.5.3 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. And once hold time configuration 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 6.13.5-2 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 our design, but user should follow I^2C protocol standard.

I^2C Baud Rate / PCLK	100k	200k	400k	800k	1200k
12 MHz	120	60	30	15	10
24 MHz	240	120	60	30	20
48 MHz	480	240	120	60	40
72 MHz	720	360	180	90	60

Table 6.13.5-2 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.

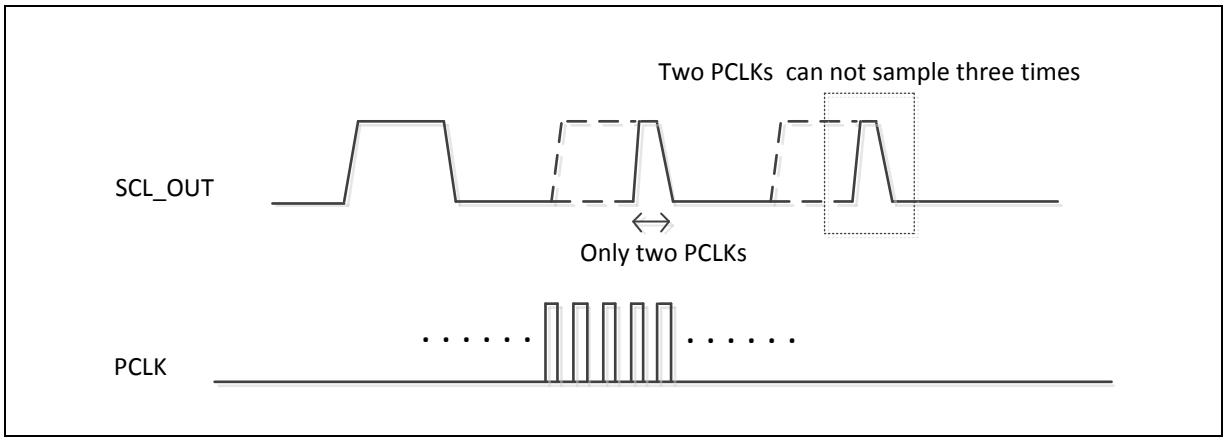


Figure 6.13-24 Setup Time Wrong Adjustment

For hold time wrong adjustment example, we use I²C Baud Rate = 1200k and PCLK = 72 MHz, 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.

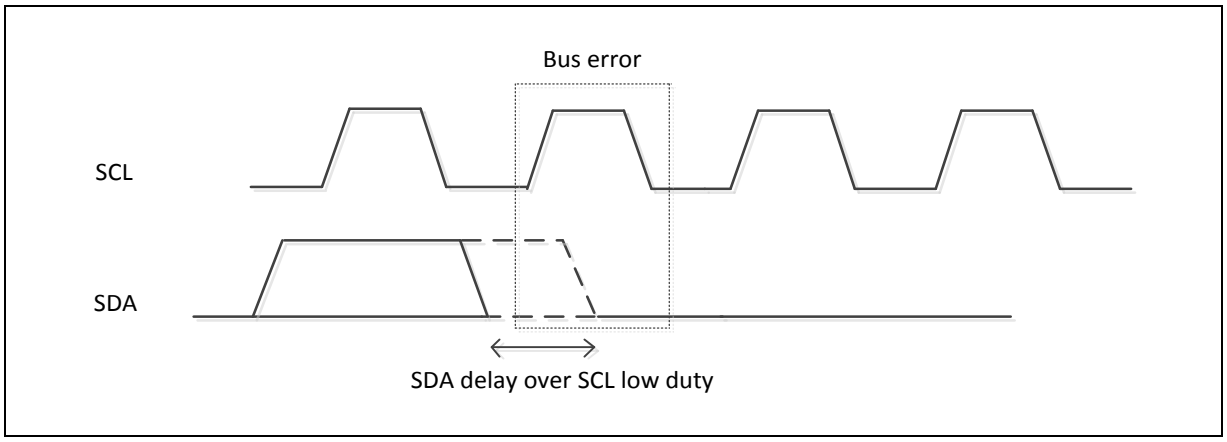


Figure 6.13-25 Hold Time Wrong Adjustment

6.13.5.4 I²C Protocol Registers

To control I²C 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 rate register), I2C_TOCTL (Time-out control register), I2C_WKCTL(wake up control register) and I2C_WKSTS(wake up status register).

6.13.5.4.1 Address Registers (I2C_ADDR)

The I²C port is equipped with four slave address registers, I2C_ADDRn (n=0~3). The contents of the register are irrelevant when I²C 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 I²C hardware will react if the contents of I2C_ADDRn are matched with the received slave address.

The I²C ports support the "General Call" function. If the GC bit (I2C_ADDRn [0]) is set the I²C 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 I²C is in Slave mode, it can receive the general call address by 00H after Master send general call address to I²C bus, then it will follow status of GC mode.

6.13.5.4.2 Slave Address Mask Registers (I2C_ADDRMSK)

The I²C 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.

6.13.5.4.3 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 I²C 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^2C 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. The Figure 6.13-26 shows I^2C Data Shifting Direction.

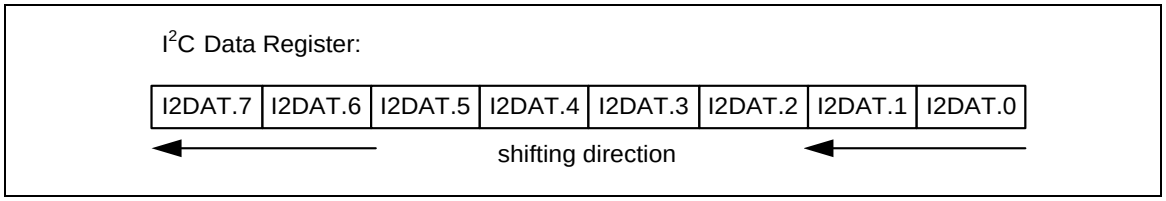


Figure 6.13-26 I^2C Data Shifting Direction

6.13.5.4.4 Control Register (I2C_CTL)

The CPU can be read from and written to I2C_CTL [7:0] directly. When the I^2C port is enabled by setting I2CEN (I2C_CTL [6]) to high, the internal states will be controlled by I2C_CTL and I^2C logic hardware.

There are two bits are affected by hardware: the SI bit is set when the I^2C 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^2C 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^2C 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.

6.13.5.4.5 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 6.13.5-3.When I2C_STATUS [7:0] is F8H, no serial interrupt is requested. All other I2C_STATUS [7:0] values correspond to the defined I^2C states. When each of these states is entered, a status interrupt is requested (SI = 1). A valid status code is present in 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^2C format frame. A Bus Error may occur during the serial transfer of an address byte, a data byte or an acknowledge bit. To recover I^2C from bus error, STO should be set and SI should be cleared to enter Not Addressed Slave mode. Then STO is cleared to release bus and to wait for a new communication. The I^2C bus cannot recognize stop condition during this action when a bus error occurs.

Master Mode	Slave Mode

STATUS	Description	STATUS	Description
0x08	Start	0xA0	Slave Transmit Repeat Start or Stop
0x10	Master Repeat Start	0xA8	Slave Transmit Address ACK
0x18	Master Transmit Address ACK	0xB8	Slave Transmit Data ACK
0x20	Master Transmit Address NACK	0xC0	Slave Transmit Data NACK
0x28	Master Transmit Data ACK	0xC8	Slave Transmit Last Data ACK
0x30	Master Transmit Data NACK	0x60	Slave Receive Address ACK
0x38	Master Arbitration Lost	0x68	Slave Receive Arbitration Lost
0x40	Master Receive Address ACK	0x80	Slave Receive Data ACK
0x48	Master Receive Address NACK	0x88	Slave Receive Data NACK
0x50	Master Receive Data ACK	0x70	GC mode Address ACK
0x58	Master Receive Data NACK	0x78	GC mode Arbitration Lost
0x00	Bus error	0x90	GC mode Data ACK
		0x98	GC mode Data NACK
		0xB0	Address Transmit Arbitration Lost
0xF0	If the BMDEN =1 and the ACKMEN bit is enabled, the information of I2C_STATUS will be fixed as 0xF0 in slave receive condition.		
0xF8	Bus Released **Note:** Status “0xF8” exists in both master/slave modes, and it won’t raise interrupt.		

Table 6.13.5-3 I^{2}C Status Code Description

6.13.5.4.6 Clock Baud Rate Bits (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^{2}C will automatically synchronize it with any clock frequency from master I^{2}C device. In the slave mode, system clock frequency should greater than I^{2}C bus maximum clock 20 times.

The data baud rate of I^{2}C setting is Data Baud Rate of I^{2}C = (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^{2}C = 16 MHz/ (4x (40 +1)) = 97.5 Kbits/sec.

6.13.5.4.7 Time-out Control Register (I2C_TOCTL)

There is a 14-bit time-out counter which can be used to deal with the I^{2}C bus hang-up. If the time-out counter is enabled, the counter starts up counting until it overflows (TOIF=1) and generates I^{2}C 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^{2}C 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^{2}C interrupt. Refer to the Figure 6.13-27 for the 14-bit time-out counter. User may write 1 to clear TOIF to 0.

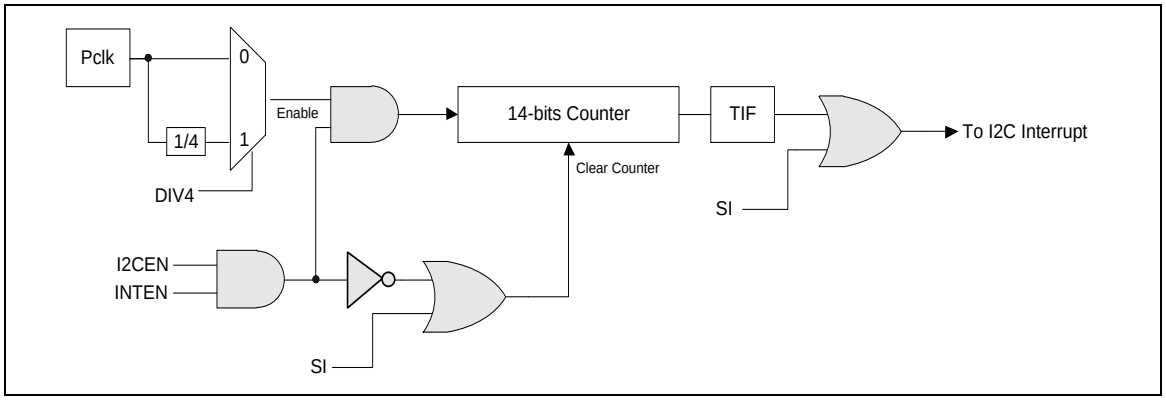


Figure 6.13-27 I²C Time-out Count Block Diagram

6.13.5.4.8 Wake-up Control Register (I2C_WKCTL)

When chip enters Power-down mode and set WKEN (I2C_WKCTL [0]) to 1, other I²C master can wake up our chip by addressing our I²C 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 I²C controller will go ahead. If NHDBUSEN (I2C_WKCTL [7]) is set, the controller will don't stretch the SCL to low. Notice that when the controller don't 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 I²C controller and execute the original operation again.

6.13.5.4.9 Wake-up Status Register (I2C_WKSTS)

When system is woken up by other I²C master device, WKIF 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. Notice that user can't release WKIF through clearing the WKAKDONE bit to 0.

The WRSTSWK (I2C_WKSTS [2]) bit records the Read/Write command before the I²C 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. Notice 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 I²C master device, WKIF is set to indicate this event. User needs write "1" to clear this bit.

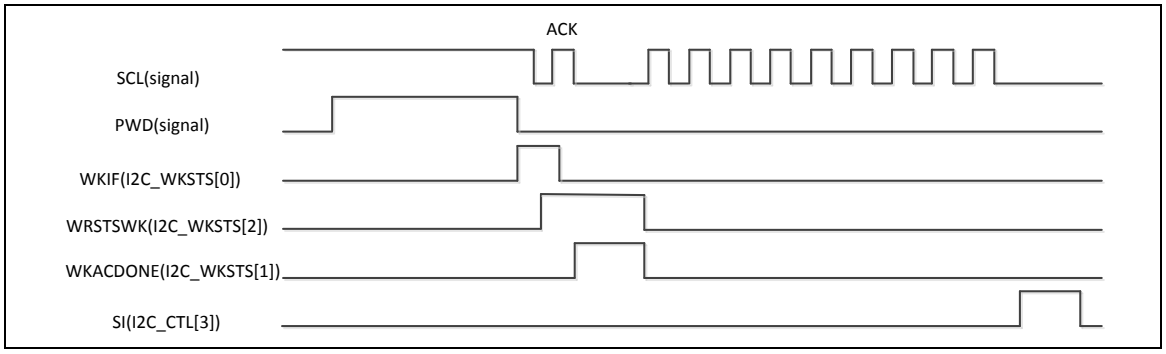


Figure 6.13-28 I²C Wake-Up Related Signals Waveform

6.13.5.4.10 The I²C Control Register 1 (I2C_CTL1)

If the enable 10-bit mode ADDR10EN (I2C_CTL1 [9]) is set, the I²C will run in 10-bit mode.

6.13.5.4.11 The I²C Status Register 1 (I2C_STATUS1)

The I²C controller supports four slave address flag registers, ADMAT0, ADMAT1, ADMAT2 and ADMAT3 (I2C_STATUS1[3:0]). Every control register represent which address is used and set 1 to inform software.

6.13.5.4.12 The 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.

6.13.5.4.13 Bus Management Control Register (I2C_BUSCTL)

The SM bus management control events are defined in this register. It includes the Acknowledge Control by Manual (ACKMEN (I2C_BUSCTL[0])), Packet Error Checking Enable (PECEN (I2C_BUSCTL[1])), device (BMDEN(I2C_BUSCTL[2])) or host (BMHEN (I2C_BUSCTL[3])) enable in this peripheral device. Both the alert and the suspend function can be set in ALERTEN (I2C_BUSCTL[4]), SCTLOSTS (I2C_BUSCTL[5])) and SCTLOEN (I2C_BUSCTL[6]).

The calculated PEC (when the PECEN is set) value is transmitted or received can be controlled by PECTXEN bit (I2C_BUSCTL[8]).

There is a special bit of ACKM9SI (I2C_BUSCTL[11]). When the ACKMEN is set, there is SI interrupt in the 8th clock input and the user can read the data and status register. If the 8th clock bus is released when the SI interrupt is cleared, there is another SI interrupt event in the 9th clock cycle when this bit is set to 1 to know the bus status in this transaction frame done.

Set the PECDIEN (I2C_BUSCTL[13]), BCDIEN (I2C_BUSCTL[12]) or PECCLR (I2C_BUSCTL[10]) for PEC control flow.

6.13.5.4.14 I²C Bus Management Timer Control Register (I2C_BUSTCTL)

Set TORSTEN (I2C_BUSTCTL[4]), CLKTOIEN (I2C_BUSTCTL[3]), BUSTOIEN (I2C_BUSTCTL[2]), CLKTOEN (I2C_BUSTCTL[1]) and BUSTOEN (I2C_BUSTCTL[0]) for bus time-out or clock low time-out control flow.

6.13.5.4.15 I²C Bus Management Status Register (I2C_BUSSTS)

Monitor the PECDONE (I2C_BUSSTS[7]), BCDONE (I2C_BUSSTS[1]) or PECERR

(I2C_BUSSTS[2]) for PEC control flow.

Monitor the SCTLDIN (I2C_BUSSTS[4]) for I2Cn_SMBSUS input status.

6.13.5.4.16 I²C Byte Number Register (I2C_PKTSIZE)

When the PECEN bit (I2C_BUSCTL[1]) is set. The I²C controller will calculate the PEC value of the data on the bus. The PLDSIZE (I2C_PKTSIZE[8:0]) is used to define the data number in the bus. When the counter reach the value of PLDSIZE, the final PEC value will be transmitted or received automatically when the PECTXEN bit (I2C_BUSCTL[8]) is set.

6.13.5.4.17 I²C PEC VALUR Register (I2C_PKTCRC)

The register indicates the calculated PECCRC (I2C_PKTCRC[7:0]) value of data on the I²C bus. The detail of information is defined the PEC section of SM Bus.

6.13.5.4.18 I²C Bus Management Timer and I²C Clock Low Timer Register (I2C_BUSTOUT/ I2C_CLKTOUT)

Both of the definitions of these registers are described in the time-out section of SM Bus.

6.13.5.5 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 by GPIO multiple function control registers.
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 6.13-29 shows the EEPROM random read operation.

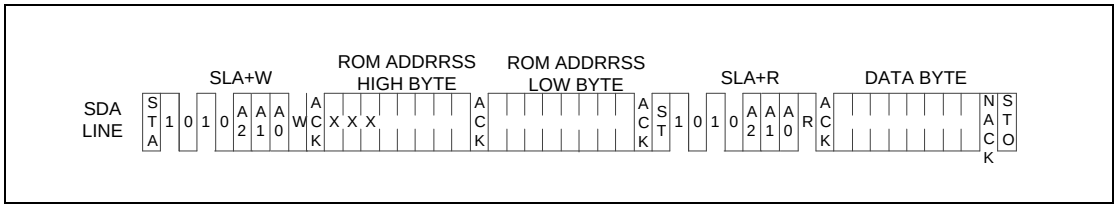


Figure 6.13-29 EEPROM Random Read

Figure 6.13-30 shows how to use I²C controller to implement the protocol of EEPROM random read.

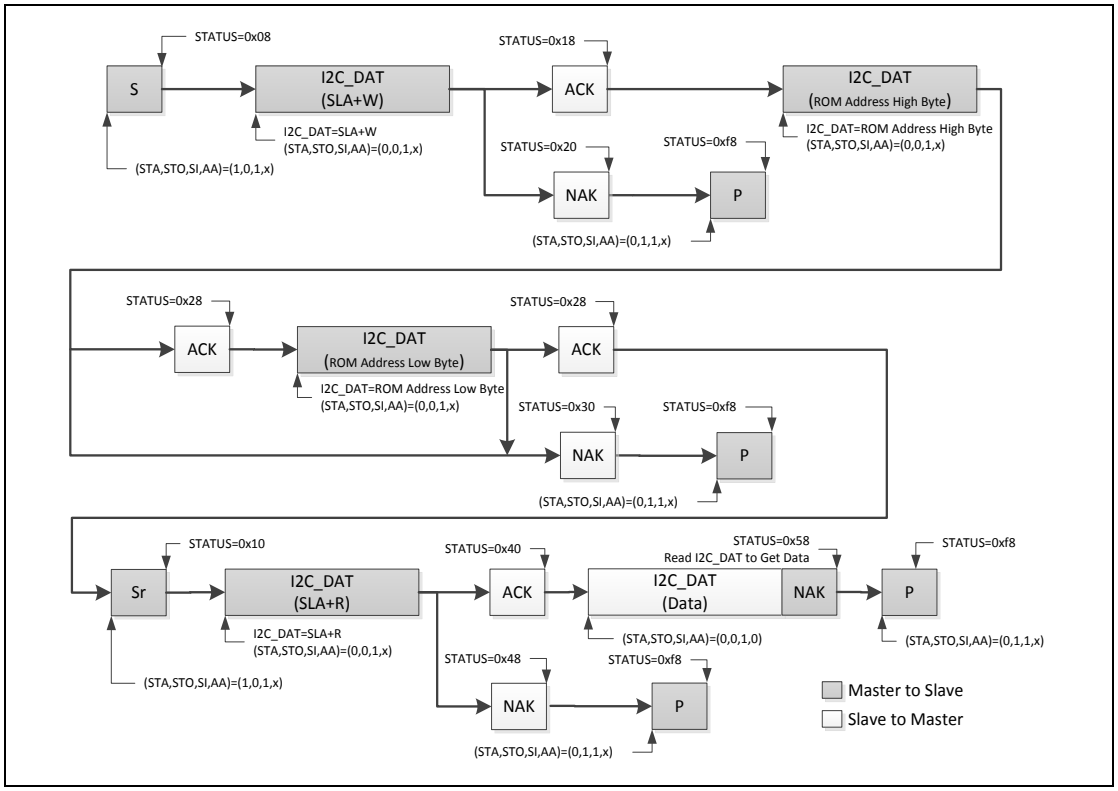


Figure 6.13-30 Protocol of EEPROM Random Read

The I^2C 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.

6.13.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
I²C Base Address: **I2Cn_BA = 0x4008_0000 + (0x1000 *n)** **n= 0,1**				
I2C_CTL	I2Cn_BA+0x00	R/W	I²C Control Register 0	0x0000_0000
I2C_ADDR0	I2Cn_BA+0x04	R/W	I²C Slave Address Register0	0x0000_0000
I2C_DAT	I2Cn_BA+0x08	R/W	I²C Data Register	0x0000_0000
I2C_STATUS	I2Cn_BA+0x0C	R	I²C Status Register 0	0x0000_00F8
I2C_CLKDIV	I2Cn_BA+0x10	R/W	I²C Clock Divided Register	0x0000_0000
I2C_TOCTL	I2Cn_BA+0x14	R/W	I²C Time-out Control Register	0x0000_0000
I2C_ADDR1	I2Cn_BA+0x18	R/W	I²C Slave Address Register1	0x0000_0000
I2C_ADDR2	I2Cn_BA+0x1C	R/W	I²C Slave Address Register2	0x0000_0000
I2C_ADDR3	I2Cn_BA+0x20	R/W	I²C Slave Address Register3	0x0000_0000
I2C_ADDRMSK0	I2Cn_BA+0x24	R/W	I²C Slave Address Mask Register0	0x0000_0000
I2C_ADDRMSK1	I2Cn_BA+0x28	R/W	I²C Slave Address Mask Register1	0x0000_0000
I2C_ADDRMSK2	I2Cn_BA+0x2C	R/W	I²C Slave Address Mask Register2	0x0000_0000
I2C_ADDRMSK3	I2Cn_BA+0x30	R/W	I²C Slave Address Mask Register3	0x0000_0000
I2C_WKCTL	I2Cn_BA+0x3C	R/W	I²C Wake-up Control Register	0x0000_0000
I2C_WKSTS	I2Cn_BA+0x40	R/W	I²C Wake-up Status Register	0x0000_0000
I2C_CTL1	I2Cn_BA+0x44	R/W	I²C Control Register 1	0x0000_0000
I2C_STATUS1	I2Cn_BA+0x48	R/W	I²C Status Register 1	0x0000_0000
I2C_TMCTL	I2Cn_BA+0x4C	R/W	I2C Timing Configure Control Register	0x0000_0000
I2C_BUSCTL	I2Cn_BA+0x50	R/W	I²C Bus Management Control Register	0x0000_0000
I2C_BUSTCTL	I2Cn_BA+0x54	R/W	I²C Bus Management Timer Control Register	0x0000_0000
I2C_BUSSTS	I2Cn_BA+0x58	R/W	I²C Bus Management Status Register	0x0000_0000
I2C_PKTSIZE	I2Cn_BA+0x5C	R/W	I²C Packet Error Checking Byte Number Register	0x0000_0000
I2C_PKTCRC	I2Cn_BA+0x60	R	I²C Packet Error Checking Byte Value Register	0x0000_0000
I2C_BUSTOUT	I2Cn_BA+0x64	R/W	I²C Bus Management Timer Register	0x0000_0005
I2C_CLKTOUT	I2Cn_BA+0x68	R/W	I²C Bus Management Clock Low Timer Register	0x0000_0005

Note:

1. Any register not listed here is reserved and must not be written. The result of a read operation on these bits is undefined.

2. The reserved register fields that listed in register description must be written to their reset value. Writing reserved fields with other than reset values may produce undefined results.

6.13.7 Register Description

I2C Control Register (I2C_CTL)

Register	Offset	R/W	Description	Reset Value
I2C_CTL	I2Cn_BA+0x00	R/W	I²C 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. Any values read should be ignored. When writing to this field always write with reset value.
[7]	**INTEN**	**Enable Interrupt** 0 = I²C interrupt Disabled. 1 = I²C interrupt Enabled.
[6]	**I2CEN**	**I²C Controller Enable Bit** Set to enable I²C serial function controller. When I2CEN=1 the I²C serial function enable. The multi-function pin function must set to SDA, and SCL of I²C function first. 0 = I²C controller Disabled. 1 = I²C controller Enabled.
[5]	**STA**	**I²C START Control** Setting STA to logic 1 to enter Master mode, the I²C hardware sends a START or repeat START condition to bus when the bus is free. This bit will be cleared by hardware automatically.
[4]	**STO**	**I²C STOP Control** In Master mode, setting STO to transmit a STOP condition to bus then I²C controller will check the bus condition if a STOP condition is detected. This bit will be cleared by hardware automatically.
[3]	**SI**	**I²C Interrupt Flag** When a new I²C state is present in the I2C_STATUS register, the SI flag is set by hardware. If bit INTEN (I2C_CTL [7]) is set, the I²C interrupt is requested. SI must be cleared by software. Clear SI by writing 1 to this bit. For ACKMEN is set in slave read mode, the SI flag is set in 8th clock period for user to confirm the acknowledge bit and 9th clock period for user to read the data in the data buffer.
[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. Any values read should be ignored. When writing to this field always write with reset value.

I2C Data Register (I2C_DAT)

Register	Offset	R/W	Description	Reset Value
I2C_DAT	I2Cn_BA+0x08	R/W	I^2C 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. Any values read should be ignored. When writing to this field always write with reset value.
[7:0]	**DAT**	**I^2C Data** Bit [7:0] is located with the 8-bit transferred/received data of I^2C serial port.

I2C Status Register (I2C_STATUS)

Register	Offset	R/W	Description	Reset Value
I2C_STATUS	I2Cn_BA+0x0C	R	I^2C 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. Any values read should be ignored. When writing to this field always write with reset value.
[7:0]	**STATUS**	**I^2C 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^2C 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	I^2C 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						**DIVIDER**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DIVIDER							

Bits	Description	
[31:10]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[9:0]	**DIVIDER**	**I^2C Clock Divided** Indicates the I^2C clock rate: Data Baud Rate of I^2C = (system clock) / (4x (I2C_CLKDIV+1)). **Note:** The minimum value of I2C_CLKDIV is 4.

I2C Time-out Control Register (I2C_TOCTL)

Register	Offset	R/W	Description	Reset Value
I2C_TOCTL	I2Cn_BA+0x14	R/W	I^2C 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. Any values read should be ignored. When writing to this field always write with reset value.
[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 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^2C time-out happened and it can interrupt CPU if I^2C interrupt enable bit (INTEN) is set to 1. **Note:** Software can write 1 to clear this bit.

I2C Slave Address Register (ADDRx)

Register	Offset	R/W	Description	Reset Value
I2C_ADDR0	I2Cn_BA+0x04	R/W	I^2C Slave Address Register0	0x0000_0000
I2C_ADDR1	I2Cn_BA+0x18	R/W	I^2C Slave Address Register1	0x0000_0000
I2C_ADDR2	I2Cn_BA+0x1C	R/W	I^2C Slave Address Register2	0x0000_0000
I2C_ADDR3	I2Cn_BA+0x20	R/W	I^2C 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					**ADDR**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ADDR							**GC**

Bits	Description	
[31:11]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[10:1]	**ADDR**	**I^2C Address** The content of this register is irrelevant when I^2C is in Master mode. In the slave mode, the seven most significant bits must be loaded with the chip's own address. The I^2C hardware will react if either of the address is matched. **Note:** When software set 10'h000, the address cannot be used.
[0]	**GC**	**General Call Function** 0 = General Call Function Disabled. 1 = General Call Function Enabled.

I2C Slave Address Mask Register (ADDRMSKx)

Register	Offset	R/W	Description	Reset Value
I2C_ADDRMSK0	I2Cn_BA+0x24	R/W	I²C Slave Address Mask Register0	0x0000_0000
I2C_ADDRMSK1	I2Cn_BA+0x28	R/W	I²C Slave Address Mask Register1	0x0000_0000
I2C_ADDRMSK2	I2Cn_BA+0x2C	R/W	I²C Slave Address Mask Register2	0x0000_0000
I2C_ADDRMSK3	I2Cn_BA+0x30	R/W	I²C Slave Address Mask Register3	0x0000_0000

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

Bits	Description	
[31:11]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[10:1]	**ADDRMSK**	**I²C 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.). I²C bus controllers support multiple address recognition with four address mask register. When the bit in the address mask register is set to one, it means the received corresponding address bit is don't-care. If the bit is set to zero, that means the received corresponding register bit should be exact the same as address register. **Note:** The wake-up function cannot use address mask.
[0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

I2C Wake-up Control Register (I2C_WKCTL)

Register	Offset	R/W	Description	Reset Value
I2C_WKCTL	I2Cn_BA+0x3C	R/W	I^2C 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. Any values read should be ignored. When writing to this field always write with reset value.
[7]	**NHDBUSEN**	**I^2C No Hold BUS Enable Bit** 0 = I^2C hold bus after wake-up 1= I^2C don't hold bus after wake-up **Note:** I^2C 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^2C controller and execute the original operation again.
[6:1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**WKEN**	**I^2C Wake-up Enable Bit** 0 = I^2C wake-up function Disabled. 1= I^2C wake-up function Enabled.

I2C Wake-up Status Register (I2C_WKSTS)

Register	Offset	R/W	Description	Reset Value
I2C_WKSTS	I2Cn_BA+0x40	R/W	I^2C 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. Any values read should be ignored. When writing to this field always write with reset value.
[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^2C Wake-up Flag** When chip is woken up from Power-down mode by I^2C, this bit is set to 1. Software can write 1 to clear this bit.

I2C 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						**ADDR10EN**	**Reserved**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:10]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[9]	**ADDR10EN**	**Address 10-bit Function Enable** 0 = Address match 10-bit function is disabled. 1 = Address match 10-bit function is enabled.
[8:0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

I2C Status Register 1 (I2C_STATUS1)

Register	Offset	R/W	Description	Reset Value
I2C_STATUS1	I2Cn_BA+0x48	R/W	I²C 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
Reserved				ADMAT3	ADMAT2	ADMAT1	ADMAT0

Bits	Description	
[31:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8]	**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 or arbitration lost condition occured. 0 = The bus is IDLE (both SCLK and SDA High). 1 = The bus is busy. **Note:** This bit is read only.
[7:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[3]	**ADMAT3**	**I²C Address 3 Match Status Register** When address 3 is matched, hardware will inform which address used. This bit will set to 1, and software can write 1 to clear this bit.
[2]	**ADMAT2**	**I²C Address 2 Match Status Register** When address 2 is matched, hardware will inform which address used. This bit will set to 1, and software can write 1 to clear this bit.
[1]	**ADMAT1**	**I²C 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**	**I²C 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.

I2C Timing Configure Control Register (I2C_TMCTL)

Register	Offset	R/W	Description	Reset Value
I2C_TMCTL	I2Cn_BA+0x4C	R/W	I^2C Timing Configure Control Register	0x0000_0000

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

Bits	Description	
[31:25]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[24:16]	**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.
[15:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8:0]	**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.

I2C Bus Manage Control Register (I2C_BUSCTL)

Register	Offset	R/W	Description	Reset Value
I2C_BUSCTL	I2Cn_BA+0x50	R/W	I^2C Bus Management 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		**PECDIEN**	**BCDIEN**	**ACKM9SI**	**PECCLR**	**TIDLE**	**PECTXEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BUSEN	**SCTLOEN**	**SCTLOSTS**	**ALERTEN**	**BMHEN**	**BMDEN**	**PECEN**	**ACKMEN**

Bits	Description	
[31:14]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[13]	**PECDIEN**	**Packet Error Checking Byte Transfer Done Interrupt Enable Bit** 0 = Indicates the PEC transfer done interrupt is Disabled. 1 = Indicates the PEC transfer done interrupt is Enabled. **Note:** This bit is used in PECEN =1 and INTEN = 1.
[12]	**BCDIEN**	**Packet Error Checking Byte Count Done Interrupt Enable Bit** 0 = Indicates the byte count done interrupt is Disabled. 1 = Indicates the byte count done interrupt is Enabled. **Note:** This bit is used in PECEN =1 and INTEN = 1.
[11]	**ACKM9SI**	**Acknowledge Manual Enable Extra SI Interrupt** 0 = There is no SI interrupt in the 9th clock cycle when the BMDEN = 1, BUSEN =1 and ACKMEN =1. 1 = There is SI interrupt in the 9th clock cycle when the BMDEN = 1, BUSEN =1 and ACKMEN =1.
[10]	**PECCLR**	**PEC Clear at Repeat Start** The calculation of PEC starts when PECEN is set to 1 and it is clear when the STA or STO bit is detected. This PECCLR bit is used to enable the condition of REPEAT START can clear the PEC calculation. 0 = The PEC calculation is cleared by “Repeat Start” function is Disabled. 1 = The PEC calculation is cleared by “Repeat Start” function is Enabled.
[9]	**TIDLE**	**Timer Check in Idle State** The BUSTOUT is used to calculate the time-out of clock low in bus active and the idle period in bus Idle. This bit is used to define which condition is enabled. 0 = The BUSTOUT is used to calculate the clock low period in bus active. 1 = The BUSTOUT is used to calculate the IDLE period in bus Idle. **Note:** The BUSY (I2C_BUSSTS[0]) indicate the current bus state.

[8]	**PECTXEN**	**Packet Error Checking Byte Transmission/Reception** 0 = No PEC transfer. 1 = PEC transmission is requested. **Note:** 1.This bit has no effect in slave mode when ACKMEN =0.
[7]	**BUSEN**	**BUS Enable Bit** 0 = The system management function is Disabled. 1 = The system management function is Enable. **Note:** When the bit is enabled, the internal 14-bit counter is used to calculate the time out event of clock low condition.
[6]	**SCTLOEN**	**Suspend or Control Pin Output Enable Bit** 0 = The I2Cn_SMBSUS pin in input. 1 = The output enable is active on the I2Cn_SMBSUS pin.
[5]	**SCTLOSTS**	**Suspend/Control Data Output Status** 0 = The output of I2Cn_SMBSUS pin is low. 1 = The output of I2Cn_SMBSUS pin is high.
[4]	**ALERTEN**	**Bus Management Alert Enable Bit** Device Mode (BMHEN =0). 0 = Release the I2Cn_SMBAL pin high and Alert Response Header disabled: 0001100x followed by NACK if both of BMDEN and ACKMEN are enabled. 1 = Drive I2Cn_SMBAL pin low and Alert Response Address Header enables: 0001100x followed by ACK if both of BMDEN and ACKMEN are enabled. Host Mode (BMHEN =1). 0 = I2Cn_SMBAL pin not supported. 1 = I2Cn_SMBAL pin supported.
[3]	**BMHEN**	**Bus Management Host Enable Bit** 0 = Host function Disabled. 1 = Host function Enabled.
[2]	**BMDEN**	**Bus Management Device Default Address Enable Bit** 0 = Device default address Disable. When the address 0'b1100001x coming and the both of BMDEN and ACKMEN are enabled, the device responses NACKed 1 = Device default address Enabled. When the address 0'b1100001x coming and the both of BMDEN and ACKMEN are enabled, the device responses ACKed.
[1]	**PECEN**	**Packet Error Checking Calculation Enable Bit** 0 = Packet Error Checking Calculation Disabled. 1 = Packet Error Checking Calculation Enabled. **Note:** When I^2C enter powerdown mode, the bit should be enabled after wake-up if needed PEC calculation.
[0]	**ACKMEN**	**Acknowledge Control by Manual** In order to allow ACK control in slave reception including the command and data, slave byte control mode must be enabled by setting the ACKMEN bit. 0 = Slave byte control Disabled. 1 = Slave byte control Enabled. The 9th bit can response the ACK or NACK according the received data by user. When the byte is received, stretching the SCLK signal low between the 8th and 9th SCLK pulse. **Note:** If the BMDEN =1 and this bit is enabled, the information of I2C_STATUS will be fixed as 0xF0 in slave receive condition.

I2C Bus Management Timer Control Register (I2C_BUSTCTL)

Register	Offset	R/W	Description	Reset Value
I2C_BUSTCTL	I2Cn_BA+0x54	R/W	I^2C Bus Management Timer 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			**TORSTEN**	**CLKTOIEN**	**BUSTOIEN**	**CLKTOEN**	**BUSTOEN**

Bits	Description	
[31:5]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[4]	**TORSTEN**	**Time Out Reset Enable Bit** 0 = Indicates the I^2C state machine reset is Disable. 1 = Indicates the I^2C state machine reset is Enable. (The clock and data bus will be released to high)
[3]	**CLKTOIEN**	**Extended Clock Time Out Interrupt Enable Bit** 0 = Indicates the clock time out interrupt is Disabled. 1 = Indicates the clock time out interrupt is Enabled.
[2]	**BUSTOIEN**	**Time-out Interrupt Enable Bit** BUSY =1. 0 = Indicates the SCLK low time-out interrupt is Disabled. 1 = Indicates the SCLK low time-out interrupt is Enabled. BUSY =0. 0 = Indicates the bus IDLE time-out interrupt is Disabled. 1 = Indicates the bus IDLE time-out interrupt is Enabled.
[1]	**CLKTOEN**	**Cumulative Clock Low Time Out Enable Bit** 0 = Indicates the cumulative clock low time-out detection is Disabled. 1 = Indicates the cumulative clock low time-out detection is Enabled. For Master, it calculates the period from START to ACK For Slave, it calculates the period from START to STOP
[0]	**BUSTOEN**	**Bus Time Out Enable Bit** 0 = Indicates the bus clock low time-out detection is Disabled. 1 = Indicates the bus clock low time-out detection is Enabled (bus clock is low for more than TTime-out (in BIDLE=0) or high more than TTime-out(in BIDLE =1).

I2C Bus Management Status Register (I2C_BUSSTS)

Register	Offset	R/W	Description	Reset Value
I2C_BUSSTS	I2Cn_BA+0x58	R/W	I^2C Bus Management 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**
PECDONE	**CLKTO**	**BUSTO**	**SCTLDIN**	**ALERT**	**PECERR**	**BCDONE**	**BUSY**

Bits	Description	
[31:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7]	**PECDONE**	**PEC Byte Transmission/Receive Done** 0 = Indicates the PEC transmission/ receive is not finished when the PECEN is set. 1 = Indicates the PEC transmission/ receive is finished when the PECEN is set. **Note:** Software can write 1 to clear this bit.
[6]	**CLKTO**	**Clock Low Cumulate Time-out Status** 0 = Indicates that the cumulative clock low is no any time-out. 1 = Indicates that the cumulative clock low time-out occurred. **Note:** Software can write 1 to clear this bit.
[5]	**BUSTO**	**Bus Time-out Status** 0 = Indicates that there is no any time-out or external clock time-out. 1 = Indicates that a time-out or external clock time-out occurred. In bus busy, the bit indicates the total clock low time-out event occurred otherwise, it indicates the bus idle time-out event occurred. **Note:** Software can write 1 to clear this bit.
[4]	**SCTLDIN**	**Bus Suspend or Control Signal Input Status** 0 = The input status of I2Cn_SMBSUS pin is 0. 1 = The input status of I2Cn_SMBSUS pin is 1.
[3]	**ALERT**	**SMBus Alert Status** Device Mode (BMHEN =0). 0 = Indicates I2Cn_SMBAL pin state is low. 1 = Indicates I2Cn_SMBAL pin state is high. Host Mode (BMHEN =1). 0 = No SMBALERT event. 1 = Indicates there is SMBALERT event (falling edge) is detected in I2Cn_SMBAL pin when the BMHEN = 1 (SMBus host configuration) and the ALERTEN = 1.

		Note: 1. The I2Cn_SMBAL pin is an open-drain pin, the pull-high resistor is must in the system. 2. Software can write 1 to clear this bit.
[2]	**PECERR**	**PEC Error in Reception** 0 = Indicates the PEC value equal the received PEC data packet. 1 = Indicates the PEC value doesn't match the receive PEC data packet. **Note:** Software can write 1 to clear this bit.
[1]	**BCDONE**	**Byte Count Transmission/Receive Done** 0 = Indicates the byte count transmission/ receive is not finished when the PECEN is set. 1 = Indicates the byte count transmission/ receive is finished when the PECEN is set. **Note:** Software can write 1 to clear this bit.
[0]	**BUSY**	**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.

I2C Byte Number Register (I2C_PKTSIZE)

Register	Offset	R/W	Description	Reset Value
I2C_PKTSIZE	I2Cn_BA+0x5C	R/W	I^2C Packet Error Checking Byte Number Register	0x0000_0000

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

Bits	Description	
[31:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8:0]	**PLDSIZE**	**Transfer Byte Number** The transmission or receive byte number in one transaction when the PECEN is set. The maximum transaction or receive byte is 256 Bytes. **Note:** The byte number counting includes address, command code, and data frame.

I2C PEC Value Register (I2C_PKTCRC)

Register	Offset	R/W	Description	Reset Value
I2C_PKTCRC	I2Cn_BA+0x60	R	I^2C Packet Error Checking Byte Value Register	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7:0]	**PECCRC**	**Packet Error Checking Byte Value** This byte indicates the packet error checking content after transmission or receive byte count by using the $C(x) = X^8 + X^2 + X + 1$. It is read only.

I2C Bus Management Timer Register (I2C_BUSTOUT)

Register	Offset	R/W	Description	Reset Value
I2C_BUSTOUT	I2Cn_BA+0x64	R/W	I^2C Bus Management Timer Register	0x0000_0005

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

Bits	Description	
[31:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7:0]	**BUSTO**	**Bus Management Time-out Value** Indicate the bus time-out value in bus is IDLE or SCLK low. **Note 1:** If the user wants to revise the value of BUSTOUT, the TORSTEN (I2C_BUSTCTL[4]) bit shall be set to 1 and clear to 0 first in the BUSEN(I2C_BUSCTL[7]) is set. **Note 2:** BUSTO must be >= 1 before set BUSTOEN = 1.

I2C Clock Low Timer Register (I2C_CLKTOUT)

Register	Offset	R/W	Description	Reset Value
I2C_CLKTOUT	I2Cn_BA+0x68	R/W	I^2C Bus Management Clock Low Timer Register	0x0000_0005

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

Bits	Description	
[31:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7:0]	**CLKTO**	**Bus Clock Low Timer** The field is used to configure the cumulative clock extension time-out. **Note 1:** If the user wants to revise the value of CLKLTOUT, the TORSTEN bit shall be set to 1 and clear to 0 first in the BUSEN is set. **Note 2:** CLKTO must be >= 1 before set CLKTOEN = 1.

6.14 Serial Peripheral Interface (SPI)

6.14.1 Overview

The Serial Peripheral Interface (SPI) is a synchronous serial data communication interface. SPI devices communicate in full duplex mode using a master-slave architecture. The single master and the slave(s) communicate bi-directionally through a 4-wire interface.

The NPCA121 series contains up to three sets of SPI controllers which perform serial-to-parallel conversion when receiving data from a peripheral device, and parallel-to-serial conversion when transmitting data to a peripheral device. Each SPI controller can be configured as a master or a slave device.

SPI0 controller supports 2-bit Transfer mode to perform full-duplex 2-bit data transfer and also supports Dual and Quad I/O Transfer mode. SPI1 and SPI2 controllers support I^2S mode to connect external audio CODEC. Each SPI controller supports the PDMA function to access the data buffer.

6.14.2 Features

- SPI Mode
  - Up to three sets of SPI controllers
  - Supports Master or Slave mode operation
  - Master mode up to 25 MHz and Slave mode up to 25 MHz (when chip works at VDD = 2.7~3.6V)
  - Supports 2-bit Transfer mode (SPI0 Only)
  - Supports Dual and Quad I/O Transfer mode (SPI0 Only)
  - Configurable bit length of a transaction 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 Byte or Word Suspend mode
  - Supports PDMA transfer
  - Supports 3-Wire, no slave selection signal, bi-direction interface (SPI0 Only)
  - Supports one data channel half-duplex transfer
  - Support receive-only mode

- I^2S Mode (for SPI1~SPI2)
  - Supports Master or Slave
  - Capable of handling 8-, 16-, 24- and 32-bit word sizes
  - Each provides two 4-level FIFO data buffers, one for transmitting and the other for receiving
  - Supports monaural and stereo audio data
  - Supports PCM mode A, PCM mode B, I^2S and MSB justified data format
  - Supports two PDMA requests, one for transmitting and the other for receiving

	SPI0	SPI1 / SPI2
Dual/Quad I/O Mode	V	X
Two-Bit Transfer Mode	V	X
FIFO Depth	8-level	SPI mode 8~16 bits data length: 8-level Otherwise: 4-level
Slave Time-out Function	V	X
Slave 3-Wired Mode	V	X
I^2S Mode	X	V

Table 6.14.2-1 SPI feature difference (SPI0~SPI2)

6.14.3 Block Diagram

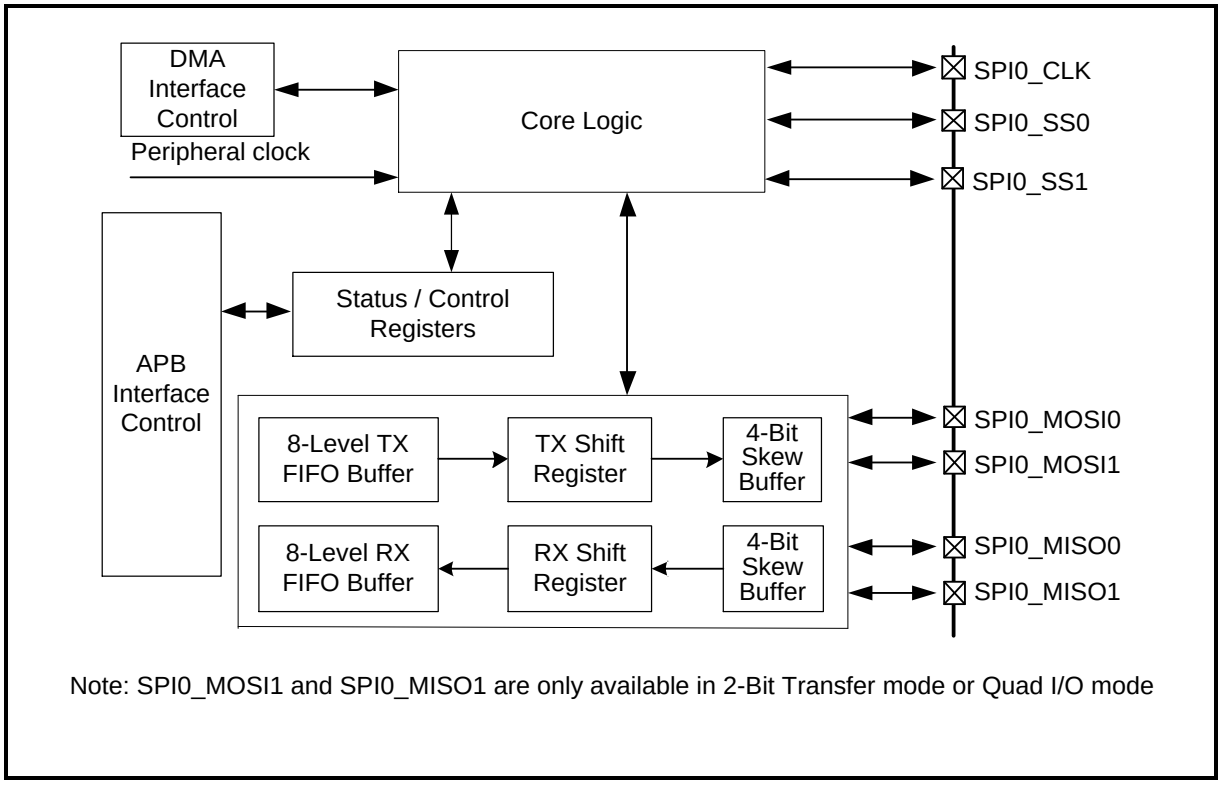


Figure 6.14-1 SPI Block Diagram (SPI0)

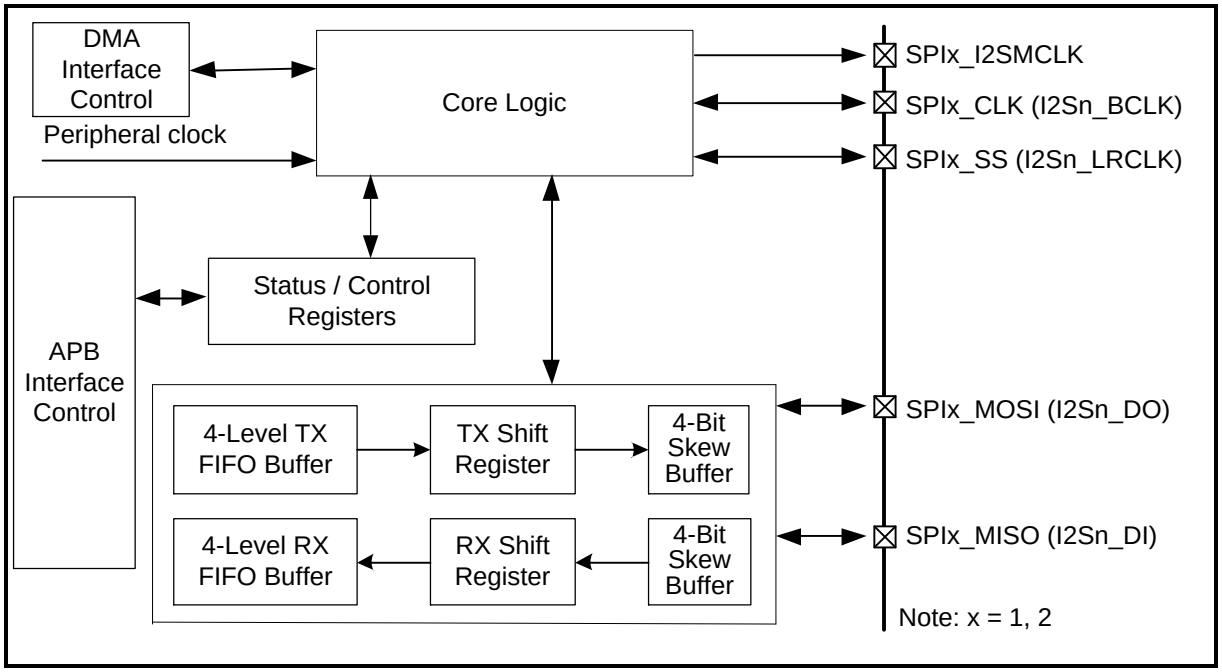


Figure 6.14-2 SPI Block Diagram (SPI1/2)

TX FIFO Buffer:

The transmit FIFO buffer is a 4-/8-level depth, 32-bit wide, first-in, first-out register buffer. The data can be written to the transmit FIFO buffer in advance through software by writing the SPIn_TX register. In SPI mode for SPI1~SPI2, The transmit FIFO will be configured as 8-level while data length is set as 8~16 bits.

RX FIFO Buffer:

The receive FIFO buffer is also a 4-/8-level depth, 32-bit wide, first-in, first-out register buffer. The receive control logic will store the receive data to this buffer. The FIFO buffer data can be read from SPIn_RX register by software. In SPI mode for SPI1~SPI2, The receive FIFO will be configured as 8-level while data length is set as 8~16 bits.

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. There are two skew buffers in transmitting and received side. In received side, it is used to shift bits into Rx shift register from SPI bus. In transmitting side, it is used to shift bits into SPI bus from Tx shift register.

6.14.4 Basic Configuration

6.14.4.1 SPI0 Basic Configurations

- Clock source configuration
  - Select the source of SPI0 peripheral clock on SPI0SEL (CLK_CLKSEL2[3:2]).
  - Enable SPI0 peripheral clock in SPI0CKEN (CLK_APBCLK0[12]).
- Reset configuration
  - Reset SPI0 controller in SPI0RST (SYS_IPRST1[12]).
- Pin configuration

Group	Pin Name	GPIO	MFP
SPI0	SPI0_CLK	PA.5	MFP1
		PC.12	MFP1
	SPI0_MISO0	PA.4	MFP1
		PC.11	MFP1
	SPI0_MISO1	PA.2	MFP1
		PC.9	MFP1
	SPI0_MOSI0	PA.3	MFP1
		PC.10	MFP1
	SPI0_MOSI1	PA.1	MFP1
		PC.8	MFP1
	SPI0_SS0	PA.6	MFP1
		PA.14	MFP1
		PC.7	MFP1
	SPI0_SS1	PA.0	MFP1
		PC.15	MFP1
		PD.13	MFP1

6.14.4.2 SPI1 Basic Configurations

- Clock source configuration
  - Select the source of SPI0 peripheral clock on SPI1SEL (CLK_CLKSEL2[5:4]).
  - Enable SPI0 peripheral clock in SPI1CKEN (CLK_APBCLK0[13]).
- Reset configuration
  - Reset SPI0 controller in SPI1RST (SYS_IPRST1[13]).
- Pin configuration

Group	Pin Name	GPIO	MFP
SPI1	SPI1_CLK	PC.2	MFP3

		PD.4	MFP2
	SPI1_MISO	PC.1	MFP3
		PD.3	MFP2
	SPI1_MOSI	PC.0	MFP3
		PD.2	MFP2
	SPI1_SS	PC.3	MFP3
		PD.5	MFP2
	SPI1_I2SMCLK	PC.4	MFP3
		PD.6	MFP2

6.14.4.3 SPI2 Basic Configurations

- Clock source configuration
  - Select the source of SPI0 peripheral clock on SPI2SEL (CLK_CLKSEL2[7:6]).
  - Enable SPI0 peripheral clock in SPI2CKEN (CLK_APBCLK0[14]).
- Reset configuration
  - Reset SPI0 controller in SPI2RST (SYS_IPRST1[14]).
- Pin configuration

Group	Pin Name	GPIO	MFP
SPI2	SPI2_CLK	PA.10	MFP4
		PC.7	MFP2
	SPI2_MISO	PA.7	MFP4
		PC.6	MFP2
	SPI2_MOSI	PA.8	MFP4
		PC.5	MFP2
	SPI2_SS	PA.9	MFP4
		PC.8	MFP2
	SPI2_I2SMCLK	PA.12	MFP4
		PC.9	MFP2

SPI/I2S (SPI1~SPI2) Interface Controller Pin description is shown as follows:

Pin	SPI Mode	I^2S Mode
SPIx_SS	SPI slave selection pin	I^2S left/right channel synchronization clock pin (I2Sx_LRCLK)
SPIx_CLK	SPI clock pin	I^2S bit clock pin (I2Sx_BCLK)
SPIx_MISO	SPI master input or slave output pin	I^2S data input pin (I2Sx_DI)

SPIx_MOSI	SPI master output or slave input pin	I^2S data output pin (I2Sx_DO)
SPIx_I2SMCLK	Not available	I^2S Master clock output pin

Note: When using the I2S function in SPI1 and SPI2, please enable schmitt trigger function (Px_SMTEN) on corresponding pins.

Table 6.14-3 SPI/I2S Interface Controller Pin Description (SPI1~SPI2)

6.14.5 Functional Description

6.14.5.1 Terminology

SPI Peripheral Clock and SPI Bus Clock

The SPI controller needs the peripheral clock to drive the SPI logic unit to perform the data transfer. The peripheral clock rate is determined by the settings of clock divider (SPIn_CLKDIV) and the clock source which can be HXT, PLL, PCLK or HIRC. SPInSEL of CLK_CLKSEL2 register determines the clock source of the peripheral clock. The DIVIDER (SPIn_CLKDIV[8:0]) setting determines the divisor of the clock rate calculation.

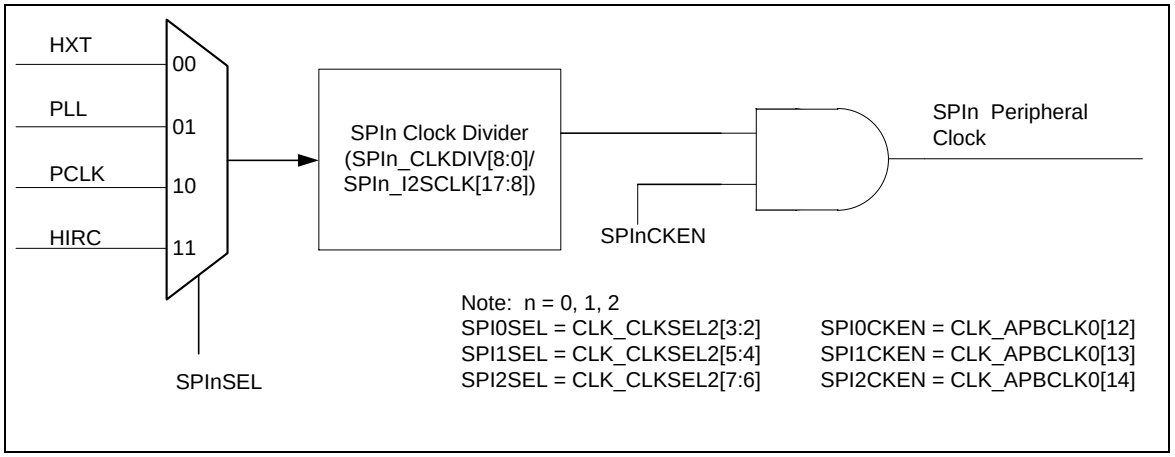


Figure 6.14-4 SPI Peripheral Clock

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^2S mode, the peripheral clock rate is equal to I^2S bit clock rate determined by SPIn_I2SCLK register.

Master/Slave mode

The SPI controllers can be set as Master or Slave mode by setting the SLAVE (SPIn_CTL[18]) to communicate with the off-chip SPI slave or master device. The HALFDPX (SPIn_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.

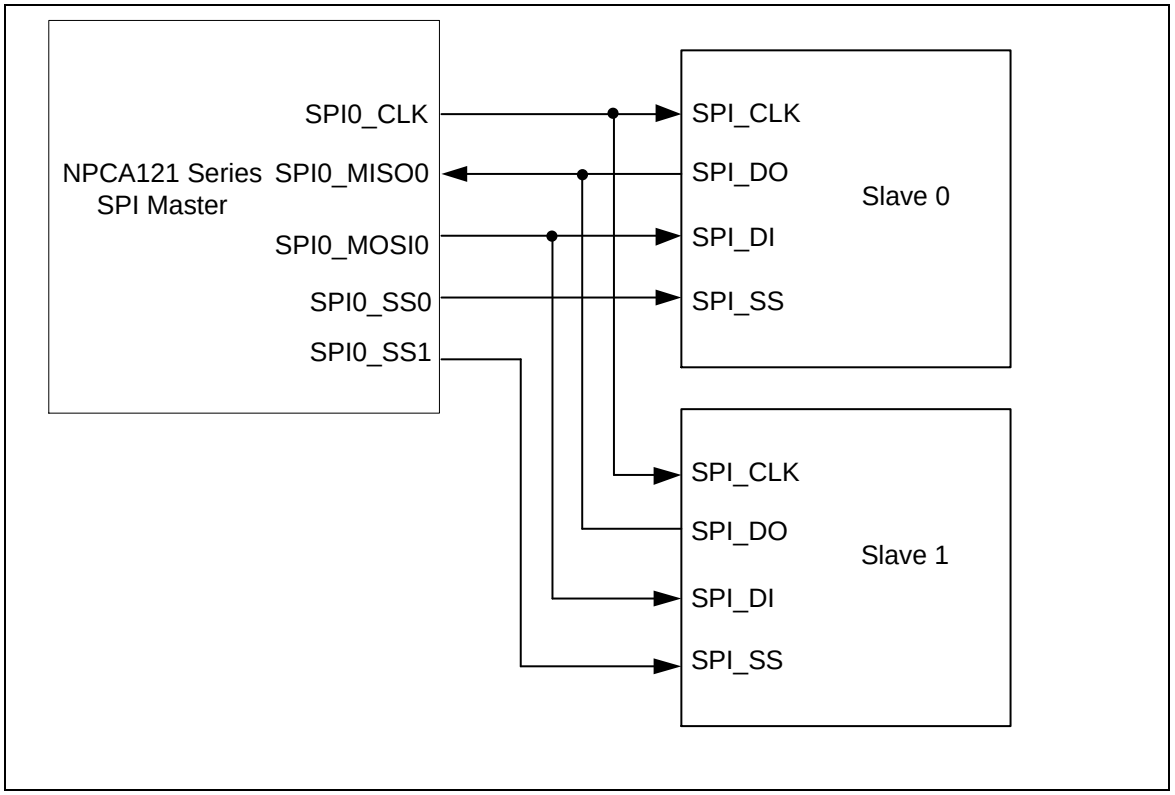


Figure 6.14-5 SPI0 Full-Duplex Master Mode Application Block Diagram

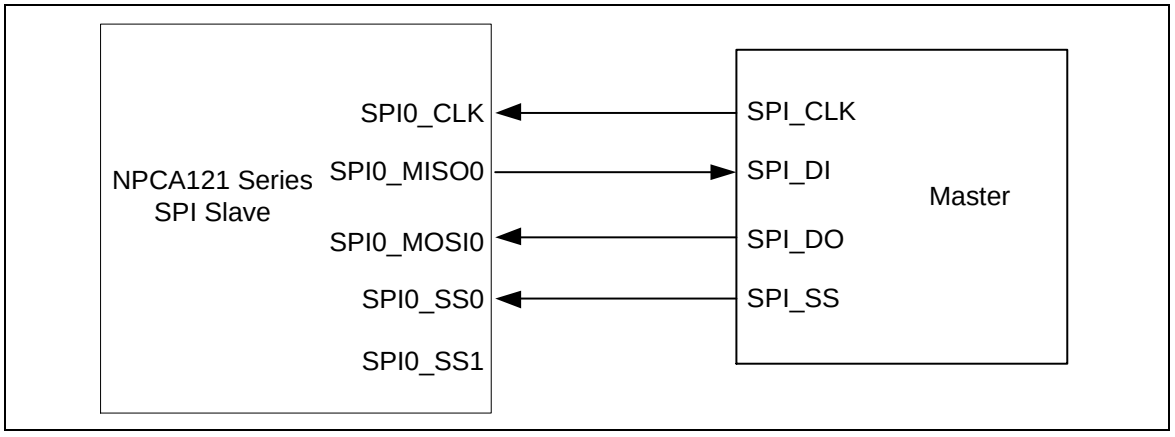


Figure 6.14-6 SPI0 Full-Duplex Slave Mode Application Block Diagram

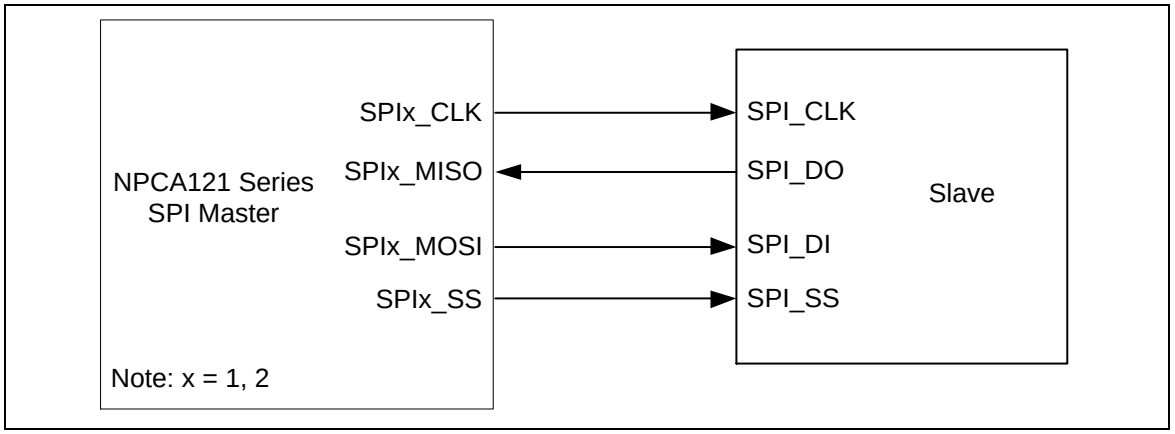


Figure 6.14-7 SPI1 ~ SPI2 Full-Duplex Master Mode Application Block Diagram

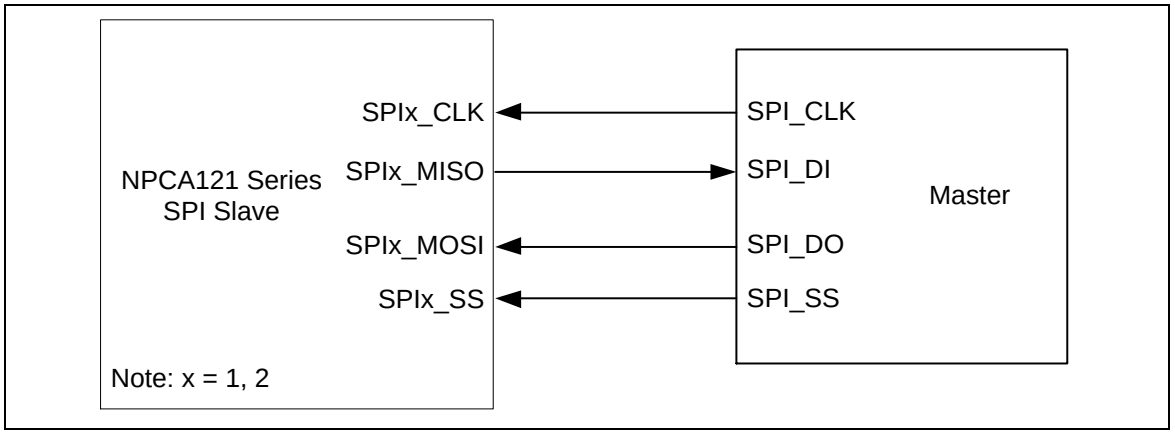


Figure 6.14-8 SPI1 ~ SPI2 Full-Duplex Slave Mode Application Block Diagram

Slave Selection

In Master mode, the SPI controller can drive off-chip slave device through the slave select output pin SPI0_SSy (y=0, 1) and SPIn_SS (n=1, 2). In Slave mode, the off-chip master device drives the slave selection signal from the SPI0_SS0 and SPIn_SS (n=1, 2) 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 (SPIn_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 (SPIn_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.

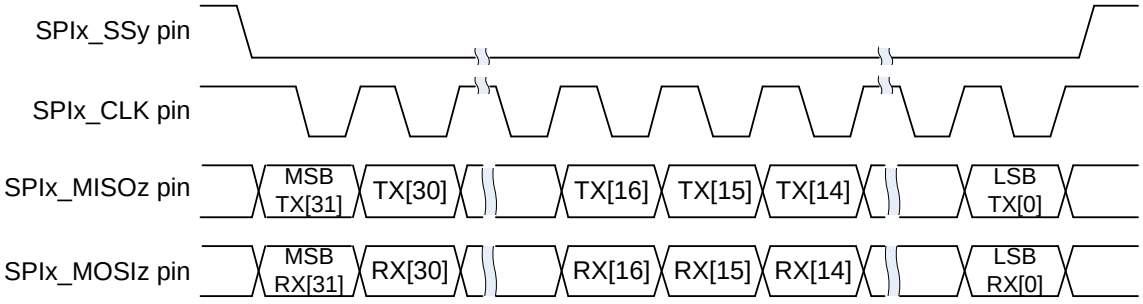
TXNEG (SPIn_CTL[2]) defines the data transmitted out either on negative edge or on positive edge of SPI clock. RXNEG (SPIn_CTL[1]) defines the data received either on negative edge or on positive edge of SPI clock.

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 (SPIn_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 (SPIn_CTL[12:8]), the unit transfer interrupt flag will be set to 1.



Note:
x: Controller number (x = 0, 1, 2), y: Slave selection pin channel number in SPI0 (y = 0, 1),
z: MOSI and MISO pin channel number in SPI0 (z = 0)

Figure 6.14-9 32-Bit in One Transaction

LSB/MSB First

LSB (SPIn_CTL[13]) defines the bit transfer sequence in a transaction. If the LSB (SPIn_CTL[13]) is set to 1, the transfer sequence is LSB first. The bit 0 will be transferred firstly. If the LSB (SPIn_CTL[13]) is cleared to 0, the transfer sequence is MSB first.

Suspend Interval

SUSPITV (SPIn_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).

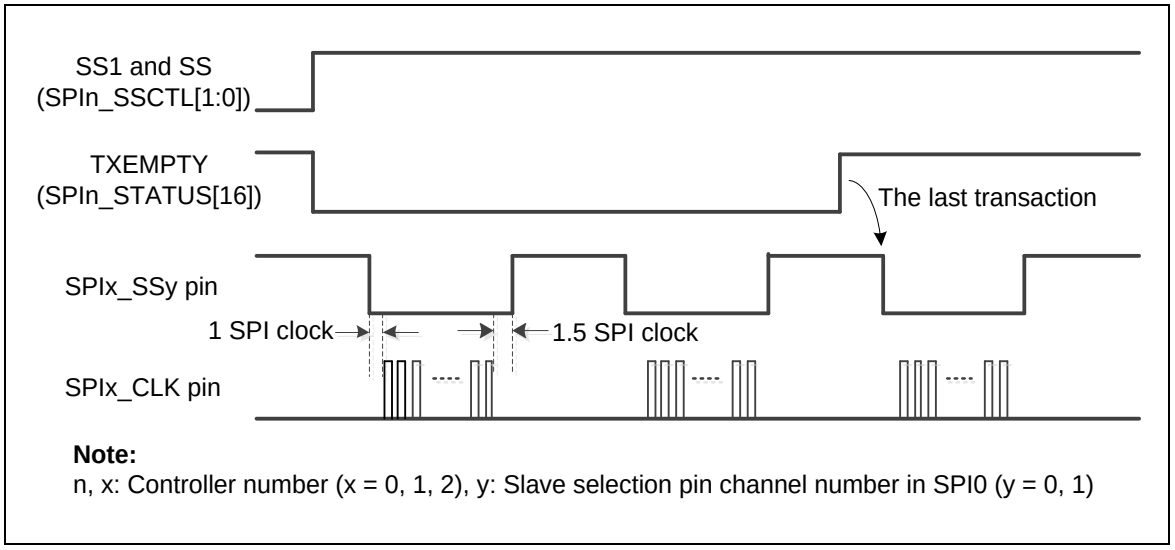
6.14.5.2 Automatic Slave Selection

In Master mode, if AUTOSS (SPIn_SSCTL[3]) is set, the slave selection signal will be generated automatically and output to the slave selection pin according to whether SS (SPIn_SSCTL[0]) and SS1 (SPI0_SSCTL[1]) 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 (SPIn_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.

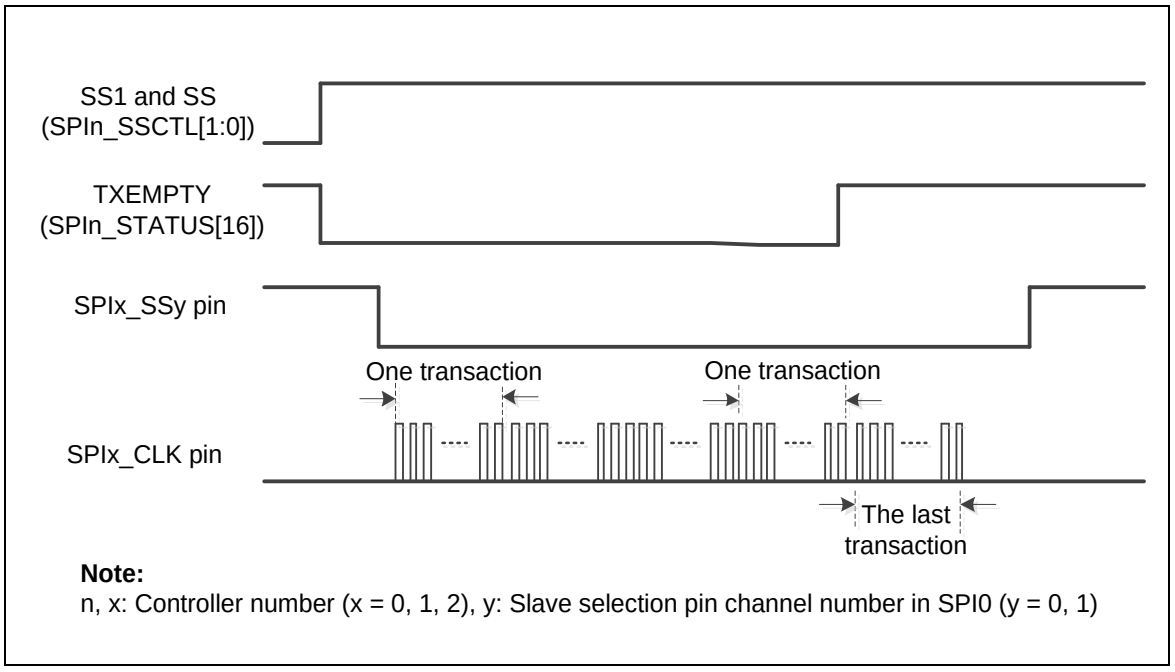
If the AUTOSS bit is cleared, the slave selection output signal will be determined by the SS setting. The active state of the slave selection output signal is specified in SSACTPOL (SPIn_SSCTL[2]).

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.



Note:
n, x: Controller number (x = 0, 1, 2), y: Slave selection pin channel number in SPI0 (y = 0, 1)

Figure 6.14-10 Automatic Slave Selection (SSACTPOL = 0, SUSPITV > 0x2)



Note:
n, x: Controller number (x = 0, 1, 2), y: Slave selection pin channel number in SPI0 (y = 0, 1)

Figure 6.14-11 Automatic Slave Selection (SSACTPOL = 0, SUSPITV < 0x3)

6.14.5.3 Byte Reorder and Suspend Function

When the transfer is set as MSB first (LSB = 0) and the REORDER (SPIn_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.

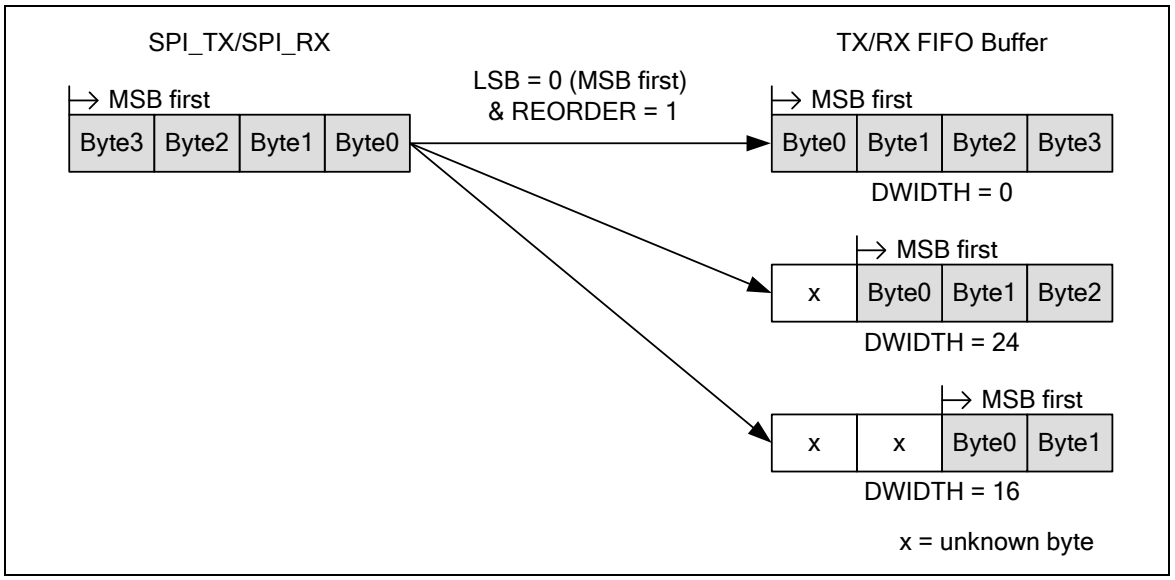
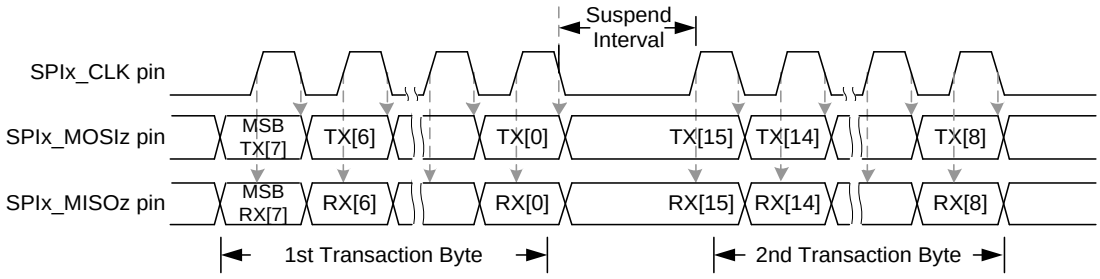


Figure 6.14-12 Byte Reorder Function

In Master mode, if REORDER (SPIn_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 (SPIn_CTL[7:4]).

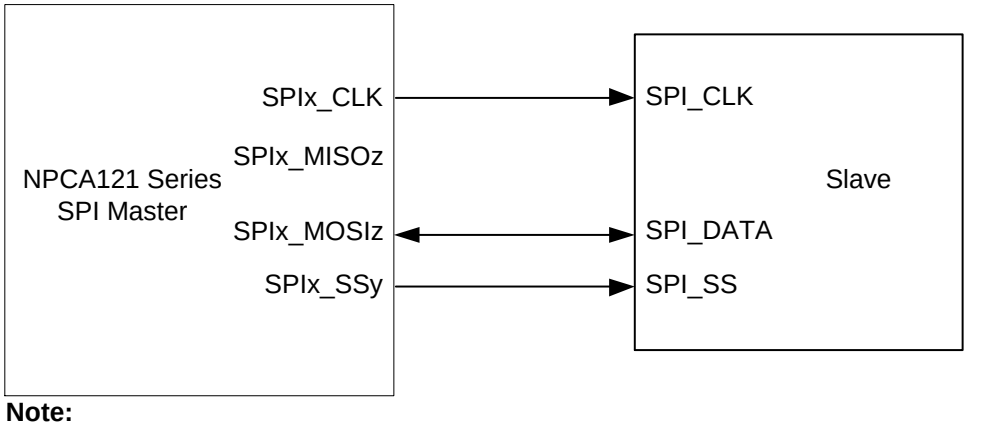


Note:
Timing Condition is CLKPOL = 0, LSB = 0, TXNEG = 1, DWIDTH = 0, REORDER = 1 and SLAVE = 0
x: Controller number (x = 0, 1, 2), z: MOSI and MISO pin channel number in SPI0 (z = 0, 1)

Figure 6.14-13 Timing Waveform for Byte Suspend

6.14.5.4 Half-Duplex Communication

The SPI controller can communicate in half-duplex mode by setting HALFDPX (SPIn_CTL[14]) bit. In half-duplex mode, there is only one data line for receiving or transmitting data direction which is defined by DATDIR (SPIn_CTL[20]). In half-duplex configuration, the SPI0_MISO0 and SPIx_MISO (x=1, 2) pin is free for other applications and it can be configured as GPIO. Enabling or disabling the control bit HALFDPX (SPIn_CTL[14]) will produce TXRST (SPIn_FIFOCTL[1]) and RXRST (SPIn_FIFOCTL[0]) at the same time automatically.



Note:
x: Controller number (x = 0, 1, 2), y: Slave selection pin channel number in SPI0 (y =
z: MOSI and MISO pin channel number in SPI0 (z = 0)

Figure 6.14-14 SPI Half-Duplex Master Mode Application Block Diagram

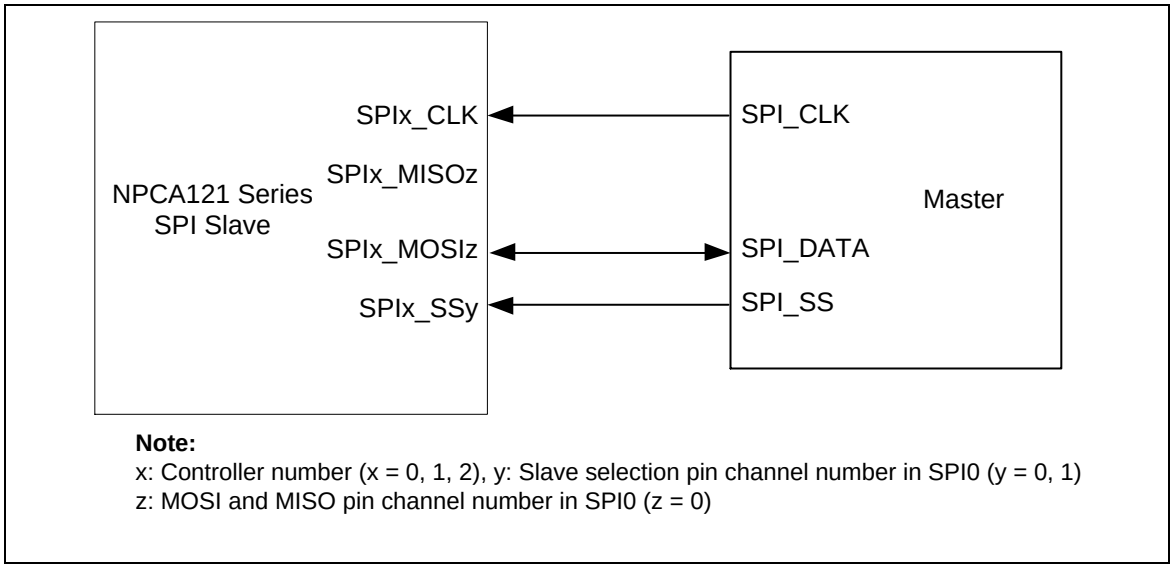


Figure 6.14-15 SPI Half-Duplex Slave Mode Application Block Diagram

6.14.5.5 Receive-Only Mode

In SPI Master device, it can communicate in receive-only mode by setting RXONLY (SPIn_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 (SPIn_SSCTL[3]) is enabled in receive-only mode, SPI Master will keep activating the slave select signal.

The remaining SPI0_MOSI0 and SPIx_MOSI (x=1, 2) pin of SPI Master device is not used for communication and can be configured as GPIO. The status BUSY (SPIn_STATUS[0]) will be asserted in receive-only mode due to the generation of SPI bus clock. Entering this mode will produce the TXRST (SPIn_FIFOCTL[1]) and RXRST (SPIn_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.

6.14.5.6 Slave 3-Wire Mode

When SLV3WIRE (SPI0_SSCTL[4]) is set by software to enable the Slave 3-Wire mode, the SPI controller can work with no slave selection signal in Slave mode. The SLV3WIRE (SPI0_SSCTL[4]) only takes effect in Slave mode. Only three pins, SPI0_CLK, SPI0_MISO0, and SPI0_MOSI0, are required to communicate with a SPI master. The SPI0_SS0 and SPI0_SS1 pin can be configured as a GPIO. When the SLV3WIRE (SPI0_SSCTL[4]) is set to 1, the SPI slave will be ready to transmit/receive data after the SPIEN (SPI0_CTL[0]) is set to 1.

Note: This function is only supported in SPI0.

6.14.5.7 PDMA Transfer Function

SPI controller supports PDMA transfer function.

When TXPDMAEN (SPIn_PDMACTL[0]) is set to 1, the controller will issue request to PDMA controller to start the PDMA transmission process automatically.

When RXPDMAEN (SPIn_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.

6.14.5.8 Two-Bit Transfer Mode

The SPI controller also supports 2-Bit Transfer mode when setting TWOBIT (SPI0_CTL[16]) to 1. In 2-Bit Transfer mode, the SPI controller performs full duplex data transfer. In other words, the two serial data bits can be transmitted and received simultaneously.

For example, in Master mode, the even data (TX Data (n)) stored in the SPI0_TX register will be transmitted through the SPI0_MOSI0 pin and the odd data (TX Data (n+1)) stored in the SPI0_TX register will be transmitted through the SPI0_MOSI1 pin respectively. In the meanwhile, the even data received from SPI0_MISO0 pin will be written to RX FIFO prior to the odd data received from SPI0_MISO1 pin.

In Slave mode, the even and odd data stored in the SPI0_TX register will be transmitted through the SPI0_MISO0 pin and SPI0_MISO1 pin respectively. In the meanwhile, the SPI0_RX register will store the even data received from the SPI0_MOSI0 pin and the odd data from SPI0_MOSI1 pin respectively. The data sequence of FIFO buffers is the same as the Master mode.

Note: This function is only supported in SPI0.

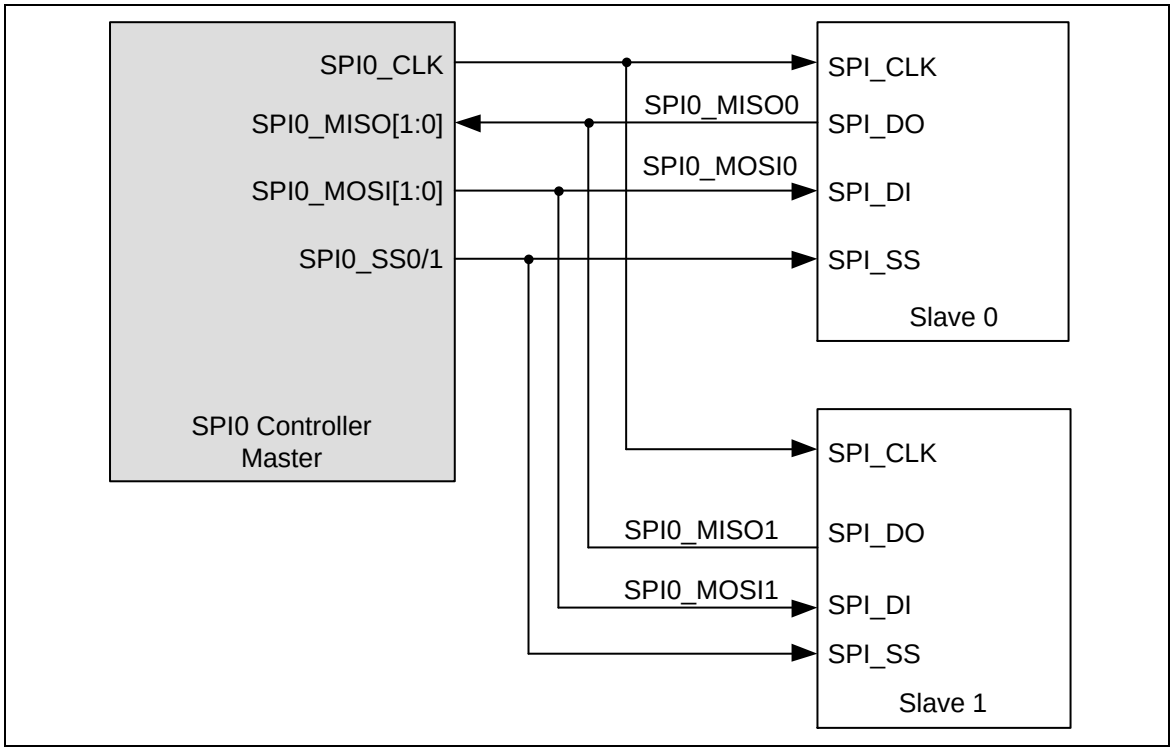


Figure 6.14-16 Two-Bit Transfer Mode System Architecture

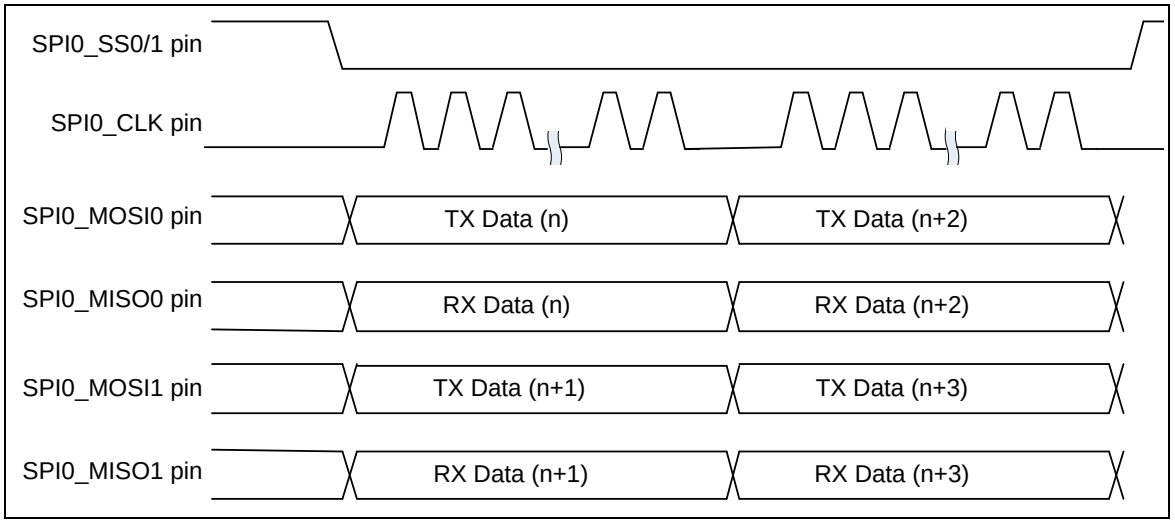


Figure 6.14-17 Two-Bit Transfer Mode Timing (Master Mode)

6.14.5.9 Dual I/O Mode

The SPI0 controller also supports Dual I/O transfer when setting the DUALIOEN (SPI0_CTL[21]) to 1. Many general SPI flashes support Dual I/O transfer. The DATDIR (SPI0_CTL[20]) is used to define the direction of the transfer data. When the DATDIR bit is set to 1, the controller will send the data to external device. When the DATDIR bit is set to 0, the controller will read the data from the external device. This function supports 8, 16, 24, and 32 bits of length.

The Dual I/O mode is not supported when the Slave 3-Wire mode or the Byte Reorder function is enabled.

For Dual I/O mode, if both the DUALIOEN (SPI0_CTL[21]) and DATDIR (SPI0_CTL[20]) are set as 1, the SPI0_MOSI0 is the even bit data output and the SPI0_MISO0 will be set as the odd bit data output. If the DUALIOEN (SPI0_CTL[21]) is set as 1 and DATDIR (SPI0_CTL[20]) is set as 0, both the SPI0_MISO0 and SPI0_MOSI0 will be set as data input ports.

Note: This function is only supported in SPI0.

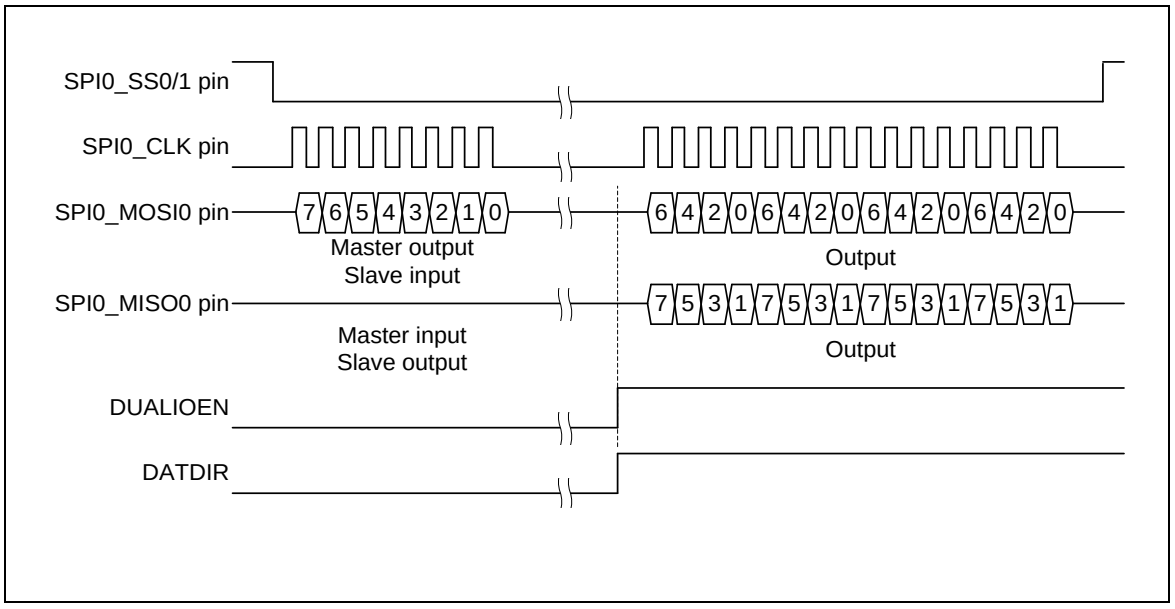


Figure 6.14-18 Bit Sequence of Dual Output Mode

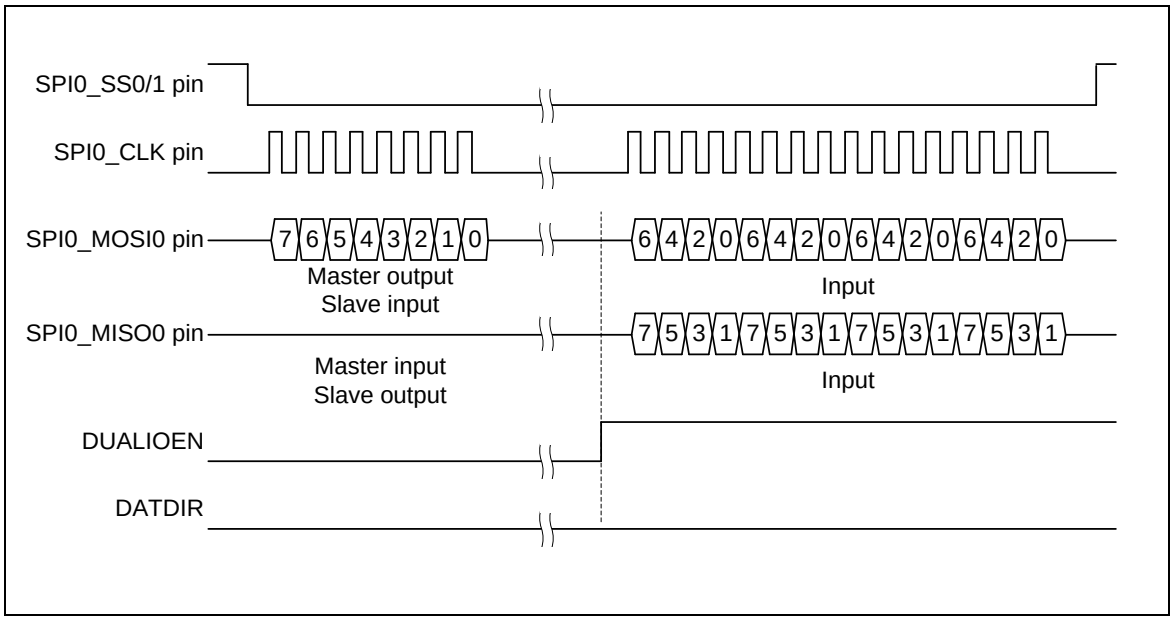


Figure 6.14-19 Bit Sequence of Dual Input Mode

6.14.5.10 Quad I/O Mode

The SPI0 controller also supports Quad I/O transfer when setting the QUADIOEN (SPI0_CTL[22]) to 1. Many general SPI flashes support Quad I/O transfer. The DATDIR bit (SPI0_CTL[20]) is used to define the direction of the transfer data. When the DATDIR (SPI0_CTL[20]) is set to 1, the controller will send the data to external device. When the DATDIR (SPI0_CTL[20]) is set to 0, the controller will read the data from the external device. This function supports 8, 16, 24, and 32 bits of length.

The Quad I/O mode is not supported when the Slave 3-Wire mode or the Byte Reorder function is enabled. The DUALIOEN (SPI0_CTL[21]) and QUADIOEN (SPI0_CTL[22]) shall not be set to 1 simultaneously.

For Quad I/O mode, if both the QUADIOEN (SPI0_CTL[22]) and DATDIR (SPI0_CTL[20]) are set as 1, the SPI0_MOSI0 and SPI0_MOSI1 are the even bit data output and the SPI0_MISO0 and SPI0_MISO1 will be set as the odd bit data output. If the QUADIOEN (SPI0_CTL[22]) is set as 1 and DATDIR (SPI0_CTL[20]) is set as 0, all the SPI0_MISO0, SPI0_MISO1, SPI0_MOSI0 and SPI0_MOSI1 pins will be set as data input ports.

Note: This function is only supported in SPI0.

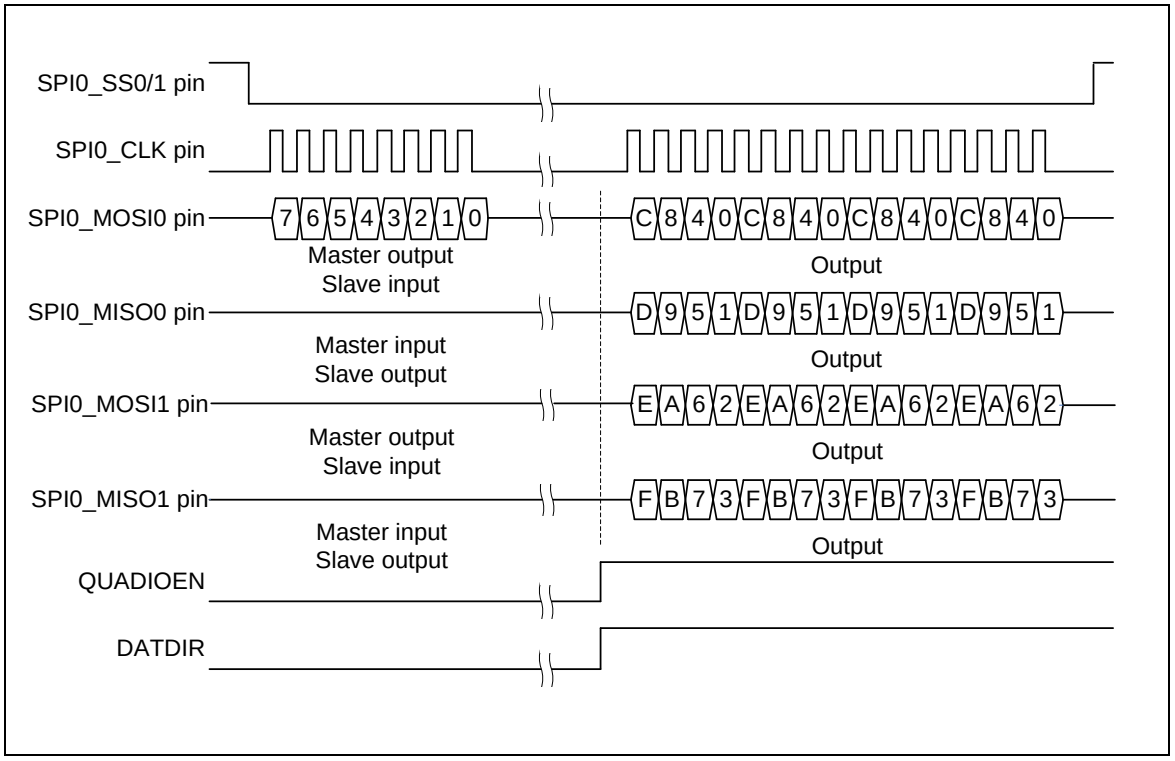


Figure 6.14-20 Bit Sequence of Quad Output Mode

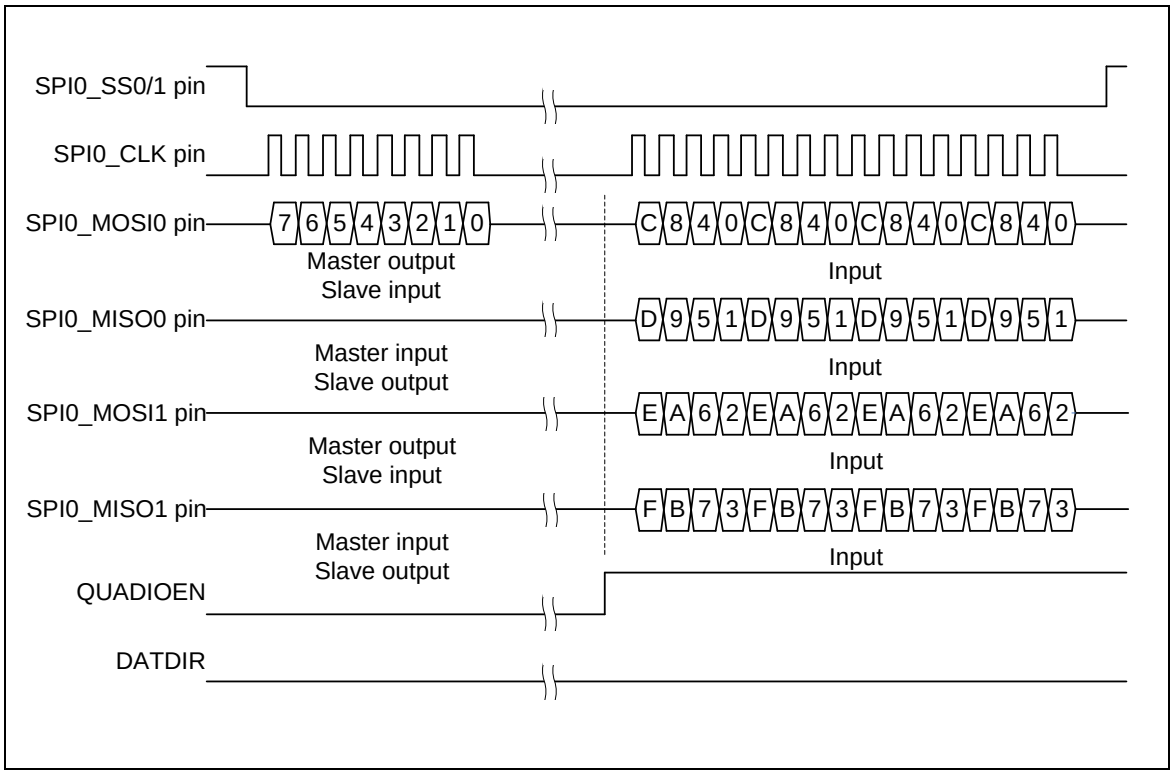


Figure 6.14-21 Bit Sequence of Quad Input Mode

6.14.5.11 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 (SPIn_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 (SPIn_STATUS[16]) will be set to 1. Notice that the TXEMPTY (SPIn_STATUS[16]) flag is set to 1 while the last transaction is still in progress. In Master mode, the BUSY (SPIn_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 (SPIn_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 (SPIn_STATUS[8]) and RXFULL (SPIn_STATUS[9]), to indicate the current status of RX FIFO buffer.

The transmitting and receiving threshold can be configured by setting TXTH (SPIn_FIFOCTL[30:28]) and RXTH (SPIn_FIFOCTL[26:24]). When the count of valid data stored in transmit FIFO buffer is less than or equal to TXTH (SPIn_FIFOCTL[30:28]) setting, TXTHIF (SPIn_STATUS[18]) will be set to 1. When the count of valid data stored in receive FIFO buffer is larger than RXTH (SPIn_FIFOCTL[26:24]) setting, RXTHIF (SPIn_STATUS[10]) will be set to 1.

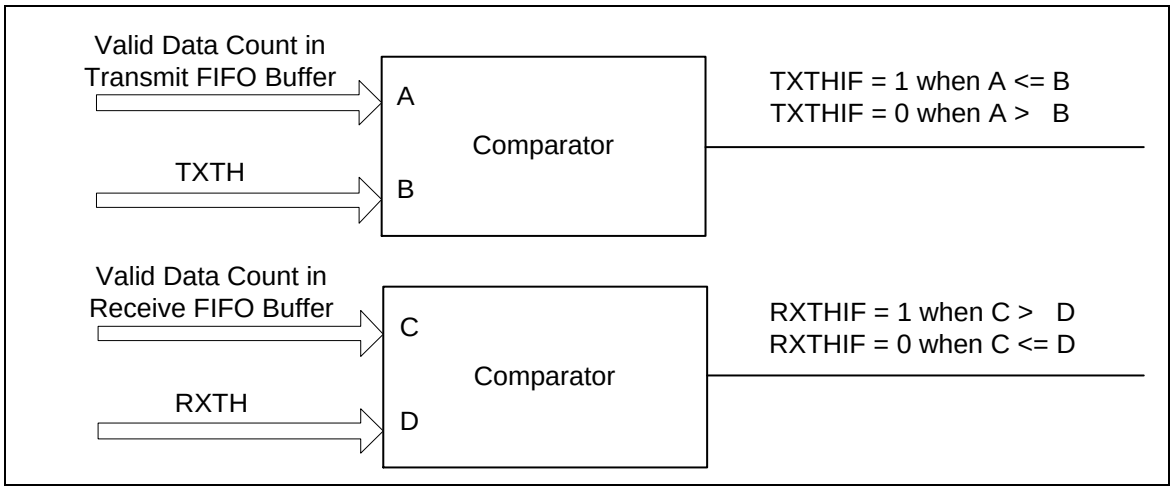


Figure 6.14-22 FIFO Threshold Comparator

In Master mode, when the first datum is written to the SPIn_TX register, the TXEMPTY flag (SPIn_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 SPIn_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 (SPIn_CTL[7:4]). If the SUSPITV (SPIn_CTL[7:4]) equals 0, SPI controller can perform continuous transfer. User can write data into SPIn_TX register as long as the TXFULL (SPIn_STATUS[17]) is 0.

In the example 1 of the following Figure, it indicates the updated condition of TXEMPTY (SPIn_STATUS[16]) and the relationship among the FIFO buffer, shift register and the skew buffer. The TXEMPTY (SPIn_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 the core logic and the TXEMPTY (SPIn_STATUS[16]) will be to 1. The Data0 in shift register will be shift into skew buffer by bit for transmission until the transfer is done.

In the Example 2, it indicates the updated condition of TXFULL (SPIn_STATUS[17]) when there are 8 data in the FIFO buffer and the next data of Data9 does not be written into the FIFO buffer when the TXFULL = 1.

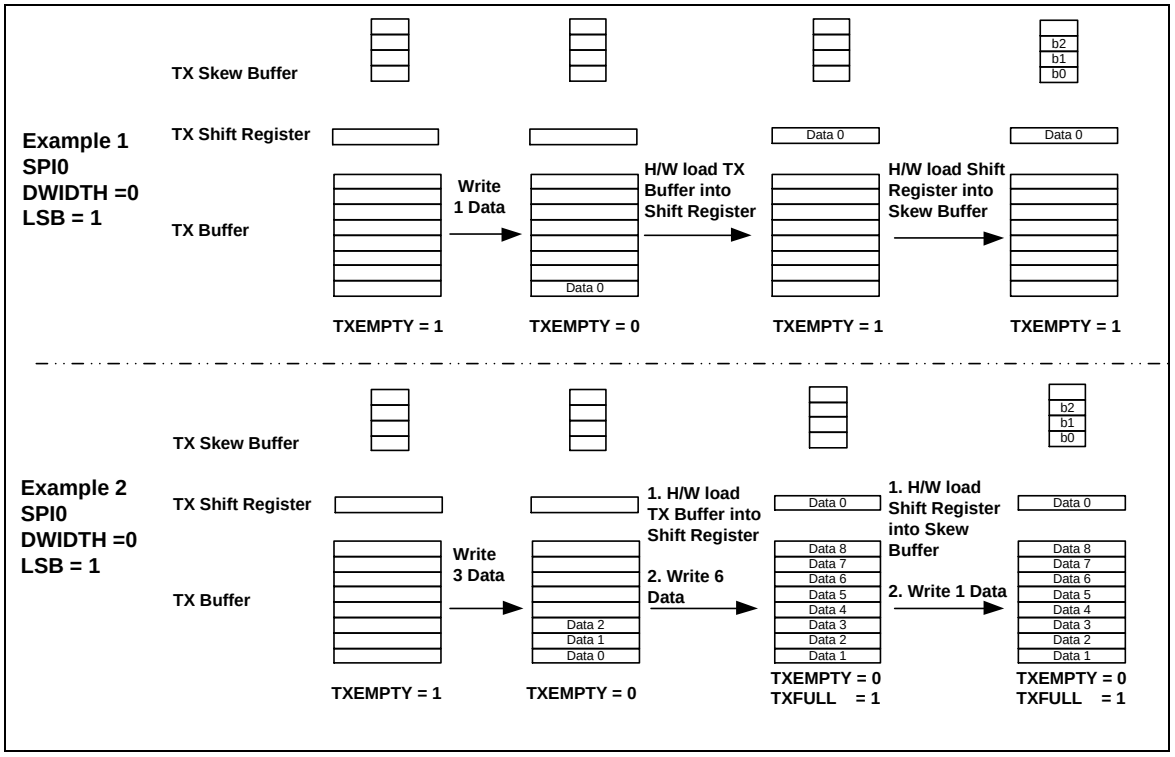


Figure 6.14-23 Transmit FIFO Buffer Example

The subsequent transactions will be triggered automatically if the transmitted data are updated in time. If the SPIn_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 SPI0_MISO0 and SPIx_MISO (x=1, 2) pin and stored to receive FIFO buffer.

The received data (Data0's b0, b1, …b31) is stored into skew buffer first according the serial clock (SPIn_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 received data bit count reach the value of DWIDTH (SPIn_CTL[12:8]). The RXEMPTY (SPIn_STATUS[8]) will be cleared to 0 while the receive FIFO buffer contains unread data (see the Example 1 of Receive FIFO Buffer Example). The received data can be read by software from SPIn_RX register as long as the RXEMPTY (SPIn_STATUS[8]) is 0. If the receive FIFO buffer contains 8 unread data, the RXFULL (SPIn_STATUS[9]) will be set to 1 (see the Example 2 of Receive FIFO Buffer Example).

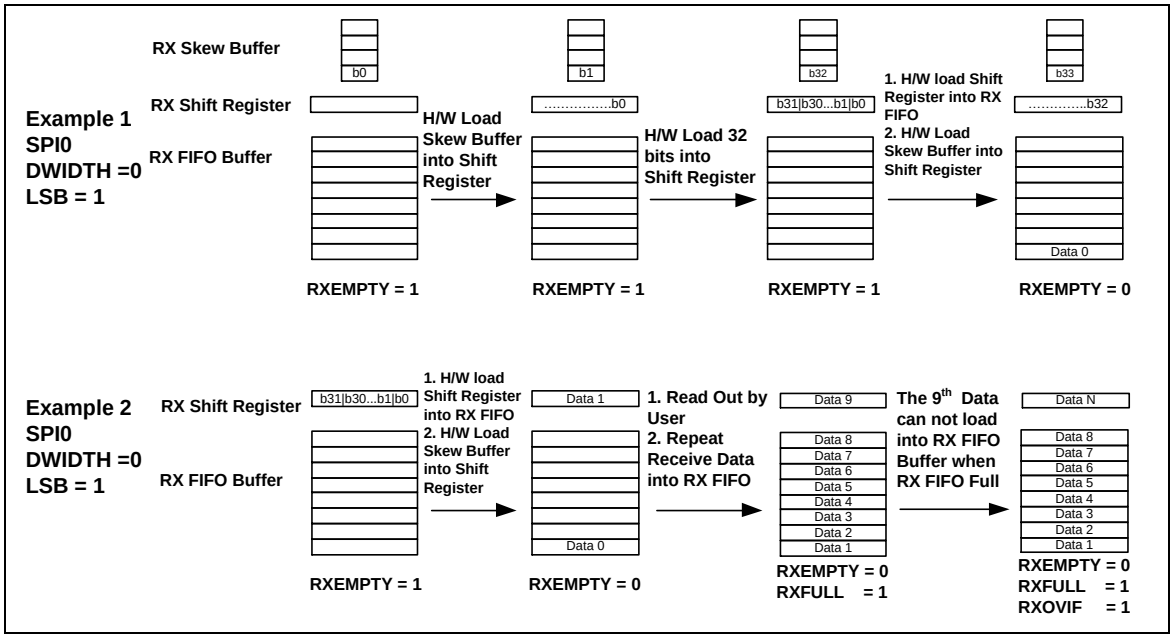
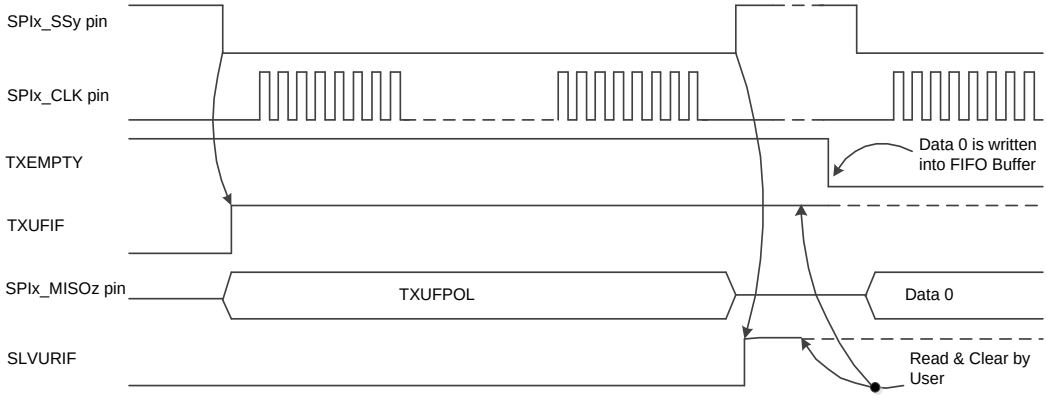


Figure 6.14-24 Receive FIFO Buffer Example

In Slave mode, during transmission operation, when data is written to the SPIn_TX register by software, the data will be loaded into transmit FIFO buffer and the TXEMPTY (SPIn_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 SPIn_TX register as long as the TXFULL (SPIn_STATUS[17]) is 0. After all data have been drawn out by the SPI transmission logic unit and the SPIn_TX register is not updated by software, the TXEMPTY (SPIn_STATUS[16]) will be set to 1.

If there is no any data written to the SPIn_TX register, the transmit underflow interrupt flag, TXUFIF (SPIn_STATUS[19]) will be set to 1 when the slave selection signal is active. The output data will be held by TXUFPOL (SPIn_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 interrupt flag, SLVURIF (SPIn_STATUS[7]), will be set to 1 as slave selection pin goes to inactive state.



Note:
x: Controller number (x = 0, 1, 2), y: Slave selection pin channel number in SPI0 (y = 0, 1),
z: MOSI and MISO pin channel number in SPI0 (z = 0)

Figure 6.14-25 TX Underflow Event and Slave Under Run Event

In 2-Bit Transfer mode, the transmit data is loaded into shift register after 2 datum have been written into the TX FIFO buffer. It uses two shift registers and two 4-level skew buffers concurrently. The detail timing of 2-Bit Transfer mode, please refer to the section of Two-Bit Transfer mode.

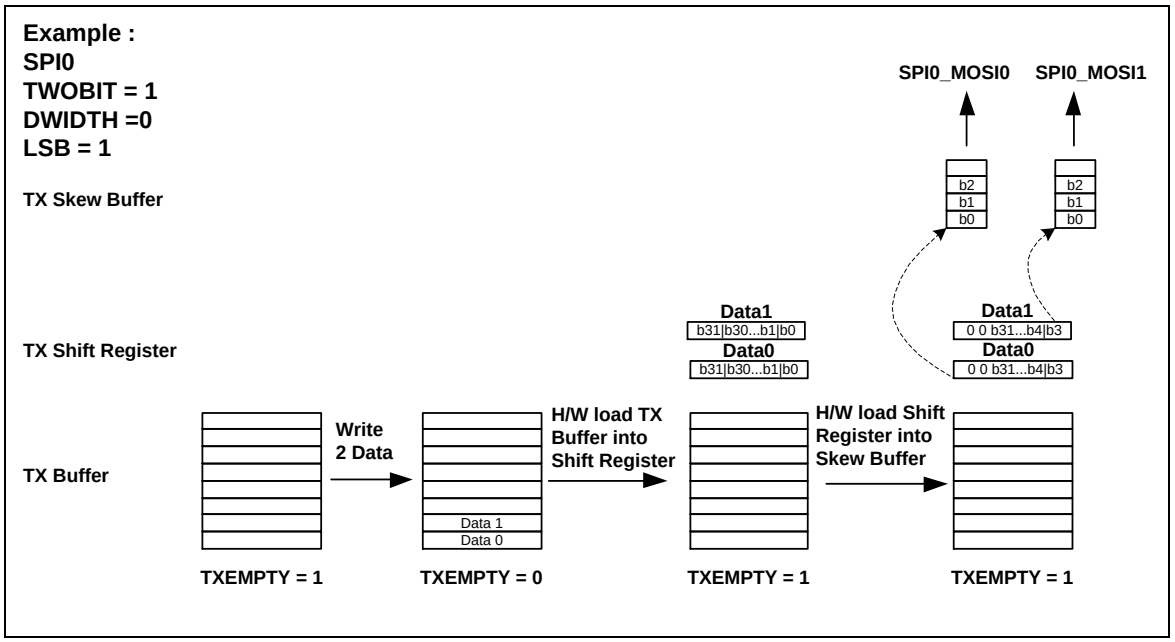


Figure 6.14-26 Two-Bit Transfer Mode FIFO Buffer Example (SPI0 Only)

In SPI0 Slave 3-Wire mode, the first 2-bit data is un-predicted (keep on the level of last bit in previously transfer) if the data is written into TX FIFO among 3 peripheral clock cycles before the SPI bus clock is presented. The other bits are held by TXUFPOL (SPI0_FIFOCTL[6]) because there is TX underflow event. The written data will be transmitted in the next transfer.

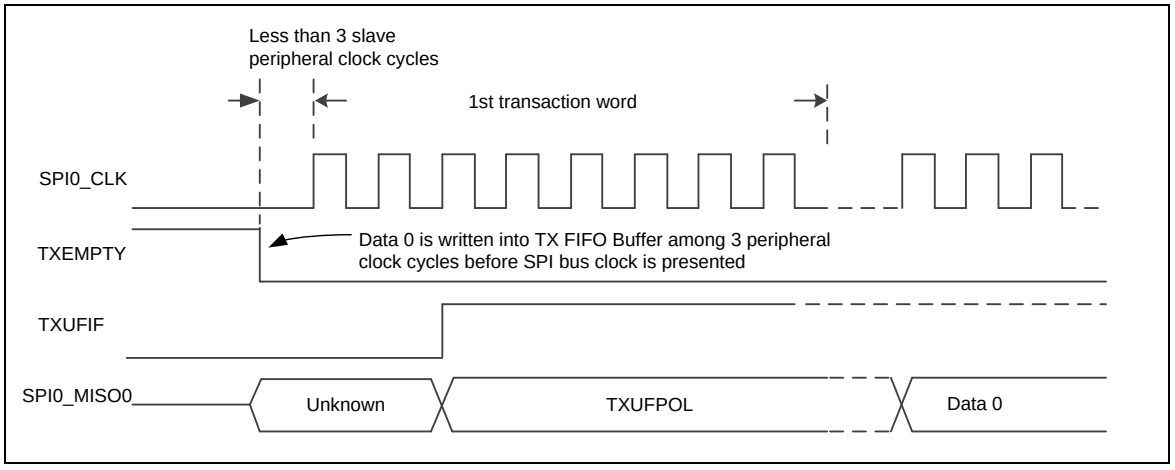


Figure 6.14-27 TX Underflow Event (SPI0 Slave 3-Wire Mode Enabled)

In Slave mode, during receiving operation, the serial data is received from SPI0_MOSI0 and SPIx_MOSI (x=1, 2) pin and stored to SPIn_RX register. The reception mechanism is similar to

Master mode reception operation. If the receive FIFO buffer contains 4 (or 8 for SPI0) unread data, the RXFULL (SPIn_STATUS[9]) will be set to 1 and the RXOVIF (SPIn_STATUS[11]) will be set to 1 if there is more serial data received from SPI0_MOSI0 and SPIx_MOSI (x=1, 2) pin and follow-up data will be dropped (refer to the Receive FIFO Buffer Example figure). If the receive bit count mismatch with the DWIDTH (SPIn_CTL[12:8]) when the slave selection line goes to inactive state, the SLVBEIF (SPIn_STATUS[6]) will be set to 1.

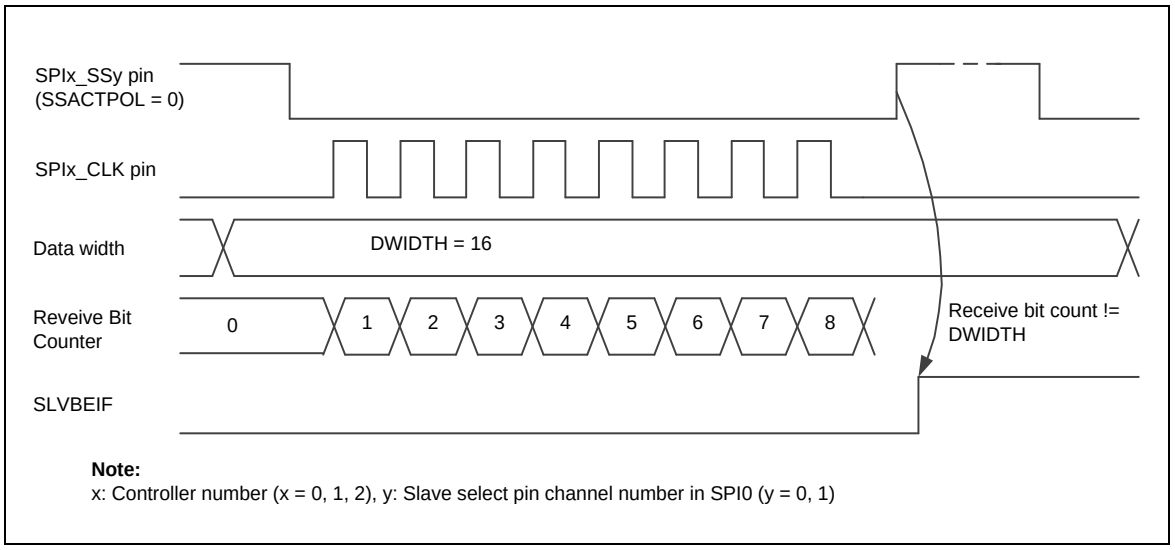


Figure 6.14-28 Slave Mode Bit Count Error

When the Slave select signal is active and the value of SLVTOCNT (SPI0_SSCTL[31:16]) is not 0, the Slave time-out counter in the SPI controller logic will start after the serial clock input. This counter will be cleared after one transaction done or the SLVTOCNT is set to 0. If the value of the time-out counter is greater than or equal to the value of SLVTOCNT before one transaction done, the slave time-out event occurs and the SLVTOIF (SPI0_STATUS[5]) will be set to 1. The Slave time-out function is only available for SPI0.

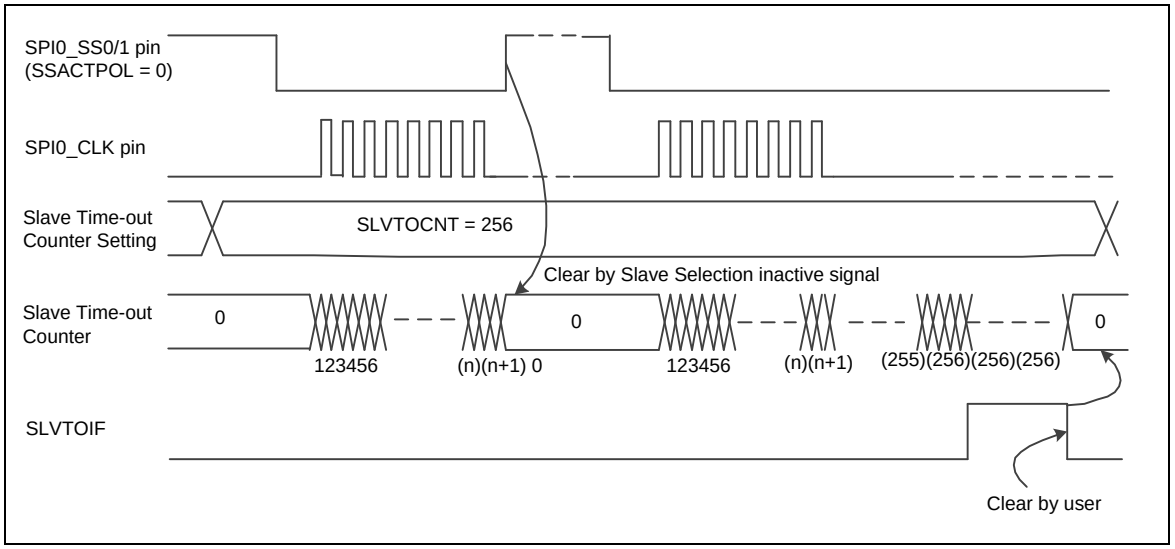


Figure 6.14-29 Slave Time-out Event (for SPI0)

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 peripheral clock periods in Master mode or over 576 SPI peripheral clock periods in Slave mode, the receive time-out occurs and the RXTOIF (SPIn_STATUS[12]) will be set to 1. When the receive FIFO is read by user, the time-out status will be cleared automatically.

6.14.5.12 Interrupt

- SPI unit transfer interrupt

As the SPI controller finishes a unit transfer, the unit transfer interrupt flag UNITIF (SPIn_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 (SPIn_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 (SPIn_STATUS[2]) and SSINAIF (SPIn_STATUS[3]), will be set to 1 when the SPIEN (SPIn_CTL[0]) and SLAVE (SPIn_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 (SPIn_SSCTL[13]) or SSACTIEN (SPIn_SSCTL[12]), are set to 1.

- Slave time-out interrupt

In SPI0 Slave mode, there is slave time-out function for user to know that there is serial clock input but one transaction is not finished over the period of SLVTOCNT (SPI0_SSCTL[31:16]) basing on Slave peripheral clock.

When the slave selection signal is active and the value of SLVTOCNT (SPI0_SSCTL[31:16]) is not 0, the slave time-out counter in the SPI0 controller logic will start after the serial clock input. This counter will be cleared after one transaction done or the SLVTOCNT (SPI0_SSCTL[31:16]) is set to 0. If the value of the time-out counter is greater than or equal to the value of SLVTOCNT (SPI0_SSCTL[31:16]) before one transaction done, the slave time-out event occurs and the SLVTOIF (SPI0_STATUS[5]) will be set to 1. The SPI controller will issue an interrupt if the SLVTOIEN (SPI0_SSCTL[5]) is set to 1.

- Slave bit count error interrupt

In Slave mode, if the transmit/receive bit count mismatch with the DWIDTH (SPIn_CTL[12:8]) when the slave selection line goes to inactive state, the SLVBEIF (SPIn_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 (SPIn_SSCTL[8]) is set to 1.

Note: If the slave selection signal is active but there is no any serial clock input, the SLVBEIF (SPIn_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 is written to the SPIn_TX register, the TXUFIF (SPIn_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 (SPIn_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 (SPIn_STATUS[7]) will be set to 1 when slave selection pin goes to inactive state. The SPI controller will issue a TX under run interrupt if the SLVURIEN (SPIn_SSCTL[9]) is set to 1.

Note: In SPI0 Slave 3-Wire mode, the slave selection signal is considered active all the time so that user shall poll the TXUFIF (SPI0_STATUS[19]) to know if there is TX underflow event or not.

- Receive Overrun interrupt

In Slave mode, if the receive FIFO buffer contains 4 (or 8 for SPI0) unread data, the RXFULL (SPIn_STATUS[9]) will be set to 1 and the RXOVIF (SPIn_STATUS[11]) will be set to 1 if there is more serial data is received from SPI bus and follow-up data will be dropped. The SPI controller will issue an interrupt if the RXOVIEN (SPIn_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 (SPIn_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 (SPIn_FIFOCTL[30:28]), the transmit FIFO interrupt flag TXTHIF (SPIn_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 (SPIn_FIFOCTL[3]), is set to 1.

- Receive FIFO interrupt

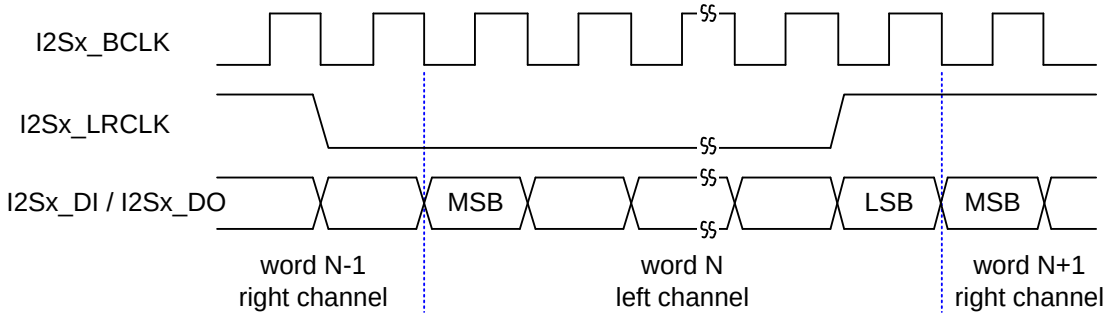
In FIFO mode, if the valid data count of the receive FIFO buffer is larger than the setting value of RXTH (SPIn_FIFOCTL[26:24]), the receive FIFO interrupt flag RXTHIF (SPIn_STATUS[10]) will be set to 1. The SPI controller will generate a receive FIFO interrupt to the system if the receive FIFO interrupt enable bit, RXTHIEN (SPIn_FIFOCTL[2]), is set to 1.

6.14.5.13 I^2S Mode

The SPI1~SPI2 controllers support I^2S mode with PCM mode A, PCM mode B, MSB justified and I^2S data format. The bit count of an audio channel is determined by WDWIDTH (SPIn_I2SCTL[5:4]). The transfer sequence is always first from the most significant bit, MSB. Data are read on rising clock edge and are driven on falling clock edge.

In I^2S data format, the MSB is sent and latched on the second clock of an audio channel. The I2Sx_LRCLK signal indicates which audio channel is in transferring.

Note that when using the I2S function in SPI1 and SPI2, please enable schmitt trigger function (Px_SMTEN) on corresponding pins.



Note:
x: Controller number (x = 1, 2)

Figure 6.14-30 I²S Data Format Timing Diagram

In MSB justified data format, the MSB is sent and latched on the first clock of an audio channel.

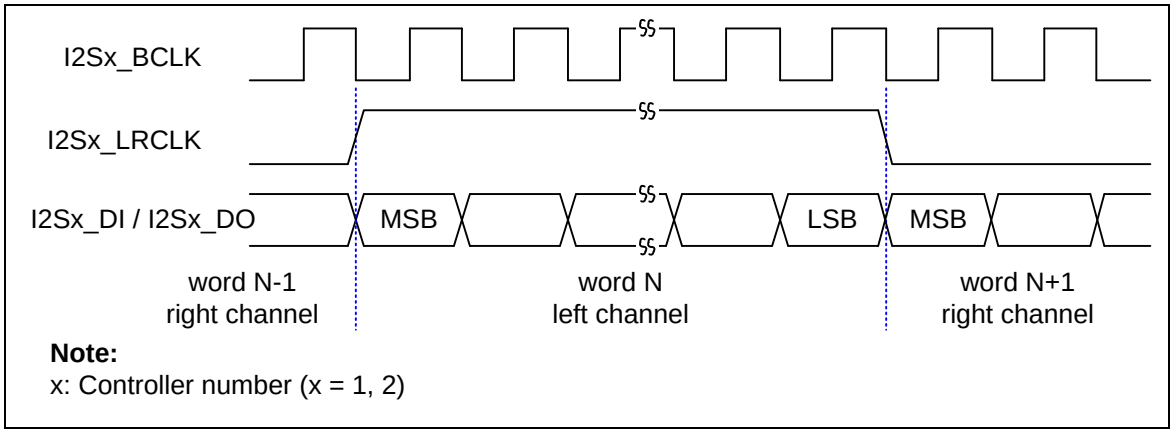


Figure 6.14-31 MSB Justified Data Format Timing Diagram

The I2Sx_LRCLK signal also supports PCM mode A and PCM mode B. The I2Sx_LRCLK signal in PCM mode indicates the beginning of an audio frame.

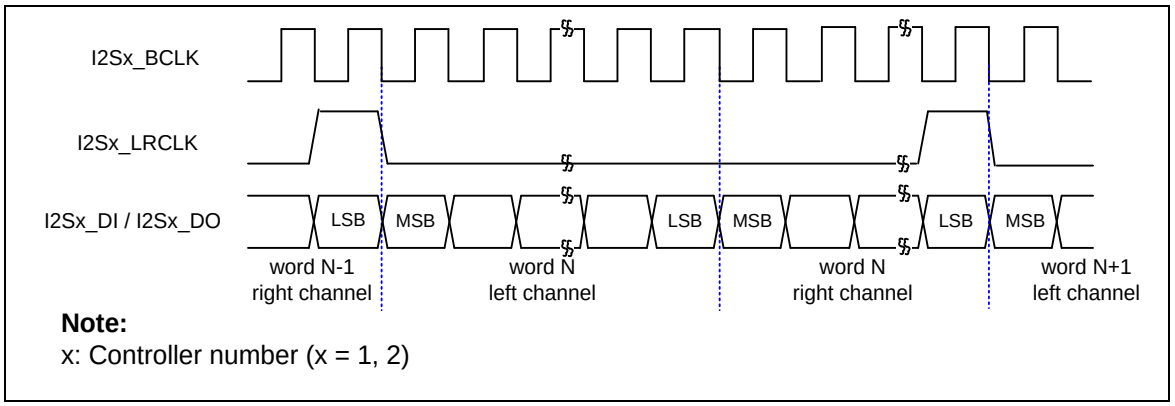


Figure 6.14-32 PCM Mode A Timing Diagram

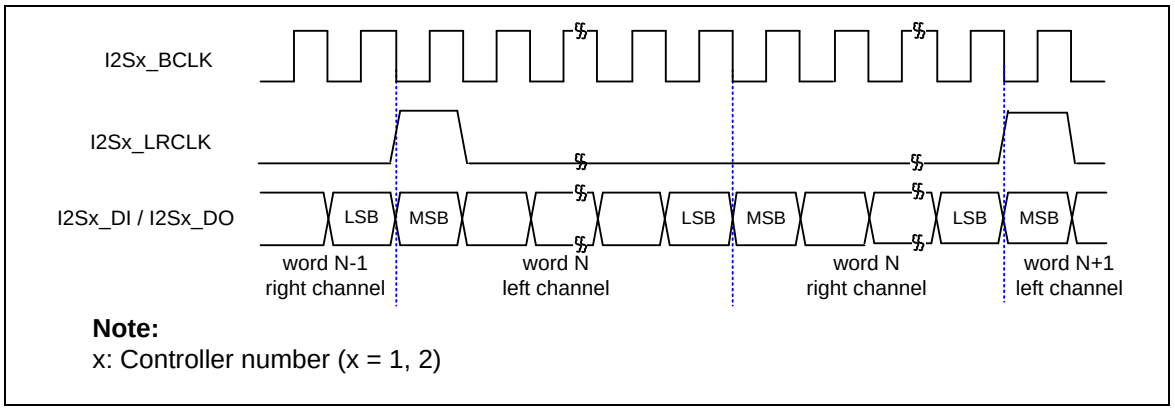


Figure 6.14-33 PCM Mode B Timing Diagram

6.14.5.14 I²S Mode FIFO operation

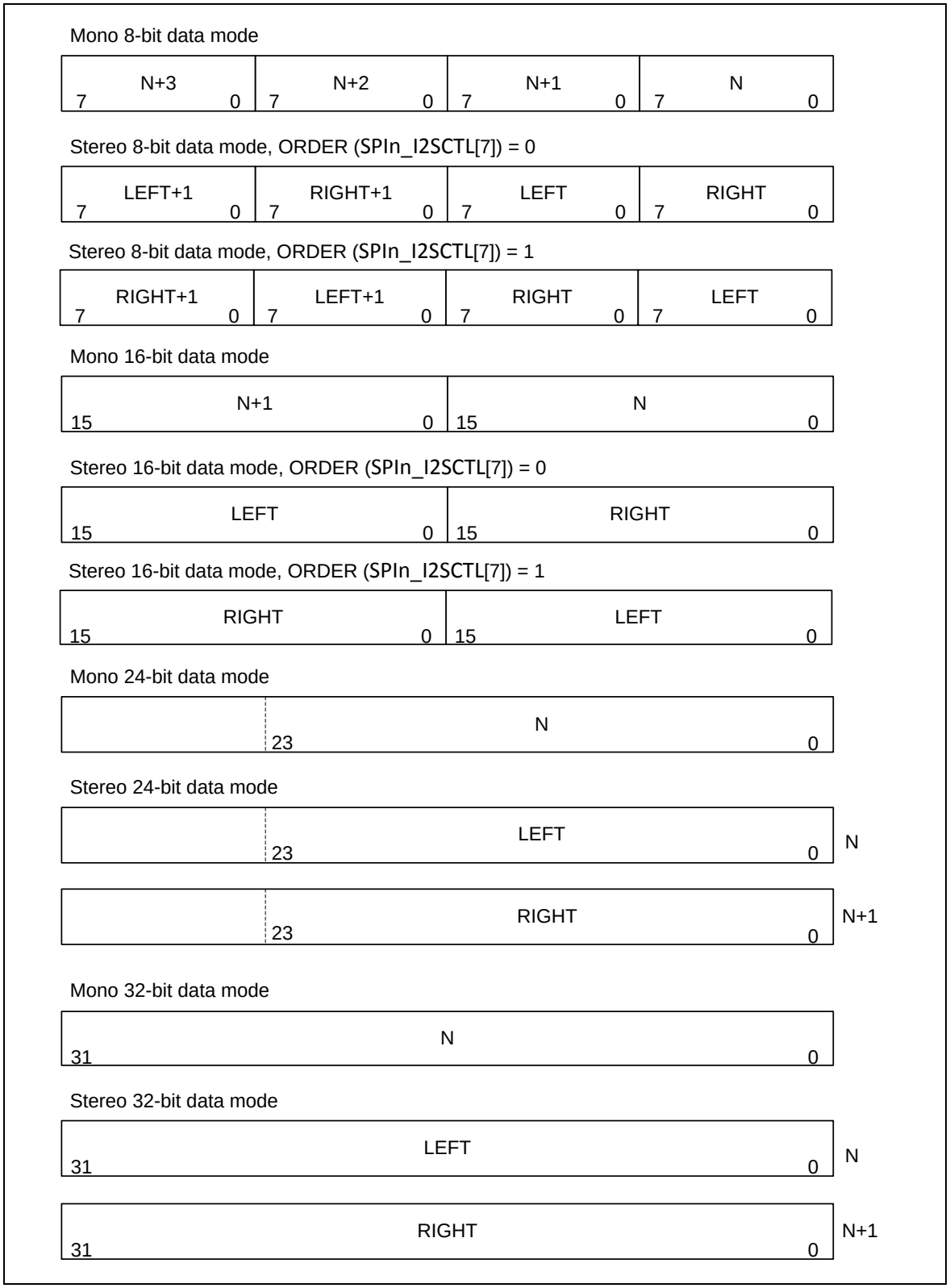
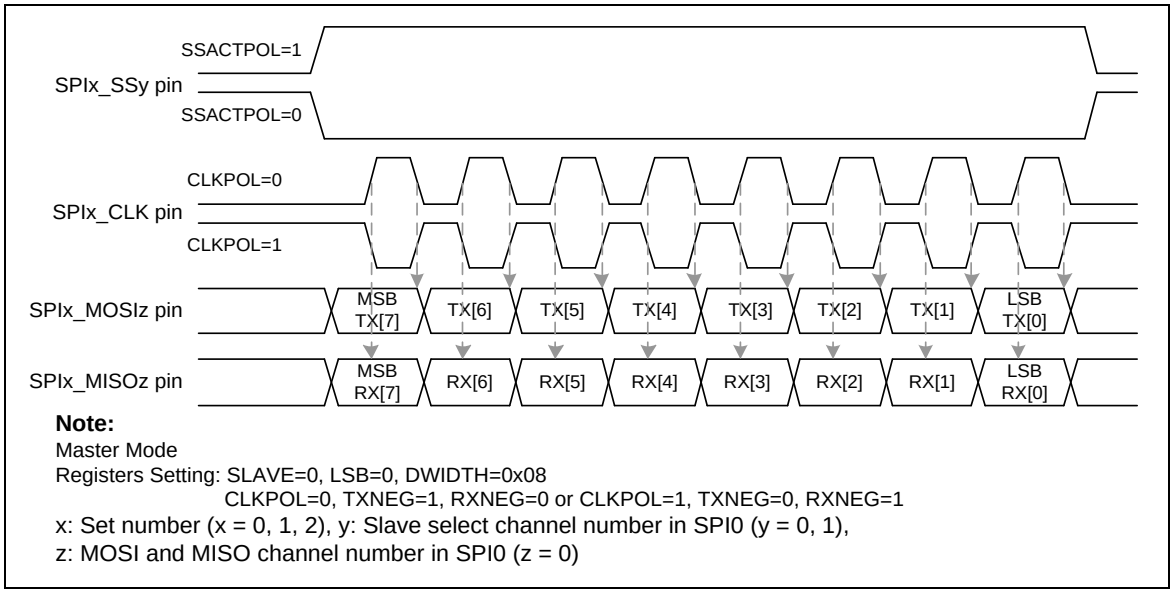


Figure 6.14-34 FIFO Contents for Various I²S Modes

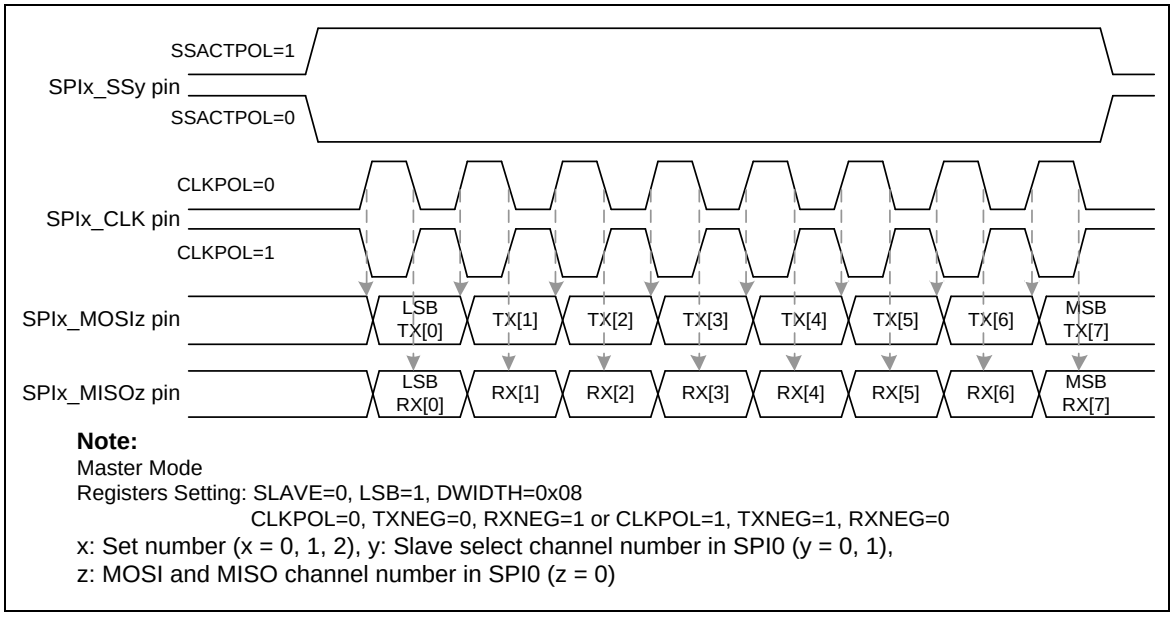
6.14.6 Timing Diagram

The active state of slave selection signal can be defined by setting the SSACTPOL (SPIn_SSCTL[2]). The SPI clock which is in idle state can be configured as high or low state by setting the CLKPOL (SPIn_CTL[3]). It also provides the bit length of a transaction word in DWIDTH (SPIn_CTL[12:8]), and transmitting/receiving data from MSB or LSB first in LSB (SPIn_CTL[13]). User can also select which edge of SPI clock to transmit/receive data in TXNEG/RXNEG (SPIn_CTL[2:1]). Four SPI timing diagrams for master/slave operations and the related settings are shown below.



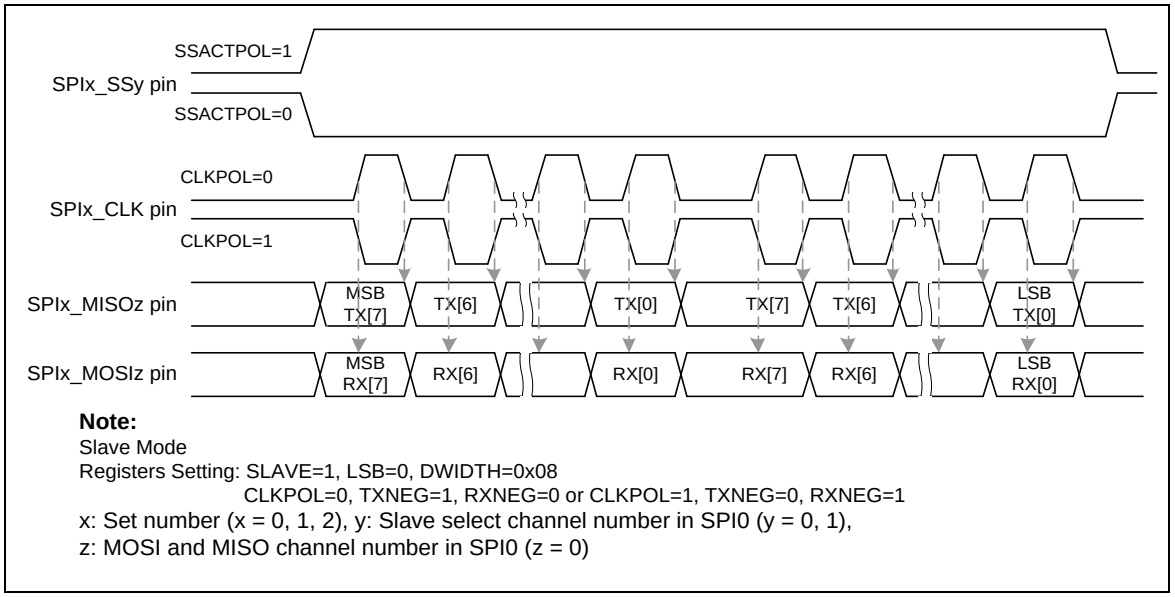
Note:
Master Mode
Registers Setting: SLAVE=0, LSB=0, DWIDTH=0x08
CLKPOL=0, TXNEG=1, RXNEG=0 or CLKPOL=1, TXNEG=0, RXNEG=1
x: Set number (x = 0, 1, 2), y: Slave select channel number in SPI0 (y = 0, 1),
z: MOSI and MISO channel number in SPI0 (z = 0)

Figure 6.14-35 SPI Timing in Master Mode



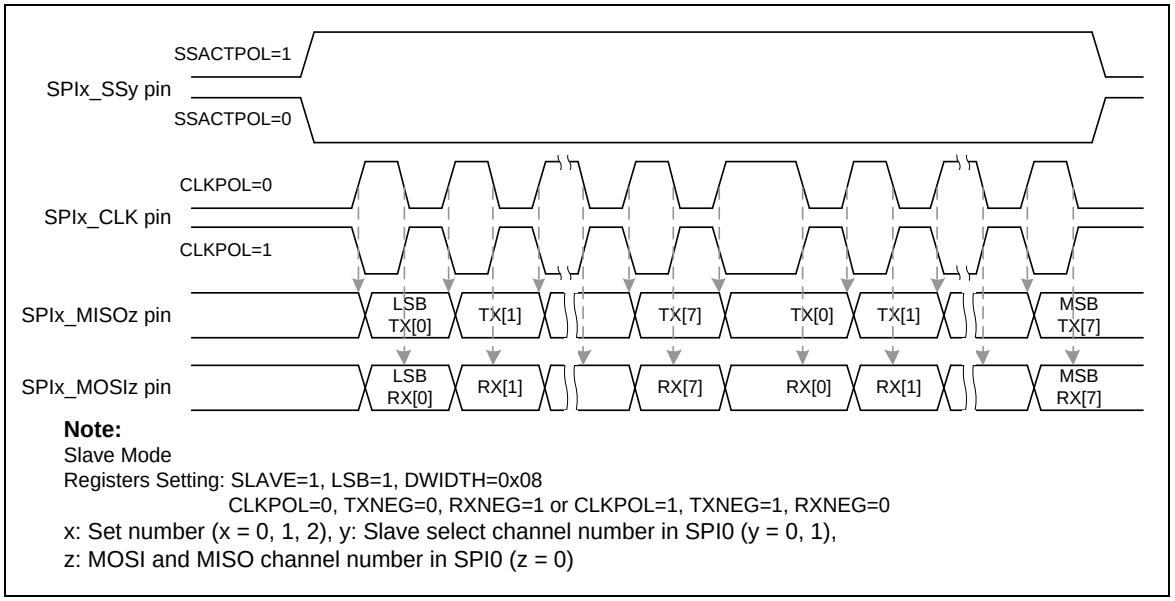
Note:
Master Mode
Registers Setting: SLAVE=0, LSB=1, DWIDTH=0x08
CLKPOL=0, TXNEG=0, RXNEG=1 or CLKPOL=1, TXNEG=1, RXNEG=0
x: Set number (x = 0, 1, 2), y: Slave select channel number in SPI0 (y = 0, 1),
z: MOSI and MISO channel number in SPI0 (z = 0)

Figure 6.14-36 SPI Timing in Master Mode (Alternate Phase of SPIx_CLK)



Note:
Slave Mode
Registers Setting: SLAVE=1, LSB=0, DWIDTH=0x08
CLKPOL=0, TXNEG=1, RXNEG=0 or CLKPOL=1, TXNEG=0, RXNEG=1
x: Set number (x = 0, 1, 2), y: Slave select channel number in SPI0 (y = 0, 1),
z: MOSI and MISO channel number in SPI0 (z = 0)

Figure 6.14-37 SPI Timing in Slave Mode



Note:
Slave Mode
Registers Setting: SLAVE=1, LSB=1, DWIDTH=0x08
CLKPOL=0, TXNEG=0, RXNEG=1 or CLKPOL=1, TXNEG=1, RXNEG=0
x: Set number (x = 0, 1, 2), y: Slave select channel number in SPI0 (y = 0, 1),
z: MOSI and MISO channel number in SPI0 (z = 0)

Figure 6.14-38 SPI Timing in Slave Mode (Alternate Phase of SPIx_CLK)

6.14.7 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 (SPIn_CLKDIV [8:0]) to determine the output frequency of SPI clock.
2) Write the SPIn_SSCTL register a proper value for the related settings of Master mode:
    1. Clear AUTOSS (SPIn_SSCTL[3]) to 0 to disable the Automatic Slave Selection function.
    2. Configure slave selection signal as active low by clearing SSACTPOL (SPIn_SSCTL[2]) to 0.
    3. Enable slave selection signal by setting SS (SPIn_SSCTL[0]) to 1 to activate the off-chip slave device.
3) Write the related settings into the SPIn_CTL register to control the SPI master actions.
    1. Configure this SPI controller as master device by setting SLAVE (SPIn_CTL[18]) to 0.
    2. Force the SPI clock idle state at low by clearing CLKPOL (SPIn_CTL[3]) to 0.
    3. Select data transmitted on negative edge of SPI bus clock by setting TXNEG (SPIn_CTL[2]) to 1.
    4. Select data latched on positive edge of SPI bus clock by clearing RXNEG (SPIn_CTL[1]) to 0.
    5. Set the bit length of a transaction as 8-bit in DWIDTH bit field (SPIn_CTL[12:8] = 0x08).
    6. Set MSB transfer first by clearing LSB (SPIn_CTL[13]) to 0.
4) Set SPIEN (SPIn_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 SPIn_TX register.
6) Waiting for SPI interrupt if the UNITIEN (SPIn_CTL[17]) is set to 1, or just polling the unit transfer interrupt flag UNITIF (SPIn_STATUS[1]).
7) Read out the received one byte data from SPIn_RX register.
8) Go to 5) to continue another data transfer or set SS (SPIn_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 SPIn_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 (SPIn_SSCTL[2]) to 1.

2) Write the related settings into the SPIn_CTL register to control this SPI slave actions
   1. Set the SPI controller as slave device by setting SLAVE (SPIn_CTL[18]) to 1.
   2. Select the SPI clock idle state at high by setting CLKPOL (SPIn_CTL[3]) to 1.
   3. Select data transmitted on negative edge of SPI bus clock by setting TXNEG (SPIn_CTL[2]) to 1.
   4. Select data latched on positive edge of SPI bus clock by clearing RXNEG (SPIn_CTL[1]) to 0.
   5. Set the bit length of a transaction as 8-bit in DWIDTH bit field (SPIn_CTL[12:8] = 0x08).
   6. Set LSB transfer first by setting LSB (SPIn_CTL[13]) to 1.
3) Set the SPIEN (SPIn_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 SPIn_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 SPIn_TX register does not need to be updated by software.
6) Waiting for SPI interrupt if the UNITIEN (SPIn_CTL[17]) is set to 1, or just polling the unit transfer interrupt flag UNITIF (SPIn_STATUS[1]).
7) Read out the received one byte data from SPIn_RX register.
8) Go to 4) to continue another data transfer or stop data transfer.

6.14.8 Register Map

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

Register	Offset	R/W	Description	Reset Value
SPI Base Address: **SPI0_BA = 0x4006_0000** **SPI1_BA = 0x4006_1000** **SPI2_BA = 0x4006_2000**				
SPI0_CTL	SPI0_BA+0x00	R/W	SPI0 Control Register	0x0000_0034
SPI0_CLKDIV	SPI0_BA+0x04	R/W	SPI0 Clock Divider Register	0x0000_0000
SPI0_SSCTL	SPI0_BA+0x08	R/W	SPI0 Slave Select Control Register	0x0000_0000
SPI0_PDMACTL	SPI0_BA+0x0C	R/W	SPI0 PDMA Control Register	0x0000_0000
SPI0_FIFOCTL	SPI0_BA+0x10	R/W	SPI0 FIFO Control Register	0x4400_0000
SPI0_STATUS	SPI0_BA+0x14	R/W	SPI0 Status Register	0x0005_0110
SPI0_TX	SPI0_BA+0x20	W	SPI0 Data Transmit Register	0x0000_0000
SPI0_RX	SPI0_BA+0x30	R	SPI0 Data Receive Register	0x0000_0000
SPI1_CTL	SPI1_BA+0x00	R/W	SPI1 Control Register	0x0000_0034
SPI1_CLKDIV	SPI1_BA+0x04	R/W	SPI1 Clock Divider Register	0x0000_0000
SPI1_SSCTL	SPI1_BA+0x08	R/W	SPI1 Slave Select Control Register	0x0000_0000
SPI1_PDMACTL	SPI1_BA+0x0C	R/W	SPI1 PDMA Control Register	0x0000_0000
SPI1_FIFOCTL	SPI1_BA+0x10	R/W	SPI1 FIFO Control Register	0x2200_0000
SPI1_STATUS	SPI1_BA+0x14	R/W	SPI1 Status Register	0x0005_0110
SPI1_TX	SPI1_BA+0x20	W	SPI1 Data Transmit Register	0x0000_0000
SPI1_RX	SPI1_BA+0x30	R	SPI1 Data Receive Register	0x0000_0000
SPI1_I2SCTL	SPI1_BA+0x60	R/W	SP1 I2S Control Register	0x0000_0000
SPI1_I2SCLK	SPI1_BA+0x64	R/W	SPI1 I2S Clock Divider Control Register	0x0000_0000
SPI1_I2SSTS	SPI1_BA+0x68	R/W	SPI1 I2S Status Register	0x0005_0100
SPI2_CTL	SPI2_BA+0x00	R/W	SPI2 Control Register	0x0000_0034
SPI2_CLKDIV	SPI2_BA+0x04	R/W	SPI2 Clock Divider Register	0x0000_0000
SPI2_SSCTL	SPI2_BA+0x08	R/W	SPI2 Slave Select Control Register	0x0000_0000
SPI2_PDMACTL	SPI2_BA+0x0C	R/W	SPI2 PDMA Control Register	0x0000_0000
SPI2_FIFOCTL	SPI2_BA+0x10	R/W	SPI2 FIFO Control Register	0x2200_0000
SPI2_STATUS	SPI2_BA+0x14	R/W	SPI2 Status Register	0x0005_0110
SPI2_TX	SPI2_BA+0x20	W	SPI2 Data Transmit Register	0x0000_0000
SPI2_RX	SPI2_BA+0x30	R	SPI2 Data Receive Register	0x0000_0000
SPI2_I2SCTL	SPI2_BA+0x60	R/W	SPI2 I2S Control Register	0x0000_0000
SPI2_I2SCLK	SPI2_BA+0x64	R/W	SPI2 I2S Clock Divider Control Register	0x0000_0000

Register	Offset	R/W	Description	Reset Value
SPI Base Address: **SPI0_BA = 0x4006_0000** **SPI1_BA = 0x4006_1000** **SPI2_BA = 0x4006_2000**				
SPI2_I2SSTS	SPI2_BA+0x68	R/W	SPI2 I2S Status Register	0x0005_0100

Note:

1. Any register not listed here is reserved and must not be written. The result of a read operation on these bits is undefined.
2. The reserved register fields that listed in register description must be written to their reset value. Writing reserved fields with other than reset values may produce undefined results.

6.14.9 Register Description

SPI Control Register (SPI0_CTL)

Register	Offset	R/W	Description	Reset Value
SPI0_CTL	SPI0_BA+0x00	R/W	SPI0 Control Register	0x0000_0034

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved	QUADIOEN	DUALIOEN	DATDIR	REORDER	SLAVE	UNITIEN	TWOBIT
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	
[31:23]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[22]	**QUADIOEN**	**Quad I/O Mode Enable Bit (Only Supported in SPI0)** 0 = Quad I/O mode Disabled. 1 = Quad I/O mode Enabled.
[21]	**DUALIOEN**	**Dual I/O Mode Enable Bit (Only Supported in SPI0)** 0 = Dual I/O mode Disabled. 1 = Dual I/O mode Enabled.
[20]	**DATDIR**	**Data Port Direction Control** This bit is used to select the data input/output direction in half-duplex transfer and Dual/Quad 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]	**TWOBIT**	**2-bit Transfer Mode Enable Bit (Only Supported in SPI0)**

		0 = 2-Bit Transfer mode Disabled. 1 = 2-Bit Transfer mode Enabled. **Note:** When 2-Bit Transfer mode is enabled, the first serial transmitted bit data is from the first FIFO buffer data, and the 2nd serial transmitted bit data is from the second FIFO buffer data. As the same as transmitted function, the first received bit data is stored into the first FIFO buffer and the 2nd received bit data is stored into the second FIFO buffer at the same time.
[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 (SPI0_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 SPI0_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.

[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.
[0]	**SPIEN**	**SPI Transfer Control Enable Bit** In Master mode, the transfer will start when there is data in the FIFO buffer after this bit 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 SPI0_CTL, SPI0_CLKDIV, SPI0_SSCTL and SPI0_FIFOCTL registers, user shall clear the SPIEN (SPI0_CTL[0]) and confirm the SPIENSTS (SPI0_STATUS[15]) is 0.

SPI Clock Divider Register (SPI0_CLKDIV)

Register	Offset	R/W	Description	Reset Value
SPI0_CLKDIV	SPI0_BA+0x04	R/W	SPI0 Clock Divider Register	0x0000_0000

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

Bits	Description	
[31:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8:0]	**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:** User should set DIVIDER carefully because the peripheral clock frequency must be slower than or equal to system frequency.

SPI Slave Select Control Register (SPI0_SSCTL)

Register	Offset	R/W	Description	Reset Value
SPI0_SSCTL	SPI0_BA+0x08	R/W	SPI0 Slave Select Control Register	0x0000_0000

31	30	29	28	27	26	25	24
SLVTOCNT							
23	22	21	20	19	18	17	16
SLVTOCNT							
15	14	13	12	11	10	9	8
Reserved		SSINAIEN	SSACTIEN	Reserved		SLVURIEN	SLVBEIEN
7	6	5	4	3	2	1	0
Reserved	SLVTORST	SLVTOIEN	SLV3WIRE	AUTOSS	SSACTPOL	SS1	SS

Bits	Description	
[31:16]	**SLVTOCNT**	**Slave Mode Time-out Period (Only Supported in SPI0)** In Slave mode, these bits indicate the time-out period when there is bus clock input during slave select active. The clock source of the time-out counter is Slave peripheral clock. If the value is 0, it indicates the slave mode time-out function is disabled.
[15:14]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.
[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]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[6]	**SLVTORST**	**Slave Mode Time-out Reset Control (Only Supported in SPI0)** 0 = When Slave mode time-out event occurs, the TX and RX control circuit will not be reset. 1 = When Slave mode time-out event occurs, the TX and RX control circuit will be reset by hardware.

[5]	**SLVTOIEN**	**Slave Mode Time-out Interrupt Enable Bit (Only Supported in SPI0)** 0 = Slave mode time-out interrupt Disabled. 1 = Slave mode time-out interrupt Enabled.
[4]	**SLV3WIRE**	**Slave 3-wire Mode Enable Bit (Only Supported in SPI0)** Slave 3-wire mode is only available in SPI0. In Slave 3-wire mode, the SPI controller can work with 3-wire interface including SPI0_CLK, SPI0_MISO and SPI0_MOSI pins. 0 = 4-wire bi-direction interface. 1 = 3-wire bi-direction interface.
[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 (SPI0_SSCTL[0]) and SS1 (SPI0_SSCTL[1]). 1 = Automatic slave selection function Enabled.
[2]	**SSACTPOL**	**Slave Selection Active Polarity** This bit defines the active polarity of slave selection signal. 0 = The slave selection signal is active low. 1 = The slave selection signal is active high.
[1]	**SS1**	**Slave Selection Control 1 (Master Only)** If AUTOSS bit is cleared to 0, 0 = Set the SPI0_SS1 line to inactive state. 1 = Set the SPI0_SS1 line to active state. If the AUTOSS bit is set to 1, 0 = Keep the SPI0_SS1 line at inactive state. 1 = SPI0_SS1 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 slave select line is specified in SSACTPOL (SPI0_SSCTL[2]). **Note:** SPI0_SS0 pin is defined as the slave select input in Slave mode.
[0]	**SS**	**Slave Selection Control (Master Only)** If AUTOSS bit is cleared to 0, 0 = Set the SPI0_SS0 line to inactive state. 1 = Set the SPI0_SS0 line to active state. If the AUTOSS bit is set to 1, 0 = Keep the SPI0_SS0 line at inactive state. 1 = SPI0_SS0 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 slave select line is specified in SSACTPOL (SPI0_SSCTL[2]). **Note:** SPI0_SS0 is defined as the slave select input in Slave mode.

SPI PDMA Control Register (SPI0_PDMACTL)

Register	Offset	R/W	Description	Reset Value
SPI0_PDMACTL	SPI0_BA+0x0C	R/W	SPI0 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. Any values read should be ignored. When writing to this field always write with reset value.
[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 = Receive PDMA function Disabled. 1 = Receive 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 (SPI0_FIFOCTL)

Register	Offset	R/W	Description	Reset Value
SPI0_FIFOCTL	SPI0_BA+0x10	R/W	SPI0 FIFO Control Register	0x4400_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]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[30:28]	**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]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[26:24]	**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.
[23:10]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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 (SPI0_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. **Note1:** The TX underflow event occurs if there is no any data in TX FIFO when the slave selection signal is active. **Note2:** When TX underflow event occurs, SPI0_MISO0 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 SPI0_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** 0 = No effect. 1 = Reset transmit FIFO pointer and transmit circuit. 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 3 system clock cycles + 2 peripheral clock cycles after it is set to 1. User can read TXRXRST (SPI0_STATUS[23]) to check if reset is accomplished or not. **Note:** If TX underflow event occurs in SPI Slave mode, this bit can be used to make SPI return to idle state.
[0]	**RXRST**	**Receive Reset** 0 = No effect. 1 = Reset receive FIFO pointer and receive circuit. 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 3 system clock cycles + 2 peripheral clock cycles after it is set to 1. User can read TXRXRST (SPI0_STATUS[23]) to check if reset is accomplished or not.

SPI Status Register (SPI0_STATUS)

Register	Offset	R/W	Description	Reset Value
SPI0_STATUS	SPI0_BA+0x14	R/W	SPI0 Status Register	0x0005_0110

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	SLVTOIF	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. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.
[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 peripheral clock periods in Master mode or over 576 SPI peripheral clock periods in Slave mode. When the received FIFO buffer is read by software, the time-out status will be cleared automatically. **Note:** This bit will be cleared by writing 1 to 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 overrun. 1 = Receive FIFO is overrun. **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 receive 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 event 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 active but there is no any bus clock input, the SLVBEIF also active when the slave select goes to inactive state. This bit will be cleared by writing 1 to it.

[5]	**SLVTOIF**	**Slave Time-out Interrupt Flag (Only Supported in SPI0)** When the slave select is active and the value of SLVTOCNT is not 0, as the bus clock is detected, the slave time-out counter in SPI controller logic will be started. When the value of time-out counter is greater than or equal to the value of SLVTOCNT (SPI0_SSCTL[31:16]) before one transaction is done, the slave time-out interrupt event will be asserted. 0 = Slave time-out is not active. 1 = Slave time-out is active. **Note:** This bit will be cleared by writing 1 to it.
[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 (SPI0_SSCTL[2]) is set 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. SPI0_CTL[0] = 1 and TXEMPTY = 0. b. For SPI Master mode, SPI0_CTL[0] = 1 and TXEMPTY = 1 but the current transaction is not finished yet. c. For SPI Master mode, SPI0_CTL[0] = 1 and RXONLY = 1. d. For SPI Slave mode, the SPI0_CTL[0] = 1 and there is serial clock input into the SPI core logic when slave select is active. For SPI Slave mode, the SPI0_CTL[0] = 1 and the transmit buffer or transmit shift register is not empty even if the slave select is inactive. **Note:** Please also check other indicators to know SPI is in idle or not.

SPI Data Transmit Register (SPI0_TX)

Register	Offset	R/W	Description	Reset Value
SPI0_TX	SPI0_BA+0x20	W	SPI0 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 8-level transmit FIFO buffers. The number of valid bits depends on the setting of DWIDTH (SPI0_CTL[12:8]). 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. **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 (SPI0_RX)

Register	Offset	R/W	Description	Reset Value
SPI0_RX	SPI0_BA+0x30	R	SPI0 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 8-level FIFO buffers in this controller. The data receive register holds the data received from SPI data input pin. If the RXEMPTY (SPI0_STATUS[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.

SPI Control Register (SPIn_CTL)

Register	Offset	R/W	Description	Reset Value
SPI1_CTL	SPI1_BA+0x00	R/W	SPI1 Control Register	0x0000_0034
SPI2_CTL	SPI2_BA+0x00	R/W	SPI2 Control Register	0x0000_0034

Note: Not supported in I^2S 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. Any values read should be ignored. When writing to this field always write with reset value.
[20]	**DATDIR**	**Data Port Direction Control** This bit is used to select the data input/output direction in half-duplex transfer and Dual/Quad 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. Any values read should be ignored. When writing to this field always write with reset value.
[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 (SPIn_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 SPIn_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. **Note:** For SPI1~SPI2, this bit field will decide the depth of TX/RX FIFO configuration in SPI mode**.** Therefore, changing this bit field will clear TX/RX FIFO by hardware automatically in SPI1~SPI2.
[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.
[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.

[0]	**SPIEN**	**SPI Transfer Control Enable Bit** In Master mode, the transfer will start when there is data in the FIFO buffer after this bit 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 SPIn_CTL, SPIn_CLKDIV, SPIn_SSCTL and SPIn_FIFOCTL registers, user shall clear the SPIEN (SPIn_CTL[0]) and confirm the SPIENSTS (SPIn_STATUS[15]) is 0.

SPI Clock Divider Register (SPIn_CLKDIV)

Register	Offset	R/W	Description	Reset Value
SPI1_CLKDIV	SPI1_BA+0x04	R/W	SPI1 Clock Divider Register	0x0000_0000
SPI2_CLKDIV	SPI2_BA+0x04	R/W	SPI2 Clock Divider Register	0x0000_0000

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

Bits	Description	
[31:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8:0]	**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. **Note1:** Not supported in I^2S mode. **Note2:** User should set DIVIDER carefully because the peripheral clock frequency must be slower than or equal to system frequency.

SPI Slave Select Control Register (SPIn_SSCTL)

Register	Offset	R/W	Description	Reset Value
SPI1_SSCTL	SPI1_BA+0x08	R/W	SPI1 Slave Select Control Register	0x0000_0000
SPI2_SSCTL	SPI2_BA+0x08	R/W	SPI2 Slave 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		**SSINAIEN**	**SSACTIEN**	**Reserved**		**SLVURIEN**	**SLVBEIEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**AUTOSS**	**SSACTPOL**	**Reserved**	**SS**

Bits	Description	
[31:14]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.
[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 (SPIn_SSCTL[0]). 1 = Automatic slave selection function Enabled.
[2]	**SSACTPOL**	**Slave Selection Active Polarity** This bit defines the active polarity of slave selection signal. 0 = The slave selection signal is active low.

		1 = The slave selection signal is active high.
[1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**SS**	**Slave Selection Control (Master Only)** If AUTOSS bit is cleared to 0, 0 = Set the SPIn_SS line to inactive state. 1 = Set the SPIn_SS line to active state. If the AUTOSS bit is set to 1, 0 = Keep the SPIn_SS line at inactive state. 1 = SPIn_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 slave select line is specified in SSACTPOL (SPIn_SSCTL[2]).

Note: Not supported in I^2S mode.

SPI PDMA Control Register (SPIn_PDMACTL)

Register	Offset	R/W	Description	Reset Value
SPI1_PDMACTL	SPI1_BA+0x0C	R/W	SPI1 PDMA Control Register	0x0000_0000
SPI2_PDMACTL	SPI2_BA+0x0C	R/W	SPI2 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. Any values read should be ignored. When writing to this field always write with reset value.
[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 = Receive PDMA function Disabled. 1 = Receive 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 (SPIn_FIFOCTL)

Register	Offset	R/W	Description	Reset Value
SPI1_FIFOCTL	SPI1_BA+0x10	R/W	SPI1 FIFO Control Register	0x2200_0000
SPI2_FIFOCTL	SPI2_BA+0x10	R/W	SPI2 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]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[30:28]	**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. For SPI1~SPI2, the MSB of this bit field is only meaningful while SPI mode 8~16 bits of data length.
[27]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[26:24]	**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. For SPI1~SPI2, the MSB of this bit field is only meaningful while SPI mode 8~16 bits of data length.
[23:10]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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 (SPIn_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. **Note1:** The TX underflow event occurs if there is no any data in TX FIFO when the slave selection signal is active. **Note2:** This bit should be set as 0 in I^2S mode. **Note3:** When TX underflow event occurs, SPIn_MISO (n=1, 2) 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 SPIn_MISO (n=1, 2) 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** 0 = No effect. 1 = Reset transmit FIFO pointer and transmit circuit. 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 3 system clock cycles + 2 peripheral clock cycles after it is set to 1. User can read TXRXRST (SPIn_STATUS[23]) to check if reset is accomplished or not. **Note:** If TX underflow event occurs in SPI Slave mode, this bit can be used to make SPI return to idle state.
[0]	**RXRST**	**Receive Reset** 0 = No effect. 1 = Reset receive FIFO pointer and receive circuit. 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 3 system clock cycles + 2 peripheral clock cycles after it is set to 1. User can read TXRXRST (SPIn_STATUS[23]) to check if reset is accomplished or not.

SPI Status Register (SPIn_STATUS)

Register	Offset	R/W	Description	Reset Value
SPI1_STATUS	SPI1_BA+0x14	R/W	SPI1 Status Register	0x0005_0110
SPI2_STATUS	SPI2_BA+0x14	R/W	SPI2 Status Register	0x0005_0110

Note: Not supported in I^2S 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. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.
[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 peripheral clock periods in Master mode or over 576 SPI peripheral clock periods in Slave mode. When the received FIFO buffer is read by software, the time-out status will be cleared automatically. **Note:** This bit will be cleared by writing 1 to 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 overrun. 1 = Receive FIFO is overrun. **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 receive 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 event 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 active but there is no any bus clock input, the SLVBEIF also active when the slave select goes to inactive state. This bit will be cleared by writing 1 to it.
[5]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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 (SPIn_SSCTL[2]) is set 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: e. SPIn_CTL[0] = 1 and TXEMPTY = 0. f. For SPI Master mode, SPIn_CTL[0] = 1 and TXEMPTY = 1 but the current transaction is not finished yet. g. For SPI Master mode, SPIn_CTL[0] = 1 and RXONLY = 1. h. For SPI Slave mode, the SPIn_CTL[0] = 1 and there is serial clock input into the SPI core logic when slave select is active. For SPI Slave mode, the SPIn_CTL[0] = 1 and the transmit buffer or transmit shift register is not empty even if the slave select is inactive. **Note:** Please also check other indicators to know SPI is in idle or not.

SPI Data Transmit Register (SPIn_TX)

Register	Offset	R/W	Description	Reset Value
SPI1_TX	SPI1_BA+0x20	W	SPI1 Data Transmit Register	0x0000_0000
SPI2_TX	SPI2_BA+0x20	W	SPI2 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 buffers. The number of valid bits depends on the setting of DWIDTH (SPIn_CTL[12:8]) in SPI mode or WDWIDTH (SPIn_I2SCTL[5:4]) in I^2S 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^2S mode, if WDWIDTH (SPIn_I2SCTL[5:4]) is set to 0x2, the data width of audio channel is 24-bit and corresponding to TX[23:0]. If WDWIDTH is set as 0x0, 0x1, or 0x3, all bits of this field are valid and referred to the data arrangement in I^2S 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 (SPIn_RX)

Register	Offset	R/W	Description	Reset Value
SPI1_RX	SPI1_BA+0x30	R	SPI1 Data Receive Register	0x0000_0000
SPI2_RX	SPI2_BA+0x30	R	SPI2 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 (SPIn_STATUS[8] or SPIn_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.

I2S Control Register (SPIn_I2SCTL)

Register	Offset	R/W	Description	Reset Value
SPI1_I2SCTL	SPI1_BA+0x60	R/W	SP1 I2S Control Register	0x0000_0000
SPI2_I2SCTL	SPI2_BA+0x60	R/W	SPI2 I2S 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					FLZCDEN	FRZCDEN
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. Any values read should be ignored. When writing to this field always write with reset value.
[29:28]	**FORMAT**	**Data Format Selection** 00 = I^2S data format. 01 = MSB justified data format. 10 = PCM mode A. 11 = PCM mode B.
[27:26]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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^2S 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. Any values read should be ignored. When writing to this field always write with reset value.

[17]	**FLZCDEN**	**Force Left Channel Zero Cross Data Option 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 SPIn_I2SSTS register is set to 1 and left channel data will force zero. This function is only available in transmit operation. 0 = Keep Left channel data, when zero crossing flag on. 1 = Force Left channel data to zero, when zero crossing flag on.
[16]	**FRZCDEN**	**Force Right Channel Zero Cross Data Option 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 SPIn_I2SSTS register is set to 1 and right channel data will force zero. This function is only available in transmit operation. 0 = Keep Right channel data, when zero crossing flag on. 1 = Force Right channel data to zero, when zero crossing flag on.
[15]	**MCLKEN**	**Master Clock Enable Bit** If MCLKEN is set to 1, I^2S controller will generate master clock on SPIx_I2SMCLK pin for external audio devices. 0 = Master clock Disabled. 1 = Master clock Enabled.
[14:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8]	**SLAVE**	**Slave Mode** I^2S can operate as master or slave. For Master mode, I2Sx_BCLK and I2Sx_LRCLK pins are output mode and send bit clock from NPCA121 series to audio CODEC chip. In Slave mode, I2Sx_BCLK and I2Sx_LRCLK pins are input mode and I2Sx_BCLK and I2Sx_LRCLK signals are received from outer audio CODEC chip. 0 = Master mode. 1 = Slave mode.
[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^2S Controller Enable Bit** 0 = Disabled I^2S mode. 1 = Enabled I^2S mode. **Note:** 1. If enable this bit, I2Sx_BCLK will start to output in Master mode. 2. Before changing the configurations of SPIn_I2SCTL, SPIn_I2SCLK, and SPIn_FIFOCTL registers, user shall clear the I2SEN (SPIn_I2SCTL[0]) and confirm the I2SENSTS (SPIn_I2SSTS[15]) is 0.

I2S Clock Divider Control Register (SPIn_I2SCLK)

Register	Offset	R/W	Description	Reset Value
SPI1_I2SCLK	SPI1_BA+0x64	R/W	SPI1 I²S Clock Divider Control Register	0x0000_0000
SPI2_I2SCLK	SPI2_BA+0x64	R/W	SPI2 I²S 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	
[31:18]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[17:8]	**BCLKDIV**	**Bit Clock Divider** The I²S controller will generate bit clock in Master mode. The clock frequency of bit clock , f_{BCLK}, is determined by the following expression: $f_{BCLK} = \frac{f_{i2s\_clock\_src}}{2 \times (\text{BCLKDIV} + 1)}$ where $f_{i2s\_clock\_src}$ is the frequency of I²S peripheral clock source, which is defined in the clock control register CLK_CLKSEL2. In I²S Slave mode, this field is used to define the frequency of peripheral clock and it's determined by $f_{i2s\_clock\_src} \div \left(\frac{\text{BCLKDIV}}{2} + 1 \right)$. The peripheral clock frequency in I²S 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]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[6:0]	**MCLKDIV**	**Master Clock Divider** If MCLKEN is set to 1, I²S 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,. $f_{MCLK} = \frac{f_{i2s\_clock\_src}}{2 \times \text{MCLKDIV}}$ If MCLKDIV = 0,. $f_{MCLK} = f_{i2s\_clock\_src}$ where $f_{i2s\_clock\_src}$ is the frequency of I²S 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.

I2S Status Register (SPIn_I2SSTS)

Register	Offset	R/W	Description	Reset Value
SPI1_I2SSTS	SPI1_BA+0x68	R/W	SPI1 I^2S Status Register	0x0005_0100
SPI2_I2SSTS	SPI2_BA+0x68	R/W	SPI2 I^2S 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. Any values read should be ignored. When writing to this field always write with reset value.
[30:28]	**TXCNT**	**Transmit FIFO Data Count (Read Only)** This bit field indicates the valid data count of transmit FIFO buffer.
[27]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.
[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/I^{2}S 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 control logic is disabled, this bit indicates the real status of SPI/I^{2}S control logic for user.
[14:13]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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 peripheral clock period in Master mode or over 576 SPI peripheral clock period in Slave mode. When the received FIFO buffer is read by software, the time-out status will be cleared automatically. **Note:** This bit will be cleared by writing 1 to 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 receive 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. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.

6.15 CRC Controller (CRC)

6.15.1 Overview

The Cyclic Redundancy Check (CRC) generator can perform CRC calculation with programmable polynomial settings.

6.15.2 Features

- Supports four common polynomials CRC-CCITT, CRC-8, CRC-16, and CRC-32
  - CRC-CCITT: $X^{16} + X^{12} + X^5 + 1$
  - CRC-8: $X^8 + X^2 + X + 1$
  - CRC-16: $X^{16} + X^{15} + X^2 + 1$
  - CRC-32: $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$
- Programmable seed value
- Supports programmable order reverse setting for input data and CRC checksum
- Supports programmable 1's complement setting for input data and CRC checksum
- Supports 8/16/32-bit of data width
  - 8-bit write mode: 1-AHB clock cycle operation
  - 16-bit write mode: 2-AHB clock cycle operation
  - 32-bit write mode: 4-AHB clock cycle operation
- Supports using PDMA to write data to perform CRC operation

6.15.3 Block Diagram

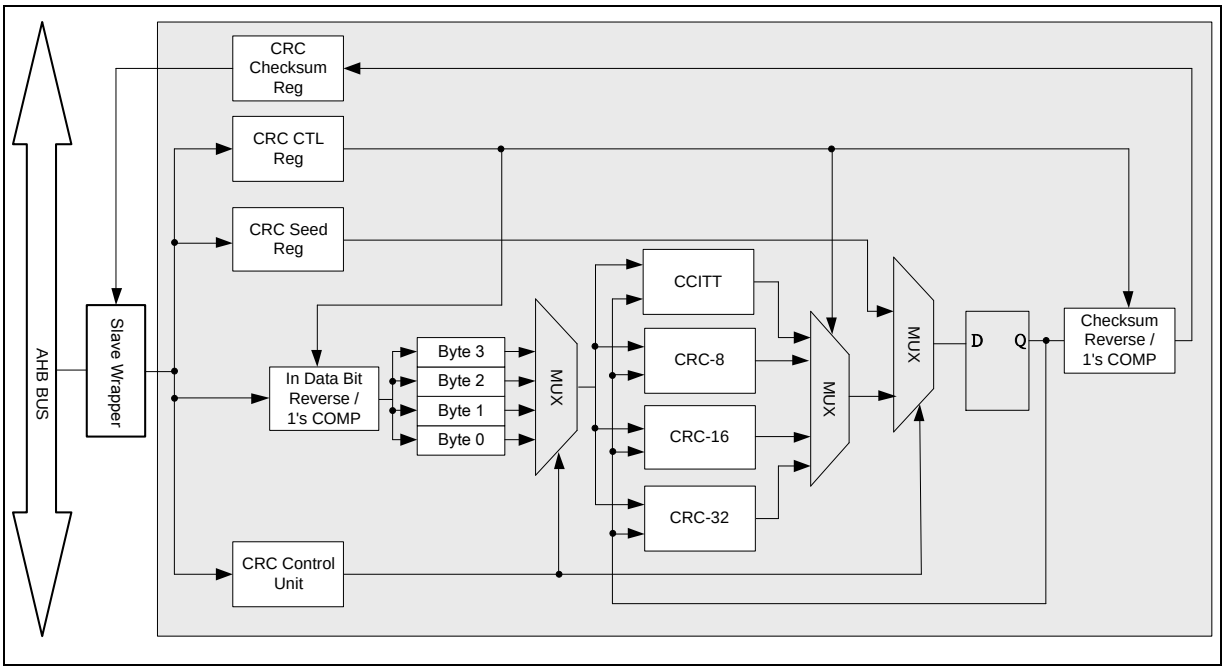


Figure 6.15-1 CRC Generator Block Diagram

6.15.4 Basic Configuration

- Clock source configuration
  - Enable CRC peripheral clock in CRCCKEN (CLK_AHBCLK[7]).
- Reset configuration
  - Reset CRC controller in CRCRST (SYS_IPRST0[7]).

6.15.5 Functional Description

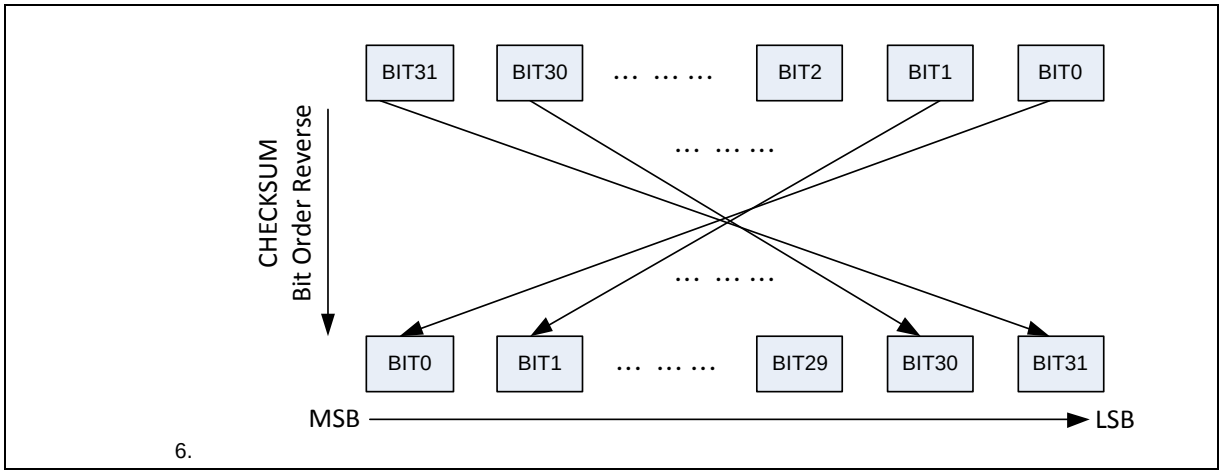
CRC generator can perform CRC calculation with programmable polynomial settings. The operation polynomial includes CRC-CCITT, CRC-8, CRC-16 and CRC-32; User can choose the CRC operation polynomial mode by setting CRCMODE[1:0] (CRC_CTL[31:30] CRC Polynomial Mode).

The following is a program sequence example.

1. Enable CRC generator by setting CRCEN (CRC_CTL[0] CRC Channel Enable Bit).
2. Initial setting for CRC calculation.
   - Configure 1's complement for CRC checksum by setting CHKSFMT (CRC_CTL[27] Checksum 1's Complement).
   - Configure bit order reverse for CRC checksum by setting CHKSREV (CRC_CTL[25] Checksum Bit Order Reverse). The functional block is also shown in Figure 6.15-2.
   - Configure 1's complement for CRC write data by setting DATFMT (CRC_CTL[26] Write Data 1's Complement).

- Configure bit order reverse for CRC write data per byte by setting DATREV (CRC_CTL[24] Write Data Bit Order Reverse). The functional block is also shown in Figure 6.15-3.

3. Perform CHKSINIT (CRC_CTL[1])] Checksum Initialization) to load the initial checksum value from CRC_SEED register value
4. Write data to CRC_DAT register to calculate CRC checksum.
5. Get the CRC checksum result by reading CRC_CHECKSUM register.



6.

Figure 6.15-2 CHECKSUM Bit Order Reverse Functional Block

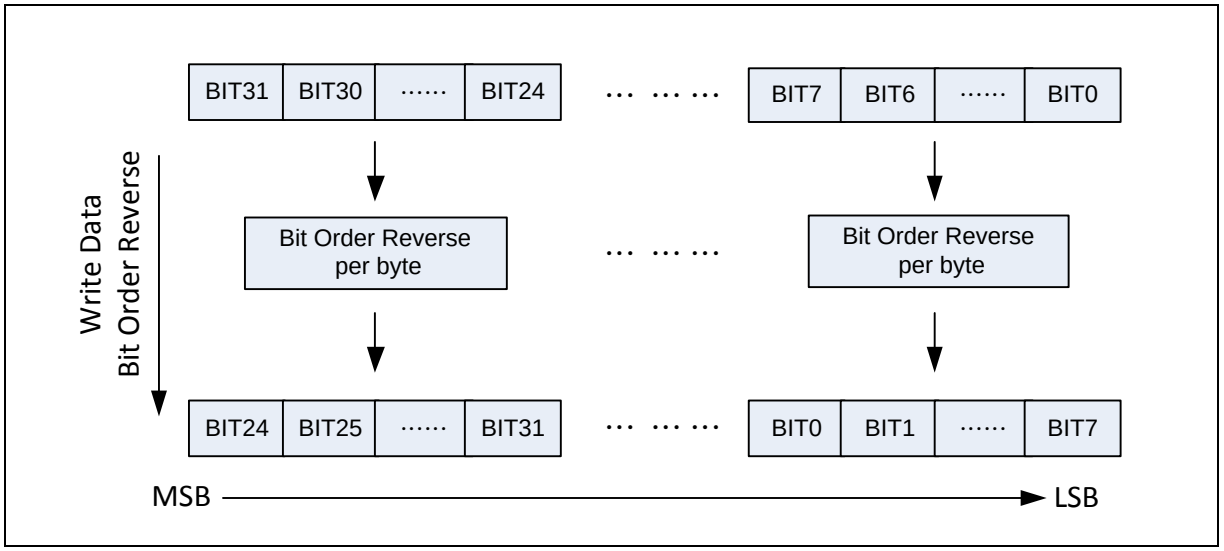


Figure 6.15-3 Write Data Bit Order Reverse Functional Block

6.15.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
CRC Base Address: **CRC_BA= 0x4003_1000**				
CRC_CTL	CRC_BA+0x00	R/W	CRC Control Register	0x2000_0000
CRC_DAT	CRC_BA+0x04	R/W	CRC Write Data Register	0x0000_0000
CRC_SEED	CRC_BA+0x08	R/W	CRC Seed Register	0xFFFF_FFFF
CRC_CHECKSUM	CRC_BA+0x0C	R	CRC Checksum Register	0xFFFF_FFFF

Note:

1. Any register not listed here is reserved and must not be written. The result of a read operation on these bits is undefined.
2. The reserved register fields that listed in register description must be written to their reset value. Writing reserved fields with other than reset values may produce undefined results.

6.15.7 Register Description

CRC Control Register (CRC_CTL)

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

31	30	29	28	27	26	25	24
CRCMODE		DATLEN		CHKSFMT	DATFMT	CHKSREV	DATREV
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						CHKSINIT	CRCEN

Bits	Description	
[31:30]	**CRCMODE**	**CRC Polynomial Mode** This field indicates the CRC operation polynomial mode. 00 = CRC-CCITT Polynomial mode. 01 = CRC-8 Polynomial mode. 10 = CRC-16 Polynomial mode. 11 = CRC-32 Polynomial mode.
[29:28]	**DATLEN**	**CPU Write Data Length** This field indicates the write data length. 00 = Data length is 8-bit mode. 01 = Data length is 16-bit mode. 1x = Data length is 32-bit mode. **Note:** When the write data length is 8-bit mode, the valid data in CRC_DAT register is only DATA[7:0] bits; if the write data length is 16-bit mode, the valid data in CRC_DAT register is only DATA[15:0].
[27]	**CHKSFMT**	**Checksum 1's Complement** This bit is used to enable the 1's complement function for checksum result in CRC_CHECKSUM register. 0 = 1's complement for CRC checksum Disabled. 1 = 1's complement for CRC checksum Enabled.
[26]	**DATFMT**	**Write Data 1's Complement** This bit is used to enable the 1's complement function for write data value in CRC_DAT register. 0 = 1's complement for CRC writes data in Disabled. 1 = 1's complement for CRC writes data in Enabled.

[25]	**CHKSREV**	**Checksum Bit Order Reverse** This bit is used to enable the bit order reverse function for write data value in CRC_CHECKSUM register. 0 = Bit order reverse for CRC checksum Disabled. 1 = Bit order reverse for CRC checksum Enabled. **Note:** If the checksum result is 0xDD7B0F2E, the bit order reverse for CRC checksum is 0x74F0DEBB.
[24]	**DATREV**	**Write Data Bit Order Reverse** This bit is used to enable the bit order reverse function for write data value in CRC_DAT register. 0 = Bit order reversed for CRC write data in Disabled. 1 = Bit order reversed for CRC write data in Enabled (per byte). **Note:** If the write data is 0xAABBCCDD, the bit order reverse for CRC write data is 0x55DD33BB.
[23:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1]	**CHKSINIT**	**Checksum Initialization** 0 = No effect. 1 = Initial checksum value by auto reload CRC_SEED register value to CRC_CHECKSUM register value. **Note:** This bit will be cleared automatically.
[0]	**CRCEN**	**CRC Channel Enable Bit** 0 = No effect. 1 = CRC operation Enabled.

CRC Write Data Register (CRC_DAT)

Register	Offset	R/W	Description	Reset Value
CRC_DAT	CRC_BA+0x04	R/W	CRC Write Data Register	0x0000_0000

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

Bits	Description	
[31:0]	DATA	**CRC Write Data Bits** User can write data directly by CPU mode or use PDMA function to write data to this field to perform CRC operation. **Note:** When the write data length is 8-bit mode, the valid data in CRC_DAT register is only DATA[7:0] bits; if the write data length is 16-bit mode, the valid data in CRC_DAT register is only DATA[15:0].

CRC Seed Register (CRC_SEED)

Register	Offset	R/W	Description	Reset Value
CRC_SEED	CRC_BA+0x08	R/W	CRC Seed Register	0xFFFF_FFFF

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

Bits	Description	
[31:0]	SEED	**CRC Seed Value** This field indicates the CRC seed value. **Note:** This field will be reloaded as checksum initial value (CRC_CHECKSUM register) after perform CHKSINIT (CRC_CTL[1]).

CRC Checksum Register (CRC_CHECKSUM)

Register	Offset	R/W	Description	Reset Value
CRC_CHECKSUM	CRC_BA+0x0C	R	CRC Checksum Register	0xFFFF_FFFF

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

Bits	Description	
[31:0]	CHECKSUM	**CRC Checksum Results** This field indicates the CRC checksum result.

6.16 Enhanced 12-bit Analog-to-Digital Converter (EADC)

6.16.1 Overview

The NPCA121 series contains one 12-bit successive approximation analog-to-digital converter (SAR ADC converter) with 13 external input channels. The ADC converter can be started by software trigger, PWM0 triggers, timer0~3 overflow pulse triggers, ADINT0, ADINT1 interrupt EOC (End of conversion) pulse trigger and external pin (EADC0_ST) input signal.

6.16.2 Features

- Analog input voltage range: 0~AV_{DD}.
- Reference voltage from AVDD.
- 12-bit resolution and 9-bit accuracy is guaranteed.
- Up to 13 single-end analog external input channels.
- Four ADC interrupts (ADINT0~3) with individual interrupt vector addresses.
- Maximum ADC clock frequency is 60 MHz.
- Up to 2 MSPS conversion rate.
- Configurable ADC internal sampling time.
- 12-bit, 10-bit, 8-bit, 6-bit configurable resolution.
- Supports calibration capability when EADC enabled.
- Supports three power saving modes:
  - Deep Power-down mode
  - Power-down mode.
  - Standby mode.
- Up to 13 sample modules
  - Each of sample modules which is configurable for ADC converter channel EADC_CH0~12 and trigger source.
  - Double buffer for sample control logic module 0~3
  - Configurable sampling time for each sample module.
  - Conversion results are held in 13 data registers with valid and overrun indicators.
- An ADC conversion can be started by:
  - Write 1 to SWTRGn (EADC_SWTRG[n] , n = 0~12)
  - External pin EADC0_ST
  - Timer0~3 overflow pulse triggers
  - ADINT0 and ADINT1 interrupt EOC (End of conversion) pulse triggers
  - PWM triggers
- Supports PDMA transfer

6.16.3 Block Diagram

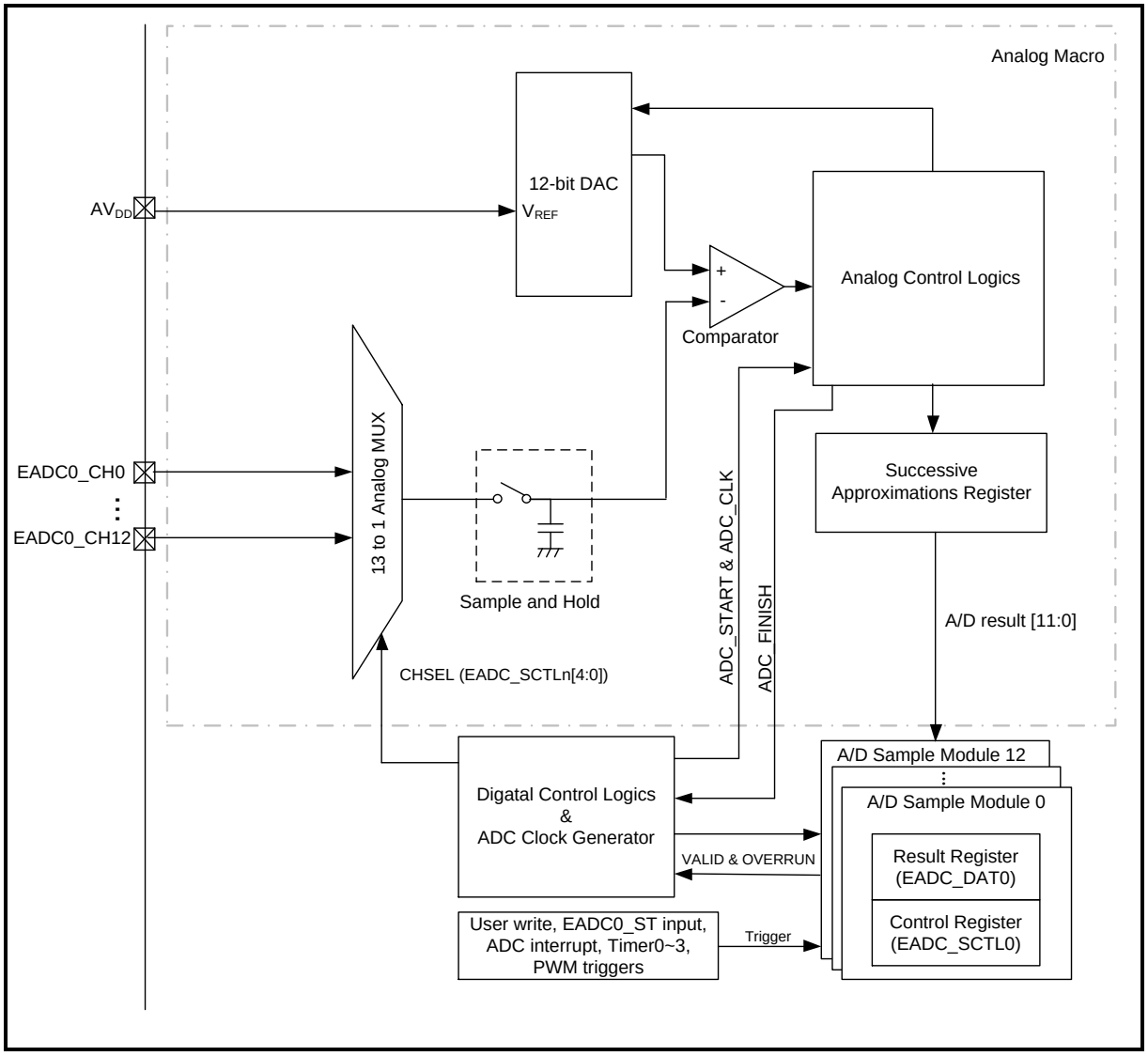


Figure 6.16-1 ADC Converter Block Diagram

6.16.4 Basic Configuration

- Clock source configuration
  - Select the clock divider number in EADCDIV (CLK_CLKDIV0[23:16]).
  - Enable EADC peripheral clock in EADCCKEN (CLK_APBCLK0[28]).
- Reset configuration
  - Reset EADC controller in ADCRST (EADC_CTL[1]).
- Pin configuration

Group	Pin Name	GPIO	MFP
EADC0	EADC0_CH0	PA.0	MFP2
	EADC0_CH1	PA.1	MFP2

	EADC0_CH2	PA.2	MFP2
	EADC0_CH3	PA.3	MFP2
	EADC0_CH4	PA.4	MFP2
	EADC0_CH5	PA.5	MFP2
	EADC0_CH6	PA.6	MFP2
	EADC0_CH7	PA.7	MFP2
	EADC0_CH8	PA.8	MFP2
	EADC0_CH9	PA.9	MFP2
	EADC0_CH10	PD.13	MFP2
	EADC0_CH11	PD.14	MFP2
	EADC0_CH12	PD.15	MFP2
	EADC0_ST	PA.10	MFP2
		PD.10	MFP2

Note: After the EADC pins is configured to ADC analog input, corresponding DINOFF (Px_DINOFF[31:16]) should be set to 1 to disable digital input path.

6.16.5 Functional Description

The EADC controller consists of a 13 channel analog switch, 13 sample modules and a 12-bit successive approximation analog-to-digital converter. The EADC operation is based on sample module 0~12, each of them has its configuration to decide which trigger source to start the conversion, which channel to convert. Sample module 0~12 can be configured to EADC_CH0~12 channel, and different trigger source. It provides user a flexible means to get the over-sampling results. The sample module 0~3 and sample module 4~12 are shows as follows.

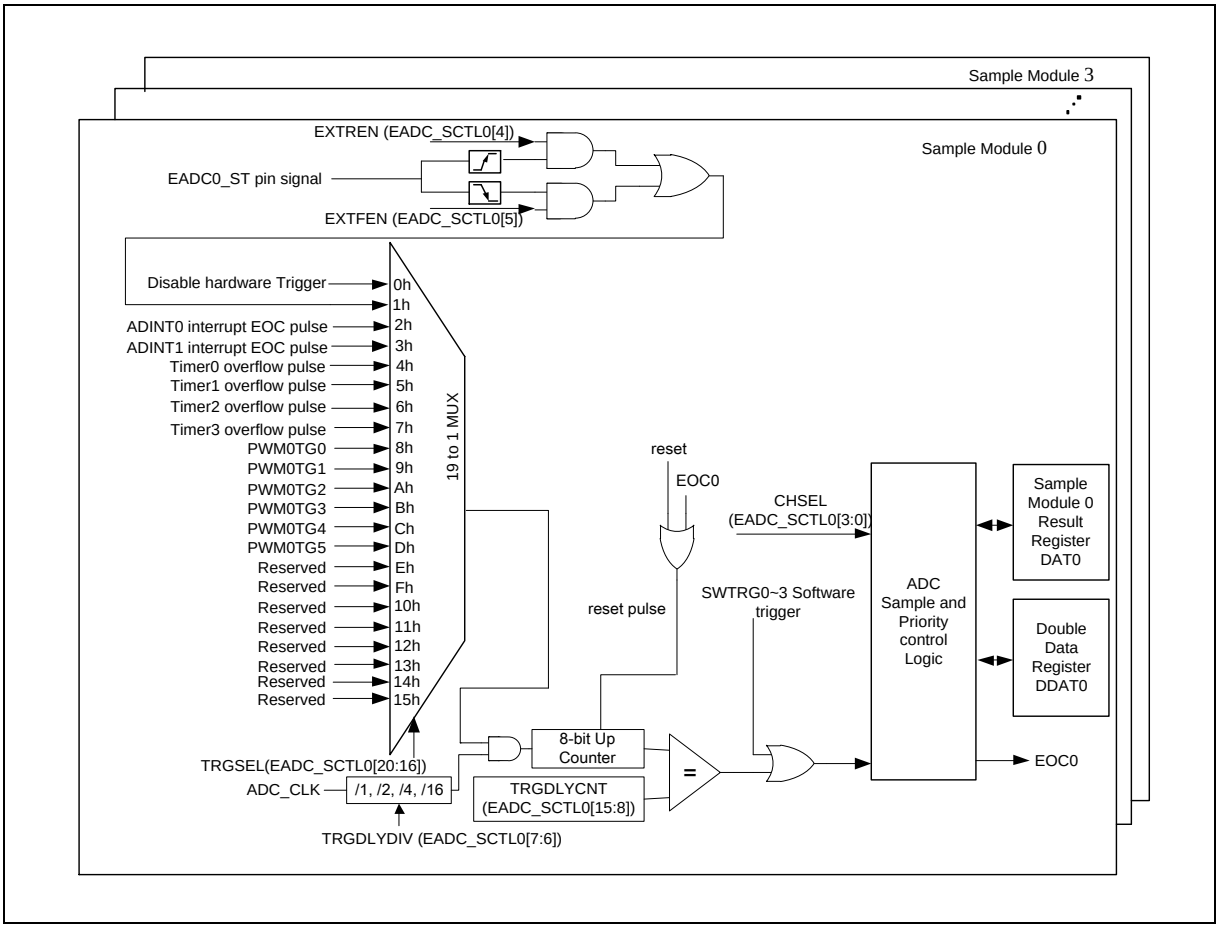


Figure 6.16-2 Sample Module 0~3 Block Diagram

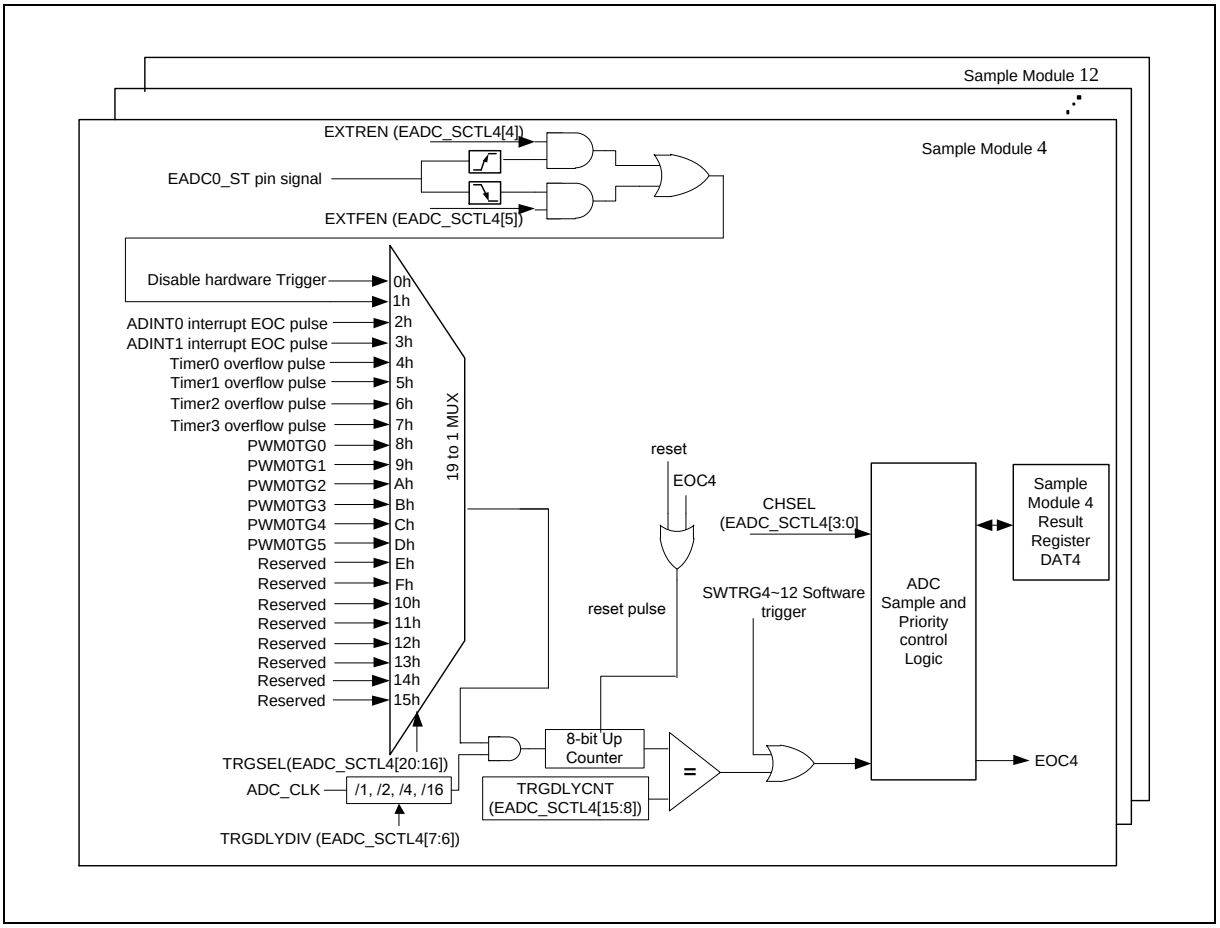


Figure 6.16-3 Sample Module 4~12 Block Diagram

The ADC conversion trigger sources in sample module 0~12 are listed below:

- Write 1 to SWTRGn (EADC_SWTRG[n], n = 0~12)
- External pin EADC0_ST
- Timer0~3 overflow pulse triggers
- ADINT0, ADINT1 ADC interrupt EOC (End of conversion) pulse triggers
- PWM triggers

The ADINT0 or ADINT1 interrupt pulses are generated whenever the specific sample module ADC EOC (End of conversion) pulse is generated. ADINT0 or ADINT1 interrupt pulse triggers can be fed back to trigger another ADC conversion, and is useful if a continuous scan conversion is needed.

6.16.5.1 ADC Clock Generator

The maximum EADC clock frequency is up to 60 MHz and the maximum sampling rate is up to 2 MSPS.

The clock control of EADC is shown as Figure 6.16-4. The EADC peripheral clock source is from HCLK clock, the ADC clock frequency is divided by an 8-bit pre-scalar with the following formula :

EADC clock frequency = (PCLK1) / (EADCDIV (CKL_CLKDIV0[23:16])+1)

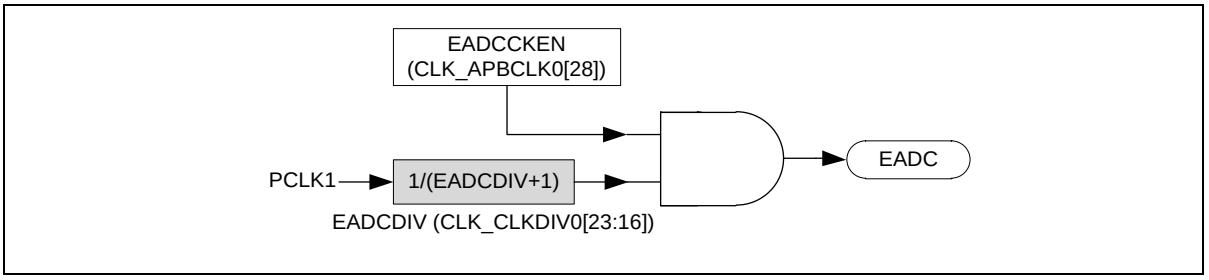


Figure 6.16-4 EADC Clock Control

6.16.5.2 ADC Software *Trigger Mode*

When a ADC conversion is performed on the sample module specified single channel, the operations are as follows:

1. ADC conversion is started when the SWTRGn (EADC_SWTRG[n], n=0~12) is set to 1 by user or other trigger inputs.
2. When ADC conversion is finished, the 12-bit result is stored in the ADC data register EADC_DATn (n=0~12) corresponding to the sample module.
3. On completion of conversion, the ADIFn (EADC_STATUS2[3:0], n=0~3) is set to 1 and ADC interrupt (ADINTn, n=0~3) is requested if the ADCIENn (EADC_CTL[5:2], n=0~3) bit is set to 1.
4. The SWTRGn (n=0~12) bit remains 1 during ADC conversion. When ADC conversion ends, the SWTRGn (n=0~12) bit is automatically cleared to 0 and the ADC converter will do another pending conversion.

The timing diagram of a conversion is shown as Figure 6.16-5

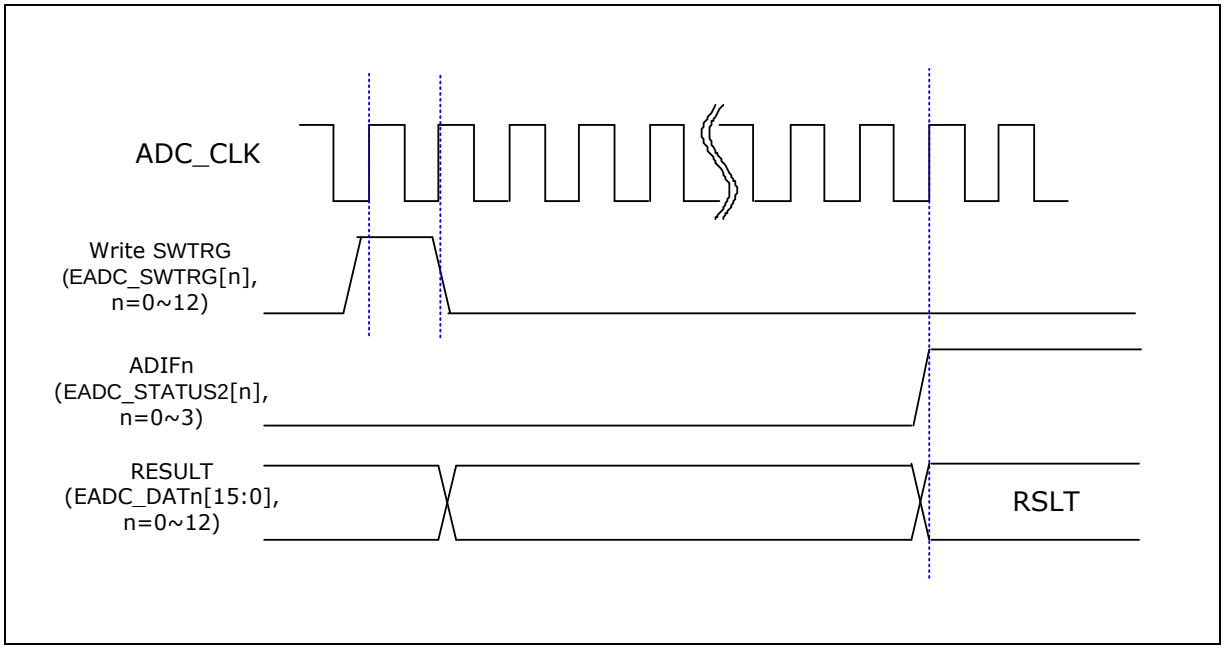


Figure 6.16-5 Example ADC Conversion Timing Diagram, n=0~12

If more than one sample module is enabled to convert analog single, the sample module specified channel with highest priority is firstly converted and other enabled sample module will be pended. The lower number sample module has higher priority. The sample module 0 is highest priority and the sample module 12 is lowest priority.

Note: If the interval between next conversion is more than 100 us, ADC would enter idle state automatically. User needs to execute a dummy conversion before normal operation. In other words, the first conversion result is incorrect when ADC is in idle state.

6.16.5.3 ADC Conversion Priority

There is a priority group converter for determining the conversion order when multiple sample module trigger flags are set at the same time.

Sample module with lower number has higher priority than the higher number sample module, The priority of sample module is shown as Figure 6.16-6. When more than one Sample Module are triggered at the same time, the Sample Module with lower number will start to convert first. The other Sample Module will be in the queue and the corresponding pending flag STPF(EADC_PENDSTS[n], n=0~12) are set to 1 by HW. After the Sample Module finish the conversion, STPF(EADC_PENDSTS[n], n=0~12) will be set to 0 automatically. If the Sample Module which is in the queue is triggered once more, the corresponding Overrun Flag SPOVF(EADC_OVSTS[n], n=0~12) will be set to 1 by HW.

For example, the Sample Module 0, 2, 3, 5 are triggered simultaneously. The input channel of Sample Module 0 will be converted first. Sample Module 2, 3, 5 will be suspended and STPF (EADC_PENDSTS[2], EADC_PENDSTS [3], EADC_PENDSTS [5]) will be set to 1. If Sample Module 5 is trigger once more in the same time, SPOVF(EADC_OVSTS[5]) will be set to 1.

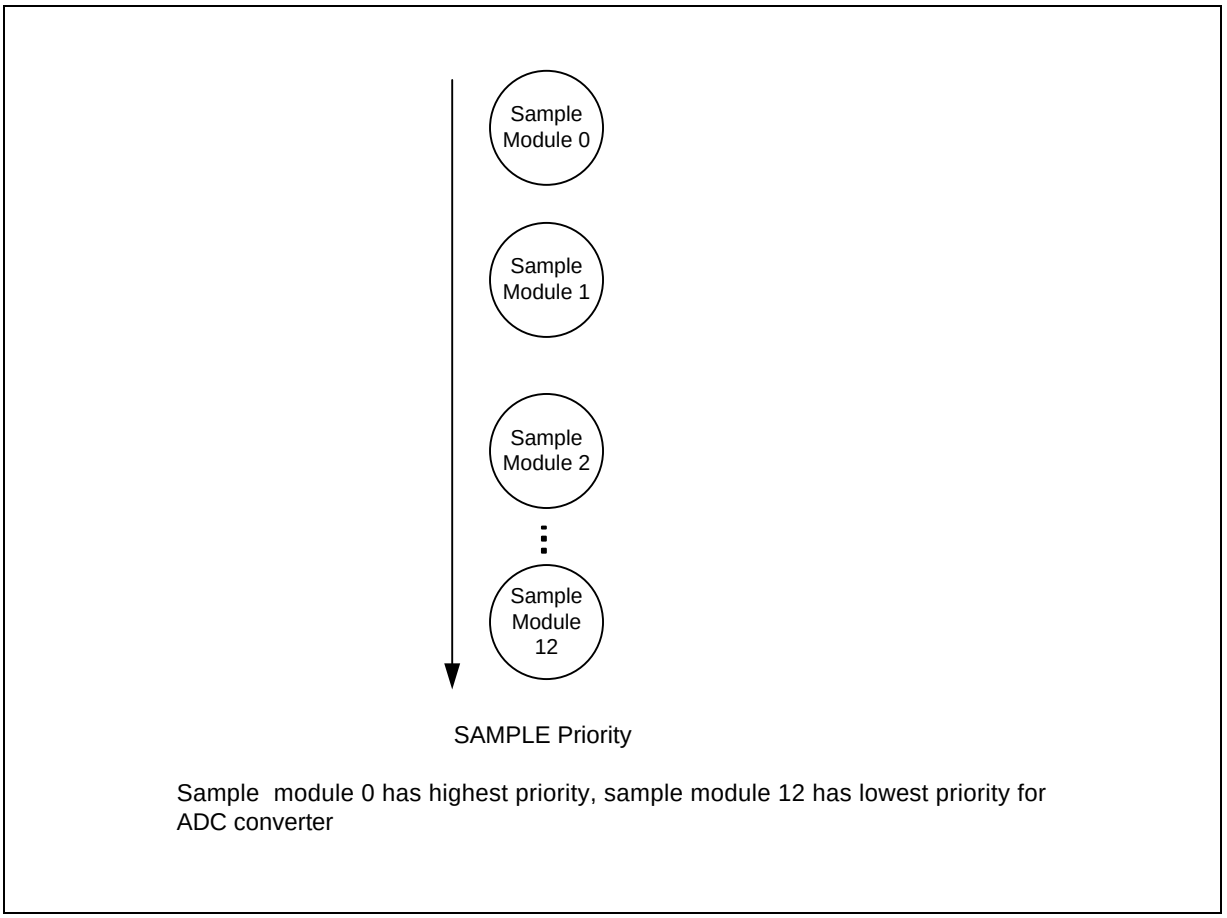


Figure 6.16-6 Sample module Conversion Priority Arbitrator Diagram

6.16.5.4 Conversion Cycles and Sampling Rate Frequency

There are four kinds of resolution which could be configured by RESSEL (EADC_CTL[7:6]). Each resolution corresponds to different conversion cycles. The relation is as Table 6.16.5-1.

Resolution	Minimum Conversion Cycles
6 bit	8 ADC_CLK
8 bit	10 ADC_CLK
10 bit	12 ADC_CLK
12 bit	14 ADC_CLK

Table 6.16.5-1 The relation between resolution and conversion cycles

The sampling rate frequency can be computed with the following formula :

Sampling rate frequency = (EADC clock frequency) / (conversion cycles)

6.16.5.5 Maximum Sampling Frequency Conversion by Software Trigger

If user need to scan the fast channel at maximum sampling frequency, the conversion need to be executed by the condition as : multiple sample modules, triggered by software, and triggered repeatedly during the last conversion. An example of continuous scan is as follows :

1. Using Module 0~12 to carry out successive conversion. Set CHSEL (EADC_SCTL0~12[3:0]) as one of channel (EADC_CH0~EADC_CH12). Set EXTSMPT (EADC_SCTL0~12[31:24]) and TRGDLYCNT (EADC_SCTL0~12[15:8]) as 0x00 to minimize the sampling time.
2. Set SWTRG (EADC_SWTRG[12:0]) as 0x1fff to trigger Module 0~12.
3. Wait CURSPL (EADC_STATUS3[4:0]) changes to 0xc which means Module 0~11 have been executed and Module 12 is in the process. Set SWTRG (EADC_SWTRG[12:0]) as 0xfff to trigger Module 0~11 again for next round.
4. Wait CURSPL (EADC_STATUS3[4:0]) changes to 0x1, set SWTRG (EADC_SWTRG[12:0]) as 0x1000 to trigger Module 12.
5. Repeat Step 3~4 to continue the conversion.

6.16.5.6 ADC Sample Module End of Conversion Interrupt Operation

There are 4 ADC interrupts ADINT0~3, and each of these interrupts has its own interrupt vector address and can be configured to set multiple sample module EOC pulse (sample module 0~12, End of conversion pulses) as its interrupt trigger source. Figure 6.16-7 shows the control logic of interrupts. Take ADINT0 as an example, when ADCIEN0 (EADC_CTL[2]) = 1 and SPLIEn (EADC_INTSRC0[n]) = 1 (n=0~12), the specific module EOC (End of conversion) pulses will set flag ADIF0 (EADC_STATUS[0]) as 1 and interrupt (ADINT0) will be asserted either.

The interrupt pulses (ADINT0/1) are generated whenever the specific sample module ADC EOC pulse is generated. It also can be the sample module conversion trigger sources, and user can use it to do the ADC continuous scan conversion.

The example of continuous scan triggered by interrupt is as follows :

1. If ADC sample module 2 EOC2 pulse is selected as ADINT0 interrupt trigger SPLIE2 (EADC_INTSRC0[2]) = 1 and ADINT0 is selected as sample module 0, 1, 2 hardware conversion trigger.
2. Set software trigger SWTRG2 (EADC_SWTRG[2]) to 1 to start a sample module 2 ADC conversion, after the conversion completes, it generates an EOC2 pulse signal and ADINT0

interrupt pulse at end of sample module 2 ADC conversion, ADINT0 interrupt pulse will trigger the sample module 0, 1, 2 to start the ADC conversions.

3. ADINT0 interrupt pulse repeats to trigger sample module 0, 1, 2 ADC conversions automatically.
4. Clear TRGSEL (EADC_SCTL2[20:16]) to 0 to disable sample module 2 ADINT0 interrupt pulse hardware trigger, if needs to stop the continuous scan.

Note: Because the system costs 3 ADC_CLK to trigger next module by interrupt pulse, the average conversion cycles of continuous scan triggered by interrupt is 17 ADC_CLK.

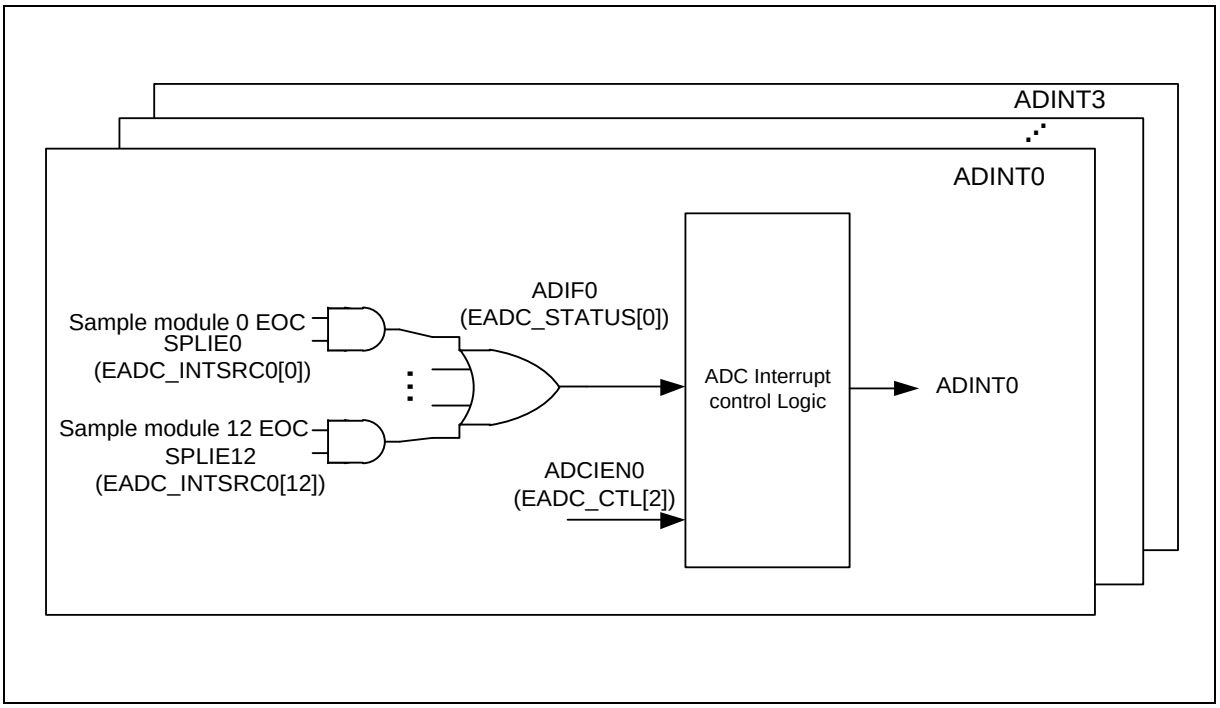


Figure 6.16-7 Specific Sample Module ADC EOC Signal for ADINT0~3 Interrupt

6.16.5.7 ADC Trigger by Timer Trigger and External Pin EADC0_ST

There are 4 Timer trigger source and an external pin EADC0_ST which can configure sample module 0~12 to trigger ADC start when Timer overflow occurs.

6.16.5.8 ADC Start Synchronous with PWM Trigger

Besides user start, ADINT0/1 interrupt pulse, external pin EADC0_ST and Timer0~3 overflow pulse to start ADC conversion, this device has new feature to allow PWM channels to trigger the ADC start. User may configure PWM trigger types: rising, falling PWM edge or center point of PWM (center-aligned mode only) to trigger ADC start. The device also allow user to configure the amount of delay prior to ADC start after hardware detected the external trigger. User can configure the trigger delay time by setting TRGDLYCNT (EADC_SCTLn[15:8], n=0~12) and TRGDLYDIV (EADC_SCTLn[7:6], n=0~12). Figure 6.16-8 shows the programmable delay time for PWM-triggered ADC start conversion. Figure 6.16-9 shows the programmable delay time for other trigger source.

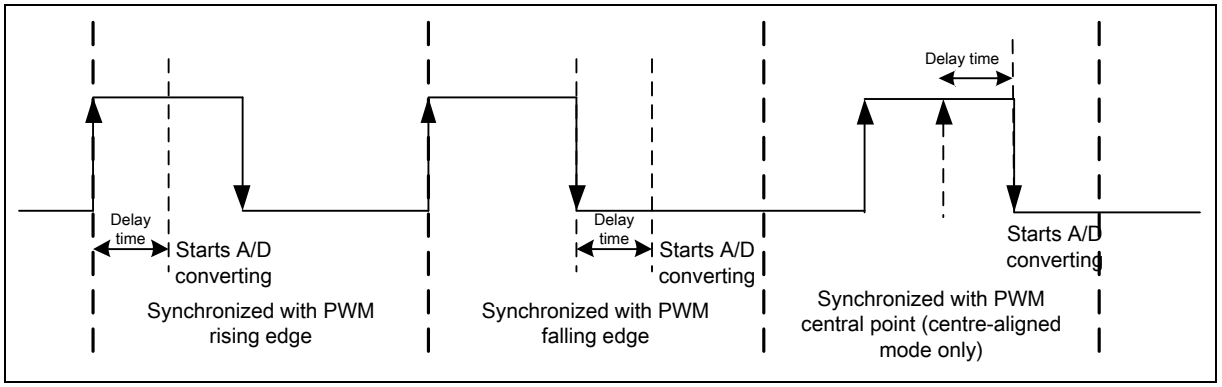


Figure 6.16-8 PWM-triggered ADC Start Conversion

The Figure 6.16-9 shows the programmable delay time for other trigger source.

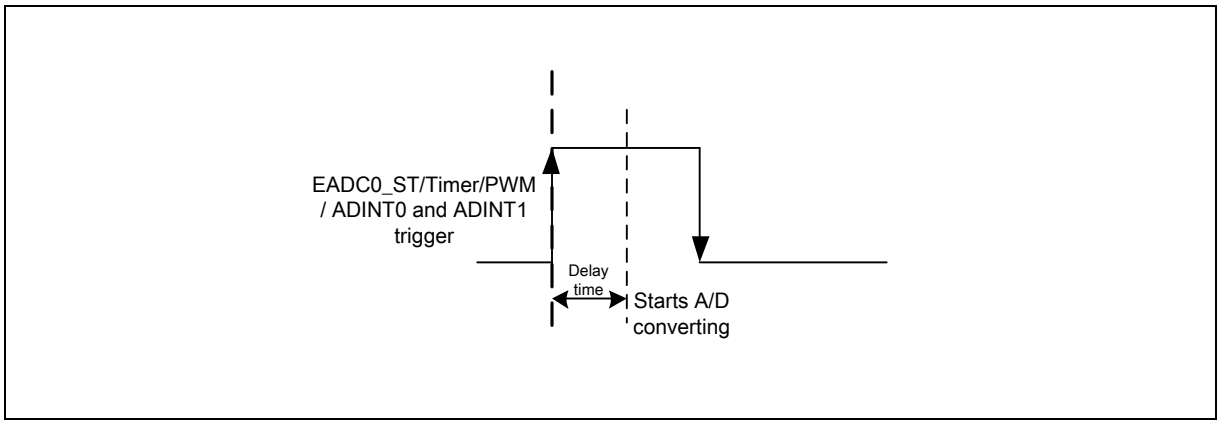


Figure 6.16-9 External triggered ADC Start Conversion

6.16.5.9 ADC Conversion Time and External trigger

The ADC converter sample the analog input when ADC conversion start delay time (Td) has passed after SWTRGn (EADC_SWTRG[n], n=0~12) is set to 1, then start conversion. Due to ADC clock is generated by PCLK divided by (EADCDIV(CLKDIV[23:16])+1), the maximum delay time from user write SWTRGn to ADC start sampling analog input time is two ADC clock cycles. The start delay time is shown in Figure 6.16-10.

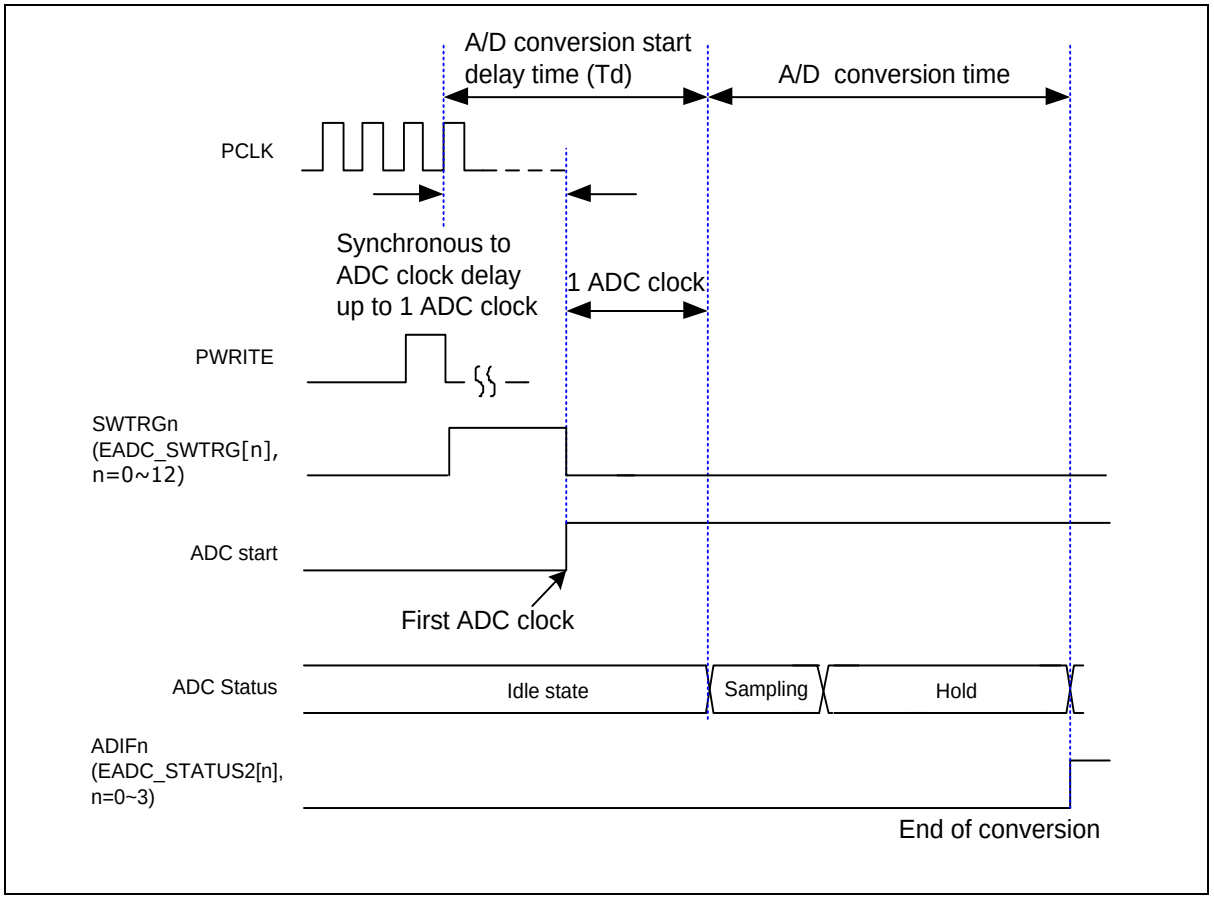


Figure 6.16-10 Conversion Start Delay Timing Diagram

ADC conversion can be triggered by external pin EADC0_ST request. Setting the TRGSEL (EADC_SCTLn[20:16], n=0~12) to 0x01 is to select external trigger input from the EADC0_ST pin. User can set EXTFEN (EADC_SCTLn[5], n=0~12) and EXTREN (EADC_SCTLn[4], n=0~12) to enable pin EADC0_ST trigger condition is falling or rising edge. There is a de-bounce circuit to detect falling or rising edge. If rising edge trigger condition is selected, the low state must be kept at least 2 PCLK cycles and the following high state must be kept at least 3 PCLK cycles. If falling edge trigger condition is selected, the high state must be kept at least 2 PCLK cycles and the following low state must be kept at least 3 PCLK cycles. Pulse that is shorter than this specification will be ignored. The external trigger timing is shown in Figure 6.16-11

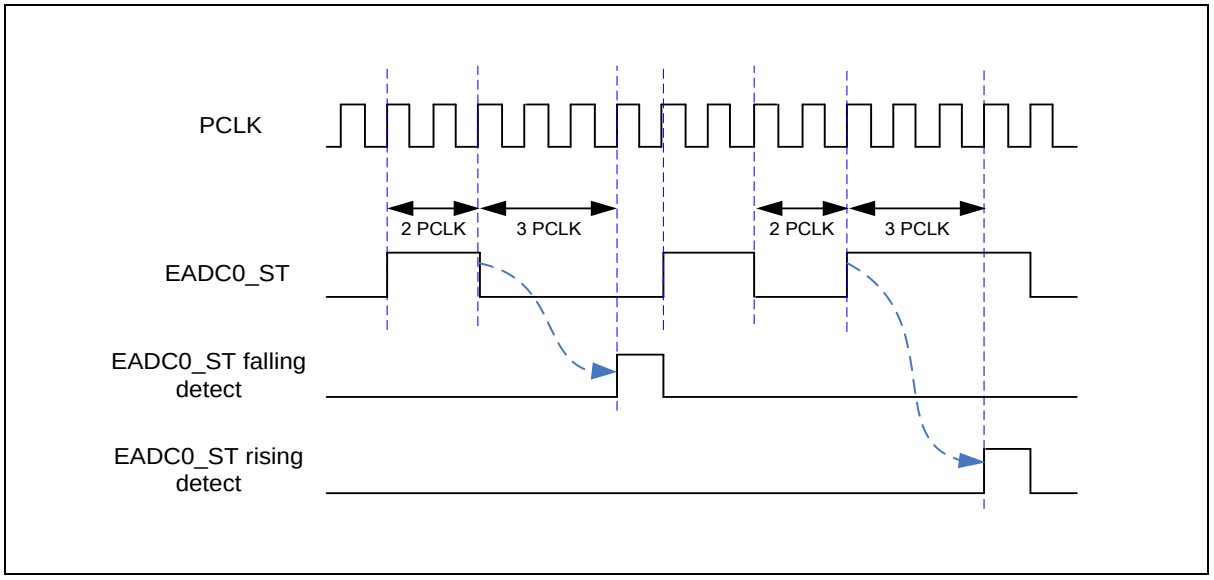
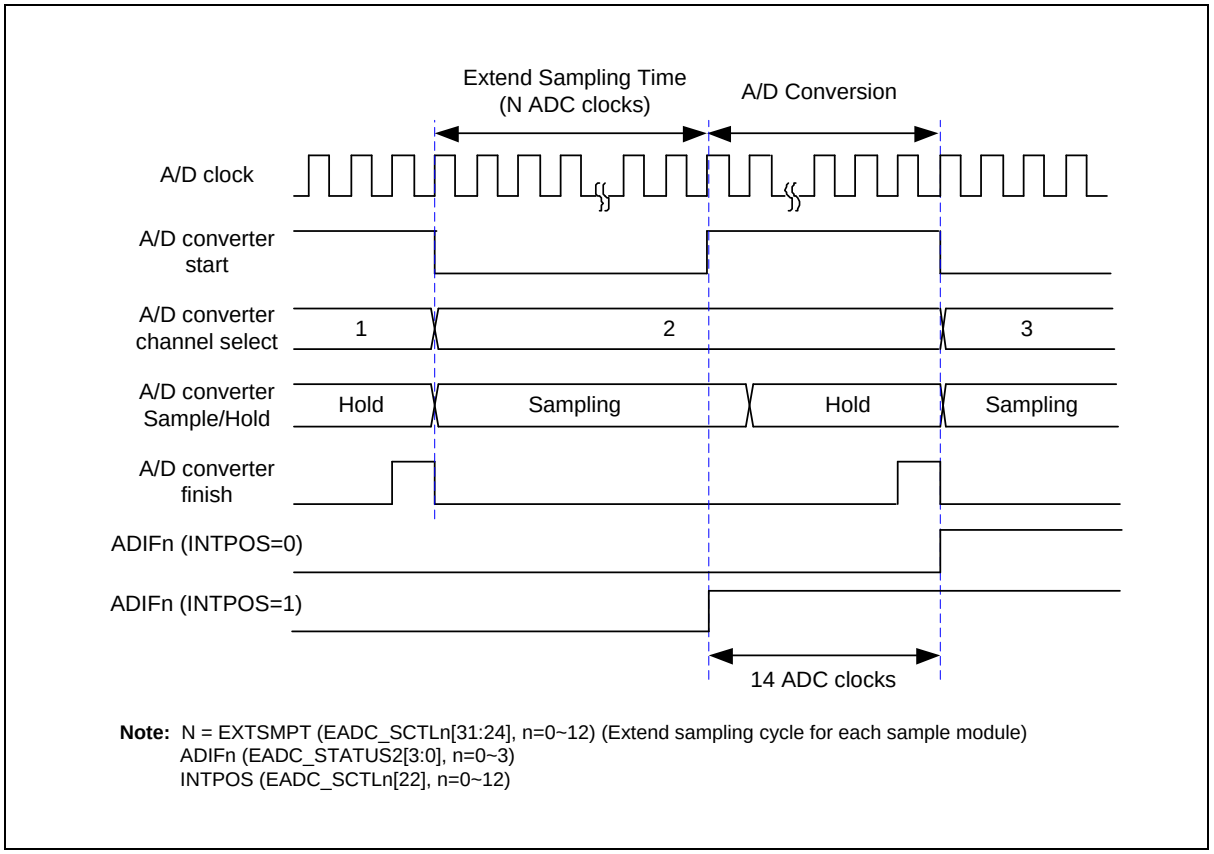


Figure 6.16-11 EADC0_ST De-bounce Timing Diagram

6.16.5.10 ADC Extend Sampling Time

When ADC operation at high ADC clock rate, the sampling time of analog input voltage may not enough if the analog channel has heavy loading to cause fully charge time is longer. User can set extend sampling time by writing EXTSMPT (EADC_SCTLn[31:24], n=0~12) for each sample module. The ADC extend sampling time is present between ADC controller judge which channel to be converting and ADC start to conversion. The range of extend sampling time is from 0 ~255 ADC clock. The extended sampling time is shown in Figure 6.16-12.



Note: N = EXTSMPT (EADC_SCTLn[31:24], n=0~12) (Extend sampling cycle for each sample module)
ADIFn (EADC_STATUS2[3:0], n=0~3)
INTPOS (EADC_SCTLn[22], n=0~12)

Figure 6.16-12 ADC Extend Sampling Timing Diagram

6.16.5.11 Conversion Result Monitor by Compare Mode

The ADC controller provides four sets of compare registers EADC_CMP0 ~ EADC_CMP3 to monitor a maximum of four specified sample module 0~12 conversion results from ADC conversion module, as shown in the Figure 6.16-13. User can select which sample module result to be monitored by set CMPSPL (EADC_CMPn[7:3], n =0~3) and CMPCOND (EADC_CMPn[2], n =0~3) is used to check conversion result is less than specify value or greater than (equal to) value specified in CMPDAT (EADC_CMPn[27:16] , n =0~3). When the conversion of the sample module specified by CMPSPL (EADC_CMPn[7:3], n =0~3) is completed, the comparing action will be triggered one time automatically. When the compare result meets the compare condition, the internal compare match counter will increase 1. If the compare result does not meet the condition, the compare match counter will reset to 0. When counter value reach the setting of (CMPMCNT (EADC_CMPn[11:8])+1, n =0~3) then ADCMPFn (EADC_STATUS2[7:4], n =0~3) bit will be set to 1, if ADCMPIE (EADC_CMPn[1] , n =0~3) is set then an ADINT3 interrupt request is generated. User can use it to monitor the external analog input pin voltage transition. Detailed logics diagram is shown in Figure 6.16-13.

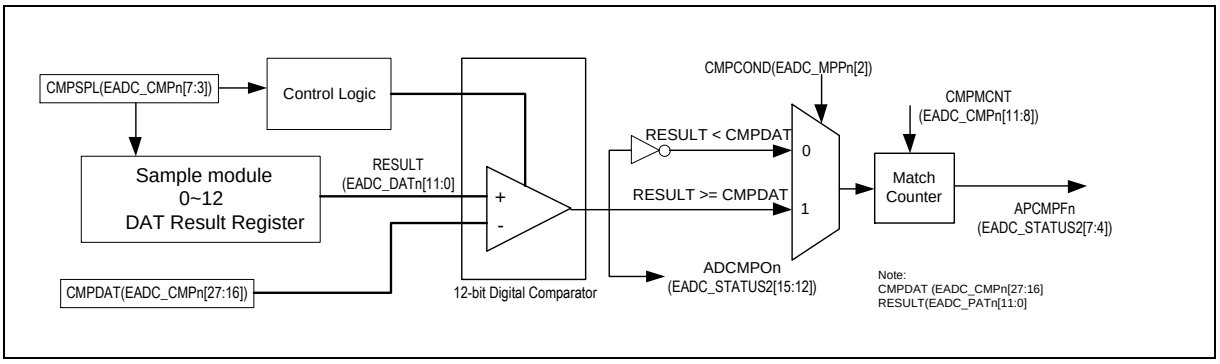


Figure 6.16-13 ADC Conversion Result Monitor Logics Diagram

The ADC controller supports a window compare mode. User can set CMPWEN (EADC_CMP0[15]/ EADC_CMP2[15]) to enable this function. If user enable this function, ADCMPF0 (EADC_STATUS2[4]) will be set when both EADC_CMP0 and EADC_CMP1 compared condition matched. ADCMPF2 (EADC_STATUS2[6]) will be set when both EADC_CMP2 and EADC_CMP3 compared condition matched.

6.16.5.12 Double Buffer Mode

The ADC controller supports a double buffer mode in sample module 0~3. If user enable DBMEN (EADC_SCTLn[23], n=0~3), the double buffer mode will enable. In double buffer mode, after first time ADC convert finish, the VALID (EADC_DATn[17], n=0~3) will set to high, but VALID (EADC_DDATn[17], n=0~3) will keep low. And the second time ADC converts finish, VALID (EADC_DDATn[17],n=0~3) will set to high either, Then, user can get the ADC results from EADC_DATn and EADC_DDATn register.

6.16.5.13 PDMA request

The ADC controller supports PDMA. User can enable PDMAEN (EADC_CTL[11]) and configure PDMA channel's source address as EADC_CURDAT (EADC_BA+0x4C). After enable PDMAEN and PDMA channel enable, if any VALID (EADC_DATn[17],n=0~12) is high, ADC controller will send request to PDMA and PDMA will read EADC_CURDAT to get result. The EADC_CURDAT register is a shadow register of highest priority EADC_DAT register. The lower number sample

module is higher priority. After PDMA read EADC_CURDAT register, the VAILD of the shadow EADC_DAT register will be automatically cleared.

6.16.5.14 Interrupt Sources

The ADC converter generates ADIFn (EADC_STATUS2[3:0], n=0~3) at the start of conversion or the end of conversion decide by INTPOS (EADC_SCTLn[22], n=0~12). If ADCIENn (EADC_CTL[5:2], n=0~3) is set then conversion end interrupt request ADINTn (n=0~3) is generated. The controller of interrupts is shown as Figure 6.16-14

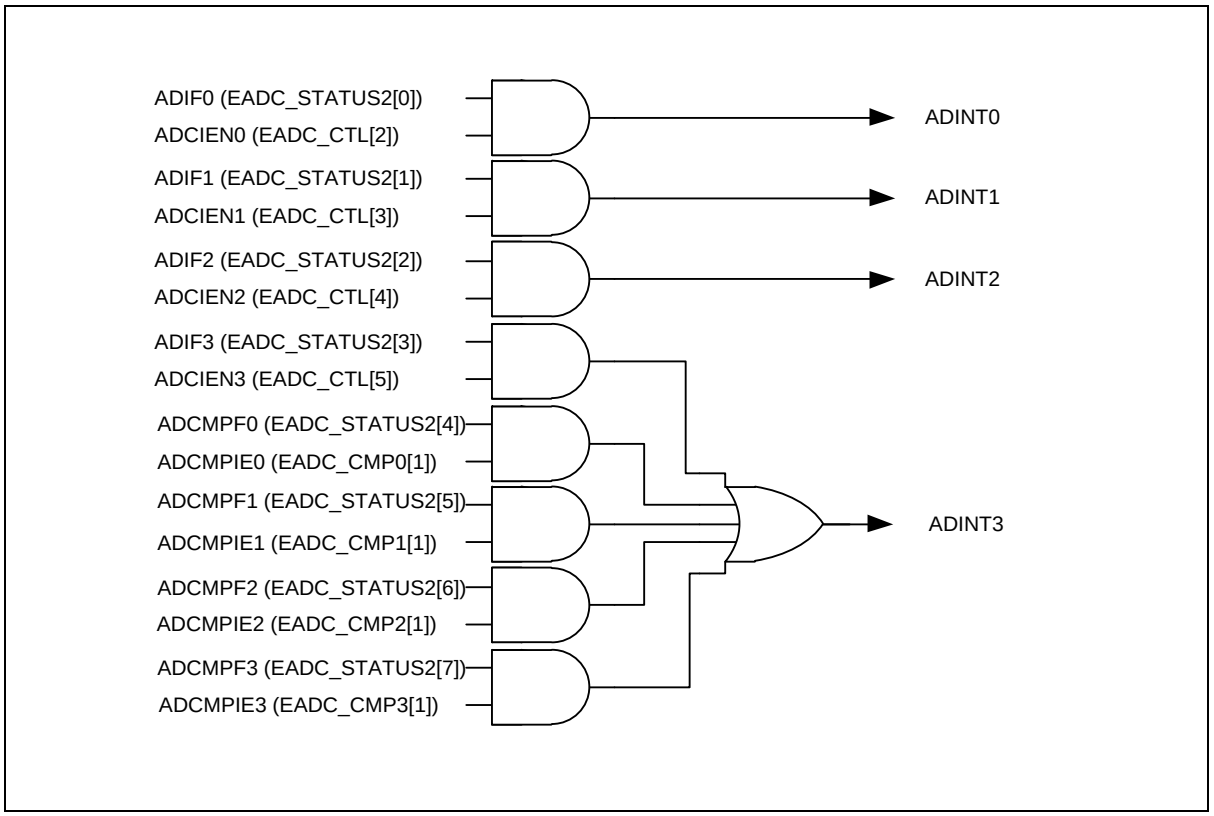


Figure 6.16-14 ADC Controller Interrupts

6.16.5.15 ADC Power Management and Calibration

There are three kind of power saving modes, including ADC Deep Power-down, ADC Power-down mode and ADC Standby mode. User can configure PWDMOD (EADC_PWRM[3:2]) to select which power saving mode EADC would enter when ADCEN (EADC_CTL[0]) is set as 0.

The difference of these power down mode is shown as Table 6.16.5-2. Because the internal LDO will be shut down in Deep Power-down and Power-down mode, EADC needs to take extra time to resume. The interval of time to resume is set by LDOSUT (EADC_PWRM[19:8]) which must be longer than 20 us. As for the Standby mode, LDO will keep enable and start-up time is unnecessary.

Power Supplies	Deep Power-Down	Power-Down	Standby
Internal LDO	Disable	Disable	Enable
Internal power switch	Disable	Enable	Enable

Table 6.16.5-2 EADC Power Saving Mode

When EADC is activated by setting ADCEN(EADC_CTL[0]) to 1, the startup sequence will execute automatically. After start up sequence finished, PWUPRDY (EADC_PWRM[0]) will be set to 1 by HW which means ready to convert. ADCEN (EADC_CTL[0]) must be kept at 1 until PWUPRDY (EADC_PWRM[0]) is set to 1 during the startup sequence. Changing ADCEN (EADC_CTL[0]) arbitrarily at start up sequence will cause EADC function failure.

The conversion results of ADC will be more accurate with calibration. User may set PWUCALEN (EADC_PWRM[1]) as 1 to carry out calibration at start up. An example about start up with calibration is shown as Figure 6.16-15.

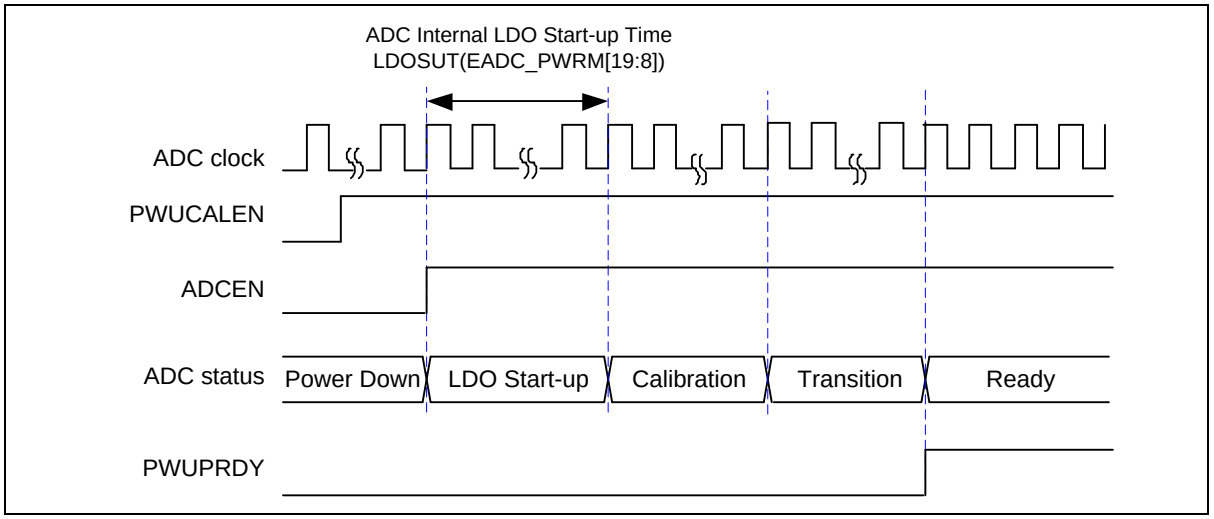


Figure 6.16-15 ADC start up sequence with calibration

6.16.5.16 Minimum ADC Sampling Time

The Figure 6.16-16 shows the (simplified) equivalent circuit of the sample and hold input network, where C_S is the storage capacitor, R_S is the resistance of the sampling switch and R_I is the output impedance of the signal source (V_I). For minimum conversion cycles, duration of the sampling phase is, approximately, $1.5/f_{ADC\_CLK}$. C_S must be charged in that phase, and it must be ensured that the voltage at its terminals becomes sufficiently near V_I. To guarantee this, R_I may not take arbitrarily large values.

Table 6.16.5-3 shows the minimum sampling time relative to different output impedance (R_I) of the signal source. The user must set appropriate extend sampling time by register EXTSMPT (EADC_SCTLx[31:24], x = 0~12). The following describes how to set the appropriate ADC sampling time extension.

Minimum sampling time < (1.5 + EXTSMPT) * (1 / f_{ADC_CLK})

Where f_{ADC_CLK} is the frequency of ADC_CLK.

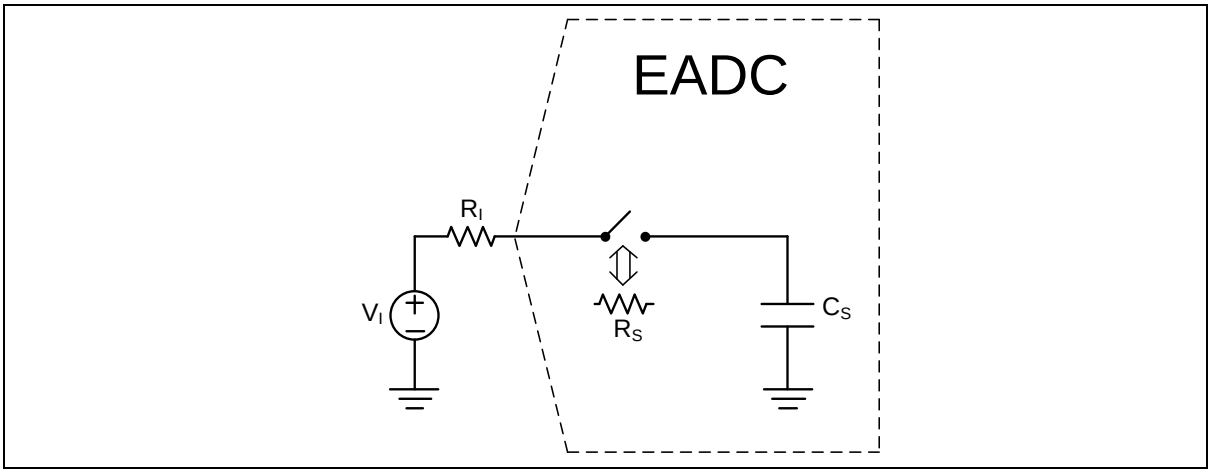


Figure 6.16-16 Model of the sampling network

R_I (kohm)	Minimum sampling time (ns)
0	43
0.05	46
0.1	50
0.2	56
0.5	74
1	105
5	348
10	651
20	1256
50	3083
100	6193

Table 6.16.5-3 EADC minimum sampling time

6.16.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
EADC Base Address: **EADC_BA = 0x4004_3000**				
EADC_DAT0	EADC_BA+0x00	R	ADC Data Register 0 for Sample Module 0	0x0000_0000
EADC_DAT1	EADC_BA+0x04	R	ADC Data Register 1 for Sample Module 1	0x0000_0000
EADC_DAT2	EADC_BA+0x08	R	ADC Data Register 2 for Sample Module 2	0x0000_0000
EADC_DAT3	EADC_BA+0x0C	R	ADC Data Register 3 for Sample Module 3	0x0000_0000
EADC_DAT4	EADC_BA+0x10	R	ADC Data Register 4 for Sample Module 4	0x0000_0000
EADC_DAT5	EADC_BA+0x14	R	ADC Data Register 5 for Sample Module 5	0x0000_0000
EADC_DAT6	EADC_BA+0x18	R	ADC Data Register 6 for Sample Module 6	0x0000_0000
EADC_DAT7	EADC_BA+0x1C	R	ADC Data Register 7 for Sample Module 7	0x0000_0000
EADC_DAT8	EADC_BA+0x20	R	ADC Data Register 8 for Sample Module 8	0x0000_0000
EADC_DAT9	EADC_BA+0x24	R	ADC Data Register 9 for Sample Module 9	0x0000_0000
EADC_DAT10	EADC_BA+0x28	R	ADC Data Register 10 for Sample Module 10	0x0000_0000
EADC_DAT11	EADC_BA+0x2C	R	ADC Data Register 11 for Sample Module 11	0x0000_0000
EADC_DAT12	EADC_BA+0x30	R	ADC Data Register 12 for Sample Module 12	0x0000_0000
EADC_CURDAT	EADC_BA+0x4C	R	ADC PDMA Current Transfer Data Register	0x0000_0000
EADC_CTL	EADC_BA+0x50	R/W	ADC Control Register	0x0000_00C0
EADC_SWTRG	EADC_BA+0x54	W	ADC Sample Module Software Start Register	0x0000_0000
EADC_PENDSTS	EADC_BA+0x58	R/W	ADC Start of Conversion Pending Flag Register	0x0000_0000
EADC_OVSTS	EADC_BA+0x5C	R/W	ADC Sample Module Start of Conversion Overrun Flag Register	0x0000_0000
EADC_SCTL0	EADC_BA+0x80	R/W	ADC Sample Module 0 Control Register	0x0000_0000
EADC_SCTL1	EADC_BA+0x84	R/W	ADC Sample Module 1 Control Register	0x0000_0000
EADC_SCTL2	EADC_BA+0x88	R/W	ADC Sample Module 2 Control Register	0x0000_0000
EADC_SCTL3	EADC_BA+0x8C	R/W	ADC Sample Module 3 Control Register	0x0000_0000
EADC_SCTL4	EADC_BA+0x90	R/W	ADC Sample Module 4 Control Register	0x0000_0000
EADC_SCTL5	EADC_BA+0x94	R/W	ADC Sample Module 5 Control Register	0x0000_0000
EADC_SCTL6	EADC_BA+0x98	R/W	ADC Sample Module 6 Control Register	0x0000_0000
EADC_SCTL7	EADC_BA+0x9C	R/W	ADC Sample Module 7 Control Register	0x0000_0000

Register	Offset	R/W	Description	Reset Value
EADC Base Address: **EADC_BA = 0x4004_3000**				
EADC_SCTL8	EADC_BA+0xA0	R/W	ADC Sample Module 8 Control Register	0x0000_0000
EADC_SCTL9	EADC_BA+0xA4	R/W	ADC Sample Module 9 Control Register	0x0000_0000
EADC_SCTL10	EADC_BA+0xA8	R/W	ADC Sample Module 10 Control Register	0x0000_0000
EADC_SCTL11	EADC_BA+0xAC	R/W	ADC Sample Module 11 Control Register	0x0000_0000
EADC_SCTL12	EADC_BA+0xB0	R/W	ADC Sample Module 12 Control Register	0x0000_0000
EADC_INTSRC0	EADC_BA+0xD0	R/W	ADC interrupt 0 Source Enable Control Register.	0x0000_0000
EADC_INTSRC1	EADC_BA+0xD4	R/W	ADC interrupt 1 Source Enable Control Register.	0x0000_0000
EADC_INTSRC2	EADC_BA+0xD8	R/W	ADC interrupt 2 Source Enable Control Register.	0x0000_0000
EADC_INTSRC3	EADC_BA+0xDC	R/W	ADC interrupt 3 Source Enable Control Register.	0x0000_0000
EADC_CMP0	EADC_BA+0xE0	R/W	ADC Result Compare Register 0	0x0000_0000
EADC_CMP1	EADC_BA+0xE4	R/W	ADC Result Compare Register 1	0x0000_0000
EADC_CMP2	EADC_BA+0xE8	R/W	ADC Result Compare Register 2	0x0000_0000
EADC_CMP3	EADC_BA+0xEC	R/W	ADC Result Compare Register 3	0x0000_0000
EADC_STATUS0	EADC_BA+0xF0	R	ADC Status Register 0	0x0000_0000
EADC_STATUS2	EADC_BA+0xF8	R/W	ADC Status Register 2	0x000F_0000
EADC_STATUS3	EADC_BA+0xFC	R	ADC Status Register 3	0x0000_001F
EADC_DDAT0	EADC_BA+0x100	R	ADC Double Data Register 0 for Sample Module 0	0x0000_0000
EADC_DDAT1	EADC_BA+0x104	R	ADC Double Data Register 1 for Sample Module 1	0x0000_0000
EADC_DDAT2	EADC_BA+0x108	R	ADC Double Data Register 2 for Sample Module 2	0x0000_0000
EADC_DDAT3	EADC_BA+0x10C	R	ADC Double Data Register 3 for Sample Module 3	0x0000_0000
EADC_PWRM	EADC_BA+0x110	R/W	ADC Power Management Register	0x0006_E012
EADC_CHSPC	EADC_BA+0x200	R/W	ADC Channel Switch Presetting Control Register	0x0000_0000

Note:

1. Any register not listed here is reserved and must not be written. The result of a read operation on these bits is undefined.
2. The reserved register fields that listed in register description must be written to their reset value. Writing reserved fields with other than reset values may produce undefined results.

6.16.7 Register Description

ADC Data Registers (EADC_DAT0~ EADC_DAT12)

Register	Offset	R/W	Description	Reset Value
EADC_DAT0	EADC_BA+0x00	R	ADC Data Register 0 for Sample Module 0	0x0000_0000
EADC_DAT1	EADC_BA+0x04	R	ADC Data Register 1 for Sample Module 1	0x0000_0000
EADC_DAT2	EADC_BA+0x08	R	ADC Data Register 2 for Sample Module 2	0x0000_0000
EADC_DAT3	EADC_BA+0x0C	R	ADC Data Register 3 for Sample Module 3	0x0000_0000
EADC_DAT4	EADC_BA+0x10	R	ADC Data Register 4 for Sample Module 4	0x0000_0000
EADC_DAT5	EADC_BA+0x14	R	ADC Data Register 5 for Sample Module 5	0x0000_0000
EADC_DAT6	EADC_BA+0x18	R	ADC Data Register 6 for Sample Module 6	0x0000_0000
EADC_DAT7	EADC_BA+0x1C	R	ADC Data Register 7 for Sample Module 7	0x0000_0000
EADC_DAT8	EADC_BA+0x20	R	ADC Data Register 8 for Sample Module 8	0x0000_0000
EADC_DAT9	EADC_BA+0x24	R	ADC Data Register 9 for Sample Module 9	0x0000_0000
EADC_DAT10	EADC_BA+0x28	R	ADC Data Register 10 for Sample Module 10	0x0000_0000
EADC_DAT11	EADC_BA+0x2C	R	ADC Data Register 11 for Sample Module 11	0x0000_0000
EADC_DAT12	EADC_BA+0x30	R	ADC Data Register 12 for Sample Module 12	0x0000_0000

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

Bits	Description	
[31:18]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[17]	**VALID**	**Valid Flag** This bit is set to 1 when corresponding sample module channel analog input conversion is completed and cleared by hardware after EADC_DAT register is read. 0 = Data in RESULT[11:0] bits is not valid. 1 = Data in RESULT[11:0] bits is valid.

[16]	**OV**	**Overrun Flag** If converted data in RESULT[11:0] has not been read before new conversion result is loaded to this register, OV is set to 1. 0 = Data in RESULT[11:0] is recent conversion result. 1 = Data in RESULT[11:0] is overwrite. **Note:** It is cleared by hardware after EADC_DAT register is read.
[15:0]	**RESULT**	**ADC Conversion Result** This field contains 12 bits conversion result. When DMOF (EADC_CTL[9]) is set to 0, 12-bit ADC conversion result with unsigned format will be filled in RESULT[11:0] and zero will be filled in RESULT[15:12]. When DMOF (EADC_CTL[9]) set to 1, 12-bit ADC conversion result with 2'complement format will be filled in RESULT[11:0] and signed bits to will be filled in RESULT[15:12].

ADC PDMA Current Transfer Data Register (EADC_CURDAT)

Register	Offset	R/W	Description	Reset Value
EADC_CURDAT	EADC_BA+0x4C	R	ADC PDMA Current Transfer Data Register	0x0000_0000

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

Bits	Description	
[31:13]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[12:0]	**CURDAT**	**ADC PDMA Current Transfer Data Register (Read Only)** This register is a shadow register of EADC_DATn (n=0~12) for PDMA support.

ADC Control Register (EADC_CTL)

Register	Offset	R/W	Description	Reset Value
EADC_CTL	EADC_BA+0x50	R/W	ADC Control Register	0x0000_00C0

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				PDMAEN	Reserved	DMOF	Reserved
7	6	5	4	3	2	1	0
RESSEL		ADCIEN3	ADCIEN2	ADCIEN1	ADCIEN0	ADCRST	ADCEN

Bits	Description	
[31:12]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[11]	**PDMAEN**	**PDMA Transfer Enable Bit** When ADC conversion is completed, the converted data is loaded into EADC_DATn (n: 0 ~ 12) register, user can enable this bit to generate a PDMA data transfer request. 0 = PDMA data transfer Disabled. 1 = PDMA data transfer Enabled. **Note:** When set this bit field to 1, user must set ADCIENn (EADC_CTL[5:2], n=0~3) = 0 to disable interrupt.
[10]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[9]	**DMOF**	**ADC Differential Input Mode Output Format** 0 = ADC conversion result will be filled in RESULT (EADC_DATn[15:0] , n= 0 ~12) with unsigned format. 1 = ADC conversion result will be filled in RESULT (EADC_DATn[15:0] , n= 0 ~12) with 2'complement format.
[8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7:6]	**RESSEL**	**Resolution Selection** 00 = 6-bit. ADC result will put at RESULT (EADC_ DATn [5:0]). 01 = 8-bit. ADC result will put at RESULT (EADC_ DATn [7:0]). 10 = 10-bit. ADC result will put at RESULT (EADC_ DATn [9:0]). 11 = 12-bit. ADC result will put at RESULT (EADC_ DATn [11:0]).
[5]	**ADCIEN3**	**Specific Sample Module ADC ADINT3 Interrupt Enable Bit** The ADC converter generates a conversion end ADIF3 (EADC_STATUS2[3]) upon the end of specific sample module ADC conversion. If ADCIEN3 bit is set then conversion end interrupt request ADINT3 is generated. 0 = Specific sample module ADC ADINT3 interrupt function Disabled. 1 = Specific sample module ADC ADINT3 interrupt function Enabled.

Bits	Description	
[4]	**ADCIEN2**	**Specific Sample Module ADC ADINT2 Interrupt Enable Bit** The ADC converter generates a conversion end ADIF2 (EADC_STATUS2[2]) upon the end of specific sample module ADC conversion. If ADCIEN2 bit is set then conversion end interrupt request ADINT2 is generated. 0 = Specific sample module ADC ADINT2 interrupt function Disabled. 1 = Specific sample module ADC ADINT2 interrupt function Enabled.
[3]	**ADCIEN1**	**Specific Sample Module ADC ADINT1 Interrupt Enable Bit** The ADC converter generates a conversion end ADIF1 (EADC_STATUS2[1]) upon the end of specific sample module ADC conversion. If ADCIEN1 bit is set then conversion end interrupt request ADINT1 is generated. 0 = Specific sample module ADC ADINT1 interrupt function Disabled. 1 = Specific sample module ADC ADINT1 interrupt function Enabled.
[2]	**ADCIEN0**	**Specific Sample Module ADC ADINT0 Interrupt Enable Bit** The ADC converter generates a conversion end ADIF0 (EADC_STATUS2[0]) upon the end of specific sample module ADC conversion. If ADCIEN0 bit is set then conversion end interrupt request ADINT0 is generated. 0 = Specific sample module ADC ADINT0 interrupt function Disabled. 1 = Specific sample module ADC ADINT0 interrupt function Enabled.
[1]	**ADCRST**	**ADC Converter Control Circuits Reset** 0 = No effect. 1 = Cause ADC control circuits reset to initial state, but not change the ADC registers value. **Note:** ADCRST bit remains 1 during ADC reset, when ADC reset end, the ADCRST bit is automatically cleared to 0.
[0]	**ADCEN**	**ADC Converter Enable Bit** 0 = Disabled EADC. 1 = Enabled EADC. **Note:** Before starting ADC conversion function, this bit should be set to 1. Clear it to 0 to disable ADC converter analog circuit power consumption.

ADC Sample Module Software Start Register (EADC_SWTRG)

Register	Offset	R/W	Description	Reset Value
EADC_SWTRG	EADC_BA+0x54	W	ADC Sample Module Software Start Register	0x0000_0000

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

Bits	Description	
[31:13]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[12:0]	**SWTRG**	**ADC Sample Module 0~12 Software Force to Start ADC Conversion** 0 = No effect. 1 = Cause an ADC conversion when the priority is given to sample module. **Note:** After write this register to start ADC conversion, the EADC_PENDSTS register will show which sample module will conversion. If user want to disable the conversion of the sample module, user can write EADC_PENDSTS register to clear it.

ADC Sample Module Start of Conversion Pending Flag Register (EADC_PENDSTS)

Register	Offset	R/W	Description	Reset Value
EADC_PENDSTS	EADC_BA+0x58	R/W	ADC Start of Conversion Pending 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			**STPF**				
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
STPF							

Bits	Description	
[31:13]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[12:0]	**STPF**	**ADC Sample Module 0~12 Start of Conversion Pending Flag** **Read:** 0 = There is no pending conversion for sample module. 1 = Sample module ADC start of conversion is pending. **Write:** 1 = clear pending flag & cancel the conversion for sample module. **Note:** This bit remains 1 during pending state, when the respective ADC conversion is end, the STPFn (n=0~12) bit is automatically cleared to 0.

ADC Sample Module Overrun Flag Register (EADC_OVSTS)

Register	Offset	R/W	Description	Reset Value
EADC_OVSTS	EADC_BA+0x5C	R/W	ADC Sample Module Start of Conversion Overrun 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			**SPOVF**				
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SPOVF							

Bits	Description	
[31:13]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[12:0]	**SPOVF**	**ADC SAMPLE0~12 Overrun Flag** 0 = No sample module event overrun. 1 = Indicates a new sample module event is generated while an old one event is pending. **Note:** This bit is cleared by writing 1 to it.

ADC Sample Module 0~3 Control Registers (EADC_SCTL0~EADC_SCTL3)

Register	Offset	R/W	Description	Reset Value
EADC_SCTL0	EADC_BA+0x80	R/W	ADC Sample Module 0 Control Register	0x0000_0000
EADC_SCTL1	EADC_BA+0x84	R/W	ADC Sample Module 1 Control Register	0x0000_0000
EADC_SCTL2	EADC_BA+0x88	R/W	ADC Sample Module 2 Control Register	0x0000_0000
EADC_SCTL3	EADC_BA+0x8C	R/W	ADC Sample Module 3 Control Register	0x0000_0000

31	30	29	28	27	26	25	24
EXTSMPT							
23	22	21	20	19	18	17	16
DBMEN	INTPOS	Reserved	TRGSEL				
15	14	13	12	11	10	9	8
TRGDLYCNT							
7	6	5	4	3	2	1	0
TRGDLYDIV		EXTFEN	EXTREN	CHSEL			

Bits	Description	
[31:24]	**EXTSMPT**	**ADC Sampling Time Extend** When ADC converting at high conversion rate, the sampling time of analog input voltage may not enough if input channel loading is heavy, user can extend ADC sampling time after trigger source is coming to get enough sampling time. The range of start delay time is from 0~255 ADC clock.
[23]	**DBMEN**	**Double Buffer Mode Enable Bit** 0 = Sample has one sample result register. (default). 1 = Sample has two sample result registers.
[22]	**INTPOS**	**Interrupt Flag Position Select** 0 = Set ADIFn (EADC_STATUS2[n], n=0~3) at ADC end of conversion. 1 = Set ADIFn (EADC_STATUS2[n], n=0~3) at ADC start of conversion.
[21]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

Bits	Description	
[20:16]	**TRGSEL**	**ADC Sample Module Start of Conversion Trigger Source Selection** 0H = Disable trigger. 1H = External trigger from EADC0_ST pin input. 2H = ADC ADINT0 interrupt EOC (End of conversion) pulse trigger. 3H = ADC ADINT1 interrupt EOC (End of conversion) pulse trigger. 4H = Timer0 overflow pulse trigger. 5H = Timer1 overflow pulse trigger. 6H = Timer2 overflow pulse trigger. 7H = Timer3 overflow pulse trigger. 8H = PWM0TG0. 9H = PWM0TG1. AH = PWM0TG2. BH = PWM0TG3. CH = PWM0TG4. DH = PWM0TG5. Others = Reserved. Do not use.
[15:8]	**TRGDLYCNT**	**ADC Sample Module Start of Conversion Trigger Delay Time** Trigger delay time = TRGDLYCNT x ADC_CLK x n (n=1,2,4,16 from TRGDLYDIV setting).
[7:6]	**TRGDLYDIV**	**ADC Sample Module Start of Conversion Trigger Delay Clock Divider Selection** Trigger delay clock frequency: 00 = ADC_CLK/1. 01 = ADC_CLK/2. 10 = ADC_CLK/4. 11 = ADC_CLK/16.
[5]	**EXTFEN**	**ADC External Trigger Falling Edge Enable Bit** 0 = Falling edge Disabled when ADC selects EADC0_ST as trigger source. 1 = Falling edge Enabled when ADC selects EADC0_ST as trigger source.
[4]	**EXTREN**	**ADC External Trigger Rising Edge Enable Bit** 0 = Rising edge Disabled when ADC selects EADC0_ST as trigger source. 1 = Rising edge Enabled when ADC selects EADC0_ST as trigger source.
[3:0]	**CHSEL**	**ADC Sample Module Channel Selection** 00H = EADC0_CH0. 01H = EADC0_CH1. 02H = EADC0_CH2. 03H = EADC0_CH3. 04H = EADC0_CH4. 05H = EADC0_CH5. 06H = EADC0_CH6. 07H = EADC0_CH7. 08H = EADC0_CH8. 09H = EADC0_CH9. 0AH = EADC0_CH10. 0BH = EADC0_CH11. 0CH = EADC0_CH12. Others = Reserved. Do not use.

ADC Sample Module 4~12 Control Registers (EADC_SCTL4~EADC_SCTL12)

Register	Offset	R/W	Description	Reset Value
EADC_SCTL4	EADC_BA+0x90	R/W	ADC Sample Module 4 Control Register	0x0000_0000
EADC_SCTL5	EADC_BA+0x94	R/W	ADC Sample Module 5 Control Register	0x0000_0000
EADC_SCTL6	EADC_BA+0x98	R/W	ADC Sample Module 6 Control Register	0x0000_0000
EADC_SCTL7	EADC_BA+0x9C	R/W	ADC Sample Module 7 Control Register	0x0000_0000
EADC_SCTL8	EADC_BA+0xA0	R/W	ADC Sample Module 8 Control Register	0x0000_0000
EADC_SCTL9	EADC_BA+0xA4	R/W	ADC Sample Module 9 Control Register	0x0000_0000
EADC_SCTL10	EADC_BA+0xA8	R/W	ADC Sample Module 10 Control Register	0x0000_0000
EADC_SCTL11	EADC_BA+0xAC	R/W	ADC Sample Module 11 Control Register	0x0000_0000
EADC_SCTL12	EADC_BA+0xB0	R/W	ADC Sample Module 12 Control Register	0x0000_0000

31	30	29	28	27	26	25	24
EXTSMPT							
23	22	21	20	19	18	17	16
Reserved	INTPOS	Reserved	TRGSEL				
15	14	13	12	11	10	9	8
TRGDLYCNT							
7	6	5	4	3	2	1	0
TRGDLYDIV		EXTFEN	EXTREN	CHSEL			

Bits	Description	
[31:24]	**EXTSMPT**	**ADC Sampling Time Extend** When ADC converting at high conversion rate, the sampling time of analog input voltage may not enough if input channel loading is heavy, SW can extend ADC sampling time after trigger source is coming to get enough sampling time. The range of start delay time is from 0~255 ADC clock.
[23]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[22]	**INTPOS**	**Interrupt Flag Position Select** 0 = Set ADIFn (EADC_STATUS2[n], n=0~3) at ADC end of conversion. 1 = Set ADIFn (EADC_STATUS2[n], n=0~3) at ADC start of conversion.
[21]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

Bits	Description	
[20:16]	**TRGSEL**	**ADC Sample Module Start of Conversion Trigger Source Selection** 0H = Disable trigger. 1H = External trigger from EADC0_ST pin input. 2H = ADC ADINT0 interrupt EOC pulse trigger. 3H = ADC ADINT1 interrupt EOC pulse trigger. 4H = Timer0 overflow pulse trigger. 5H = Timer1 overflow pulse trigger. 6H = Timer2 overflow pulse trigger. 7H = Timer3 overflow pulse trigger. 8H = PWM0TG0. 9H = PWM0TG1. AH = PWM0TG2. BH = PWM0TG3. CH = PWM0TG4. DH = PWM0TG5. Other = Reserved. Do not use.
[15:8]	**TRGDLYCNT**	**ADC Sample Module Start of Conversion Trigger Delay Time** Trigger delay time = TRGDLYCNT x ADC_CLK x n (n=1,2,4,16 from TRGDLYDIV setting).
[7:6]	**TRGDLYDIV**	**ADC Sample Module Start of Conversion Trigger Delay Clock Divider Selection** Trigger delay clock frequency: 00 = ADC_CLK/1. 01 = ADC_CLK/2. 10 = ADC_CLK/4. 11 = ADC_CLK/16.
[5]	**EXTFEN**	**ADC External Trigger Falling Edge Enable Bit** 0 = Falling edge Disabled when ADC selects EADC0_ST as trigger source. 1 = Falling edge Enabled when ADC selects EADC0_ST as trigger source.
[4]	**EXTREN**	**ADC External Trigger Rising Edge Enable Bit** 0 = Rising edge Disabled when ADC selects EADC0_ST as trigger source. 1 = Rising edge Enabled when ADC selects EADC0_ST as trigger source.
[3:0]	**CHSEL**	**ADC Sample Module Channel Selection** 00H = EADC0_CH0. 01H = EADC0_CH1. 02H = EADC0_CH2. 03H = EADC0_CH3. 04H = EADC0_CH4. 05H = EADC0_CH5. 06H = EADC0_CH6. 07H = EADC0_CH7. 08H = EADC0_CH8. 09H = EADC0_CH9. 0AH = EADC0_CH10. 0BH = EADC0_CH11. 0CH = EADC0_CH12. Others = Reserved. Do not use.

ADC Interrupt Source Enable Control Registers (EADC_INTSRC0~EADC_INTSRC3)

Register	Offset	R/W	Description	Reset Value
EADC_INTSRC0	EADC_BA+0xD0	R/W	ADC interrupt 0 Source Enable Control Register.	0x0000_0000
EADC_INTSRC1	EADC_BA+0xD4	R/W	ADC interrupt 1 Source Enable Control Register.	0x0000_0000
EADC_INTSRC2	EADC_BA+0xD8	R/W	ADC interrupt 2 Source Enable Control Register.	0x0000_0000
EADC_INTSRC3	EADC_BA+0xDC	R/W	ADC interrupt 3 Source Enable Control Register.	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved			SPLIE12	SPLIE11	SPLIE10	SPLIE9	SPLIE8
7	6	5	4	3	2	1	0
SPLIE7	SPLIE6	SPLIE5	SPLIE4	SPLIE3	SPLIE2	SPLIE1	SPLIE0

Bits	Description	
[31:13]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[12]	**SPLIE12**	**Sample Module 12 Interrupt Enable Bit** 0 = Sample Module 12 interrupt Disabled. 1 = Sample Module 12 interrupt Enabled.
[11]	**SPLIE11**	**Sample Module 11 Interrupt Enable Bit** 0 = Sample Module 11 interrupt Disabled. 1 = Sample Module 11 interrupt Enabled.
[10]	**SPLIE10**	**Sample Module 10 Interrupt Enable Bit** 0 = Sample Module 10 interrupt Disabled. 1 = Sample Module 10 interrupt Enabled.
[9]	**SPLIE9**	**Sample Module 9 Interrupt Enable Bit** 0 = Sample Module 9 interrupt Disabled. 1 = Sample Module 9 interrupt Enabled.
[8]	**SPLIE8**	**Sample Module 8 Interrupt Enable Bit** 0 = Sample Module 8 interrupt Disabled. 1 = Sample Module 8 interrupt Enabled.
[7]	**SPLIE7**	**Sample Module 7 Interrupt Enable Bit** 0 = Sample Module 7 interrupt Disabled. 1 = Sample Module 7 interrupt Enabled.

Bits	Description	
[6]	**SPLIE6**	**Sample Module 6 Interrupt Enable Bit** 0 = Sample Module 6 interrupt Disabled. 1 = Sample Module 6 interrupt Enabled.
[5]	**SPLIE5**	**Sample Module 5 Interrupt Enable Bit** 0 = Sample Module 5 interrupt Disabled. 1 = Sample Module 5 interrupt Enabled.
[4]	**SPLIE4**	**Sample Module 4 Interrupt Enable Bit** 0 = Sample Module 4 interrupt Disabled. 1 = Sample Module 4 interrupt Enabled.
[3]	**SPLIE3**	**Sample Module 3 Interrupt Enable Bit** 0 = Sample Module 3 interrupt Disabled. 1 = Sample Module 3 interrupt Enabled.
[2]	**SPLIE2**	**Sample Module 2 Interrupt Enable Bit** 0 = Sample Module 2 interrupt Disabled. 1 = Sample Module 2 interrupt Enabled.
[1]	**SPLIE1**	**Sample Module 1 Interrupt Enable Bit** 0 = Sample Module 1 interrupt Disabled. 1 = Sample Module 1 interrupt Enabled.
[0]	**SPLIE0**	**Sample Module 0 Interrupt Enable Bit** 0 = Sample Module 0 interrupt Disabled. 1 = Sample Module 0 interrupt Enabled.

ADC Result Compare Register 0/1/2/3 (EADC_CMP0/1/2/3)

Register	Offset	R/W	Description	Reset Value
EADC_CMP0	EADC_BA+0xE0	R/W	ADC Result Compare Register 0	0x0000_0000
EADC_CMP1	EADC_BA+0xE4	R/W	ADC Result Compare Register 1	0x0000_0000
EADC_CMP2	EADC_BA+0xE8	R/W	ADC Result Compare Register 2	0x0000_0000
EADC_CMP3	EADC_BA+0xEC	R/W	ADC Result Compare Register 3	0x0000_0000

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

Bits	Description	
[31:28]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[27:16]	**CMPDAT**	**Comparison Data** The 12 bits data is used to compare with conversion result of specified sample module. User can use it to monitor the external analog input pin voltage transition without imposing a load on software.
[15]	**CMPWEN**	**Compare Window Mode Enable Bit** 0 = ADCMPF0 (EADC_STATUS2[4]) will be set when EADC_CMP0 compared condition matched. ADCMPF2 (EADC_STATUS2[6]) will be set when EADC_CMP2 compared condition matched 1 = ADCMPF0 (EADC_STATUS2[4]) will be set when both EADC_CMP0 and EADC_CMP1 compared condition matched. ADCMPF2 (EADC_STATUS2[6]) will be set when both EADC_CMP2 and EADC_CMP3 compared condition matched. **Note:** This bit is only present in EADC_CMP0 and EADC_CMP2 register.
[14:12]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[11:8]	**CMPMCNT**	**Compare Match Count** When the specified ADC sample module analog conversion result matches the compare condition defined by CMPCOND (EADC_CMPn[2], n=0~3), the internal match counter will increase 1. If the compare result does not meet the compare condition, the internal compare match counter will reset to 0. When the internal counter reaches the value to (CMPMCNT +1), the ADCMPFn (EADC_STATUS2[7:4], n=0~3) will be set.

Bits	Description	
[7:3]	**CMPSPL**	**Compare Sample Module Selection** 00000 = Sample Module 0 conversion result EADC_DAT0 is selected to be compared. 00001 = Sample Module 1 conversion result EADC_DAT1 is selected to be compared. 00010 = Sample Module 2 conversion result EADC_DAT2 is selected to be compared. 00011 = Sample Module 3 conversion result EADC_DAT3 is selected to be compared. 00100 = Sample Module 4 conversion result EADC_DAT4 is selected to be compared. 00101 = Sample Module 5 conversion result EADC_DAT5 is selected to be compared. 00110 = Sample Module 6 conversion result EADC_DAT6 is selected to be compared. 00111 = Sample Module 7 conversion result EADC_DAT7 is selected to be compared. 01000 = Sample Module 8 conversion result EADC_DAT8 is selected to be compared. 01001 = Sample Module 9 conversion result EADC_DAT9 is selected to be compared. 01010 = Sample Module 10 conversion result EADC_DAT10 is selected to be compared. 01011 = Sample Module 11 conversion result EADC_DAT11 is selected to be compared. 01100 = Sample Module 12 conversion result EADC_DAT12 is selected to be compared. Others = Reserved. Do not use.
[2]	**CMPCOND**	**Compare Condition** 0= Set the compare condition as that when a 12-bit ADC conversion result is less than the 12-bit CMPDAT (EADC_CMPn [27:16]), the internal match counter will increase one. 1= Set the compare condition as that when a 12-bit ADC conversion result is greater or equal to the 12-bit CMPDAT (EADC_CMPn [27:16]), the internal match counter will increase one. **Note:** When the internal counter reaches the value to (CMPMCNT (EADC_CMPn[11:8], n=0~3) +1), the CMPF bit will be set.
[1]	**ADCMPIE**	**ADC Result Compare Interrupt Enable Bit** 0 = Compare function interrupt Disabled. 1 = Compare function interrupt Enabled. If the compare function is enabled and the compare condition matches the setting of CMPCOND (EADC_CMPn[2], n=0~3) and CMPMCNT (EADC_CMPn[11:8], n=0~3), ADCMPFn (EADC_STATUS2[7:4], n=0~3) will be asserted, in the meanwhile, if ADCMPIE is set to 1, a compare interrupt request is generated.
[0]	**ADCMPEN**	**ADC Result Compare Enable Bit** 0 = Compare Disabled. 1 = Compare Enabled. Set this bit to 1 to enable compare CMPDAT (EADC_CMPn[27:16], n=0~3) with specified sample module conversion result when converted data is loaded into EADC_DAT register.

ADC Status Register 0 (EADC_STATUS0)

Register	Offset	R/W	Description	Reset Value
EADC_STATUS0	EADC_BA+0xF0	R	ADC Status Register 0	0x0000_0000

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

Bits	Description	
[31:29]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[28:16]	**OV**	**EADC_DAT0~12 Overrun Flag** It is a mirror to OV bit in sample module ADC result data register EADC_DATn. (n=0~12).
[15:13]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[12:0]	**VALID**	**EADC_DAT0~12 Data Valid Flag** It is a mirror of VALID bit in sample module ADC result data register EADC_DATn. (n=0~12).

ADC Status Register 2 (EADC_STATUS2)

Register	Offset	R/W	Description	Reset Value
EADC_STATUS2	EADC_BA+0xF8	R/W	ADC Status Register 2	0x000F_0000

31	30	29	28	27	26	25	24
Reserved				AOV	AVALID	STOVF	ADOVIF
23	22	21	20	19	18	17	16
BUSY	Reserved		CHANNEL				
15	14	13	12	11	10	9	8
ADCMPO3	ADCMPO2	ADCMPO1	ADCMPO0	ADOVIF3	ADOVIF2	ADOVIF1	ADOVIF0
7	6	5	4	3	2	1	0
ADCMPF3	ADCMPF2	ADCMPF1	ADCMPF0	ADIF3	ADIF2	ADIF1	ADIF0

Bits	Description	
[31:28]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[27]	**AOV**	**for All Sample Module ADC Result Data Register Overrun Flags Check (Read Only)** n=0~12. 0 = None of sample module data register overrun flag Ovn (EADC_DATn[16]) is set to 1. 1 = Any one of sample module data register overrun flag Ovn (EADC_DATn[16]) is set to 1. **Note:** This bit will keep 1 when any Ovn Flag is equal to 1.
[26]	**AVALID**	**for All Sample Module ADC Result Data Register EADC_DAT Data Valid Flag Check(Read Only)** n=0~12. 0 = None of sample module data register valid flag VALIDn (EADC_DATn[17]) is set to 1. 1 = Any one of sample module data register valid flag VALIDn (EADC_DATn[17]) is set to 1. **Note:** This bit will keep 1 when any VALIDn Flag is equal to 1.
[25]	**STOVF**	**for All ADC Sample Module Start of Conversion Overrun Flags Check(Read Only)** n=0~12. 0 = None of sample module event overrun flag SPOVFn (EADC_OVSTS[n]) is set to 1. 1 = Any one of sample module event overrun flag SPOVFn (EADC_OVSTS[n]) is set to 1. **Note:** This bit will keep 1 when any SPOVFn Flag is equal to 1.
[24]	**ADOVIF**	**All ADC Interrupt Flag Overrun Bits Check (Read Only)** n=0~3. 0 = None of ADINT interrupt flag ADOVIFn (EADC_STATUS2[11:8]) is overwritten to 1. 1 = Any one of ADINT interrupt flag ADOVIFn (EADC_STATUS2[11:8]) is overwritten to 1. **Note:** This bit will keep 1 when any ADOVIFn Flag is equal to 1.

Bits	Description	
[23]	**BUSY**	**Busy/Idle(Read Only)** 0 = EADC is in idle state. 1 = EADC is busy at conversion.
[22:21]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[20:16]	**CHANNEL**	**Current Conversion Channel(Read Only)** This filed reflects ADC current conversion channel when BUSY=1. It is read only. 00H = EADC0_CH0. 01H = EADC0_CH1. 02H = EADC0_CH2. 03H = EADC0_CH3. 04H = EADC0_CH4. 05H = EADC0_CH5. 06H = EADC0_CH6. 07H = EADC0_CH7. 08H = EADC0_CH8. 09H = EADC0_CH9. 0AH = EADC0_CH10. 0BH = EADC0_CH11. 0CH = EADC0_CH12.
[15]	**ADCMPO3**	**ADC Compare 3 Output Status(Read Only)** The 12 bits compare3 data CMPDAT3 (EADC_CMP3[27:16]) is used to compare with conversion result of specified sample module. User can use it to monitor the external analog input pin voltage status. 0 = Conversion result in EADC_DAT less than CMPDAT3 setting. 1 = Conversion result in EADC_DAT great than or equal CMPDAT3 setting.
[14]	**ADCMPO2**	**ADC Compare 2 Output Status(Read Only)** The 12 bits compare2 data CMPDAT2 (EADC_CMP2[27:16]) is used to compare with conversion result of specified sample module. User can use it to monitor the external analog input pin voltage status. 0 = Conversion result in EADC_DAT less than CMPDAT2 setting. 1 = Conversion result in EADC_DAT great than or equal CMPDAT2 setting.
[13]	**ADCMPO1**	**ADC Compare 1 Output Status(Read Only)** The 12 bits compare1 data CMPDAT1 (EADC_CMP1[27:16]) is used to compare with conversion result of specified sample module. User can use it to monitor the external analog input pin voltage status. 0 = Conversion result in EADC_DAT less than CMPDAT1 setting. 1 = Conversion result in EADC_DAT great than or equal CMPDAT1 setting.
[12]	**ADCMPO0**	**ADC Compare 0 Output Status(Read Only)** The 12 bits compare0 data CMPDAT0 (EADC_CMP0[27:16]) is used to compare with conversion result of specified sample module. User can use it to monitor the external analog input pin voltage status. 0 = Conversion result in EADC_DAT less than CMPDAT0 setting. 1 = Conversion result in EADC_DAT great than or equal CMPDAT0 setting.

Bits	Description	
[11]	**ADOVIF3**	**ADC ADINT3 Interrupt Flag Overrun** 0 = ADINT3 interrupt flag is not overwritten to 1. 1 = ADINT3 interrupt flag is overwritten to 1. **Note:** This bit is cleared by writing 1 to it.
[10]	**ADOVIF2**	**ADC ADINT2 Interrupt Flag Overrun** 0 = ADINT2 interrupt flag is not overwritten to 1. 1 = ADINT2 interrupt flag is s overwritten to 1. **Note:** This bit is cleared by writing 1 to it.
[9]	**ADOVIF1**	**ADC ADINT1 Interrupt Flag Overrun** 0 = ADINT1 interrupt flag is not overwritten to 1. 1 = ADINT1 interrupt flag is overwritten to 1. **Note:** This bit is cleared by writing 1 to it.
[8]	**ADOVIF0**	**ADC ADINT0 Interrupt Flag Overrun** 0 = ADINT0 interrupt flag is not overwritten to 1. 1 = ADINT0 interrupt flag is overwritten to 1. **Note:** This bit is cleared by writing 1 to it.
[7]	**ADCMPF3**	**ADC Compare 3 Flag** When the specific sample module ADC conversion result meets setting condition in EADC_CMP3 then this bit is set to 1. 0 = Conversion result in EADC_DAT does not meet EADC_CMP3 register setting. 1 = Conversion result in EADC_DAT meets EADC_CMP3 register setting. **Note:** This bit is cleared by writing 1 to it.
[6]	**ADCMPF2**	**ADC Compare 2 Flag** When the specific sample module ADC conversion result meets setting condition in EADC_CMP2 then this bit is set to 1. 0 = Conversion result in EADC_DAT does not meet EADC_CMP2 register setting. 1 = Conversion result in EADC_DAT meets EADC_CMP2 register setting. **Note:** This bit is cleared by writing 1 to it.
[5]	**ADCMPF1**	**ADC Compare 1 Flag** When the specific sample module ADC conversion result meets setting condition in EADC_CMP1 then this bit is set to 1. 0 = Conversion result in EADC_DAT does not meet EADC_CMP1 register setting. 1 = Conversion result in EADC_DAT meets EADC_CMP1 register setting. **Note:** This bit is cleared by writing 1 to it.
[4]	**ADCMPF0**	**ADC Compare 0 Flag** When the specific sample module ADC conversion result meets setting condition in EADC_CMP0 then this bit is set to 1. 0 = Conversion result in EADC_DAT does not meet EADC_CMP0 register setting. 1 = Conversion result in EADC_DAT meets EADC_CMP0 register setting. **Note:** This bit is cleared by writing 1 to it.

Bits	Description	
[3]	**ADIF3**	**ADC ADINT3 Interrupt Flag** 0 = No ADINT3 interrupt pulse received. 1 = ADINT3 interrupt pulse has been received. **Note1:** This bit is cleared by writing 1 to it. **Note2:**This bit indicates whether an ADC conversion of specific sample module has been completed
[2]	**ADIF2**	**ADC ADINT2 Interrupt Flag** 0 = No ADINT2 interrupt pulse received. 1 = ADINT2 interrupt pulse has been received. **Note1:** This bit is cleared by writing 1 to it. **Note2:**This bit indicates whether an ADC conversion of specific sample module has been completed
[1]	**ADIF1**	**ADC ADINT1 Interrupt Flag** 0 = No ADINT1 interrupt pulse received. 1 = ADINT1 interrupt pulse has been received. **Note1:** This bit is cleared by writing 1 to it. **Note2:**This bit indicates whether an ADC conversion of specific sample module has been completed
[0]	**ADIF0**	**ADC ADINT0 Interrupt Flag** 0 = No ADINT0 interrupt pulse received. 1 = ADINT0 interrupt pulse has been received. **Note1:** This bit is cleared by writing 1 to it. **Note2:**This bit indicates whether an ADC conversion of specific sample module has been completed

ADC Status Register 3 (EADC_STATUS3)

Register	Offset	R/W	Description	Reset Value
EADC_STATUS3	EADC_BA+0xFC	R	ADC Status Register 3	0x0000_001F

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

Bits	Description	
[31:5]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[4:0]	**CURSPL**	**ADC Current Sample Module** This register show the current ADC is controlled by which sample module control logic modules. If the ADC is Idle, this bit filed will set to 0x1F. This is a read only register.

ADC Double Data Register n for Sample Module n (EADC_DDAT0~3)

Register	Offset	R/W	Description	Reset Value
EADC_DDAT0	EADC_BA+0x100	R	ADC Double Data Register 0 for Sample Module 0	0x0000_0000
EADC_DDAT1	EADC_BA+0x104	R	ADC Double Data Register 1 for Sample Module 1	0x0000_0000
EADC_DDAT2	EADC_BA+0x108	R	ADC Double Data Register 2 for Sample Module 2	0x0000_0000
EADC_DDAT3	EADC_BA+0x10C	R	ADC Double Data Register 3 for Sample Module 3	0x0000_0000

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

Bits	Description	
[31:17]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[17]	**VALID**	**Valid Flag** 0 = Double data in RESULT (EADC_DDATn[15:0]) is not valid. 1 = Double data in RESULT (EADC_DDATn[15:0]) is valid. This bit is set to 1 when corresponding sample module channel analog input conversion is completed and cleared by hardware after EADC_DDATn register is read. (n=0~3).
[16]	**OV**	**Overrun Flag** 0 = Data in RESULT (EADC_DATn[15:0], n=0~3) is recent conversion result. 1 = Data in RESULT (EADC_DATn[15:0], n=0~3) is overwrite. If converted data in RESULT[15:0] has not been read before new conversion result is loaded to this register, OV is set to 1. It is cleared by hardware after EADC_DDAT register is read.
[15:0]	**RESULT**	**ADC Conversion Results** This field contains 12 bits conversion results. When the DMOF (EADC_CTL[9]) is set to 0, 12-bit ADC conversion result with unsigned format will be filled in RESULT [11:0] and zero will be filled in RESULT [15:12]. When DMOF (EADC_CTL[9]) set to 1, 12-bit ADC conversion result with 2'complement format will be filled in RESULT [11:0] and signed bits to will be filled in RESULT [15:12].

ADC Power Management Register (EADC_PWRM)

Register	Offset	R/W	Description	Reset Value
EADC_PWRM	EADC_BA+0x110	R/W	ADC Power Management Register	0x0006_E012

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

Bits	Description	
[31:20]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[19:8]	**LDOSUT**	**ADC Internal LDO Start-up Time** Set this bit fields to control LDO start-up time. The minimum required LDO start-up time is 20us. LDO start-up time = (1/ADC_CLK) x LDOSUT
[7:4]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[3:2]	**PWDMOD**	**ADC Power-down Mode** Set this bit fields to select ADC Power-down mode when system power-down. 00 = ADC Deep Power-down mode. 01 = ADC Power down. 10 = ADC Standby mode. 11 = ADC Deep Power-down mode. **Note:** Different PWDMOD has different power down/up sequence, in order to avoid ADC powering up with wrong sequence; user must keep PWMOD consistent each time in power down and start up.
[1]	**PWUCALEN**	**Power Up Calibration Function Enable Control** 0 = Disable the function of calibration at power up. 1 = Enable the function of calibration at power up.
[0]	**PWUPRDY**	**ADC Power-up Sequence Completed and Ready for Conversion(Read Only)** 0 = ADC is not ready for conversion may be in power down state or in the progress of start up. 1 = ADC is ready for conversion.

ADC Channel Switch Presetting Control Register (EADC_CHSPC)

Register	Offset	R/W	Description	Reset Value
EADC_CHSPC	EADC_BA+0x200	R/W	ADC Channel Switch Presetting 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		**CHSPC**					

Bits	Description	
[31:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5:0]	**CHSPC**	**ADC Channel Switch Presetting Control** 0x00 = No channel switch presetting function. 0x21 = Enable switch presetting. Channel switch is preset 1 EADC clock before the end of conversion, then switched when end of conversion. Others = Reserved. Do not use. **Note:** For EADC converting multi-channel input signal, please set 0x21 to CHSPC

6.17 I²S Controller (I²S)

6.17.1 Overview

The I²S controller consists of I²S protocol to interface with external audio CODEC. Two 16-level depth FIFO for reading path and writing path respectively and is capable of handling 8/16/24/32 bits audio data sizes. PDMA controller handles the data movement between FIFO and memory.

6.17.2 Features

- Support Master mode and Slave mode
- Capable of handling 8, 16, 24 and 32 bits data sizes in each audio channel
- Supports monaural and stereo audio data
- Supports I²S protocols: Philips standard, MSB-justified, and LSB-justified data format
- Supports PCM protocols: PCM standard, MSB-justified, and LSB-justified data format
- PCM protocol supports TDM multi-channel transmission in one audio sample, and the number of data channel can be set as 2, 4, 6, or 8
- Provides two 16-level FIFO data buffers, one for transmitting and the other for receiving
- Generates interrupt requests when buffer levels cross a programmable boundary
- Supports two PDMA requests, one for transmitting and the other for receiving

6.17.3 Block Diagram

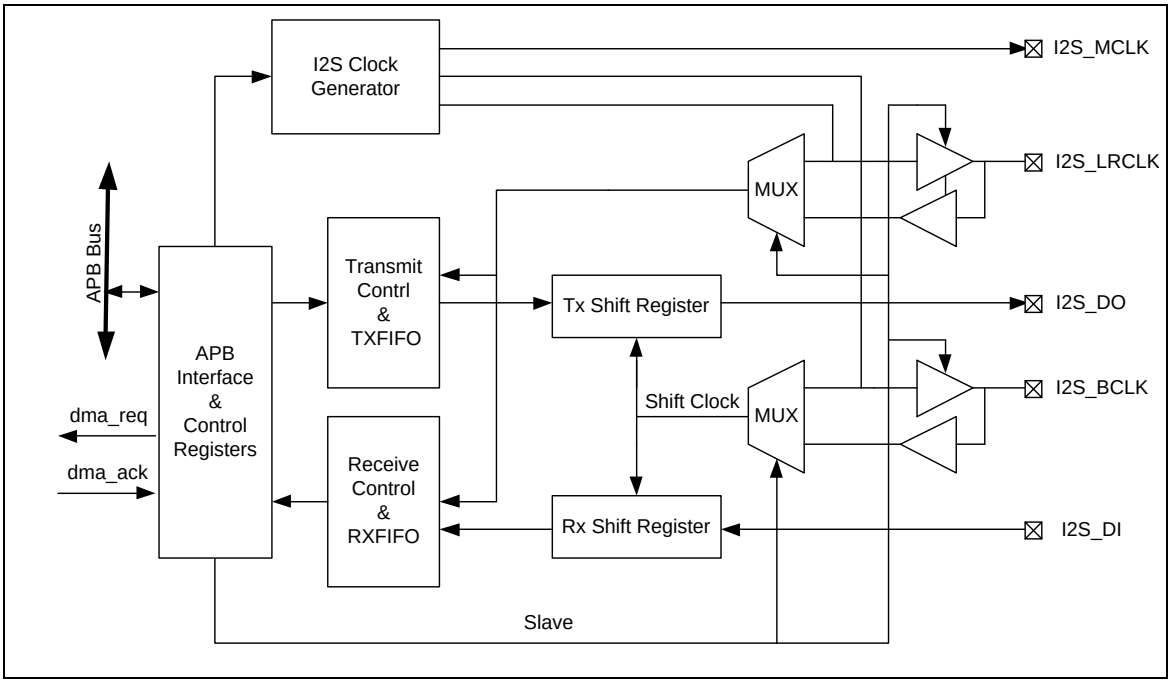


Figure 6.17-1 I²S Controller Block Diagram

6.17.4 Basic Configuration

- Clock source configuration
  - Select the source of I²S peripheral clock on I2SSEL (CLK_CLKSEL3[17:16]).
  - Enable I²S peripheral clock in I2SCKEN (CLK_APBCLK0[29]).
- Reset configuration
  - Reset I²S controller in I2SRST (SYS_IPRST1[29]).
- Pin configuration

Group	Pin Name	GPIO	MFP
I2S0	I2S0_BCLK	PD.0	MFP4
		PD.6	MFP3
	I2S0_DI	PB.13	MFP2
		PD.4	MFP4
	I2S0_DO	PB.14	MFP2
		PD.5	MFP3
	I2S0_LRCK	PD.1	MFP4
		PD.3	MFP3
	I2S0_MCLK	PB.15	MFP2
		PD.2	MFP3

6.17.5 Functional Description

6.17.5.1 I²S Clock

The I²S controller has four clock sources selected by I2SSEL (CLK_CLKSEL3[17:16]). The I²S clock rate must be slower than or equal to system clock rate.

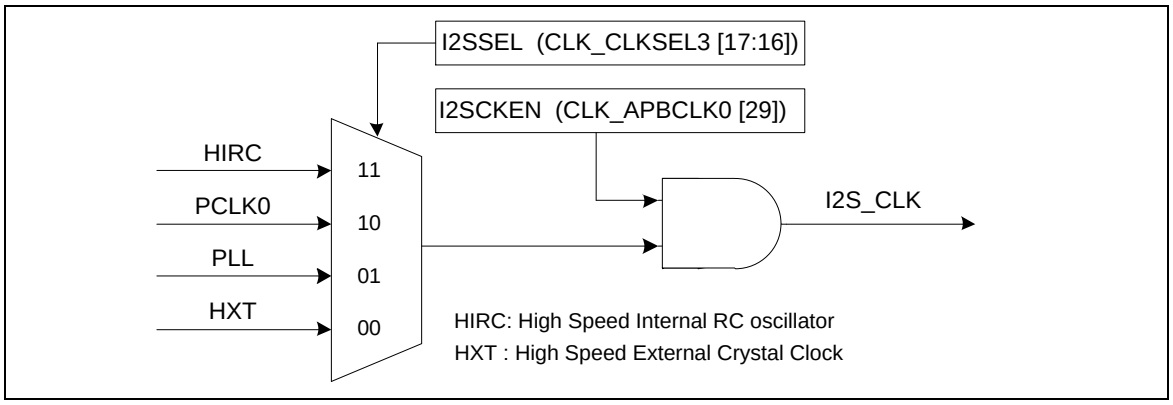


Figure 6.17-2 I²S Clock Control Diagram

6.17.5.2 Master/Slave Interface

The I²S function can operate as master or slave mode by setting SLAVE (I2S_CTL0[8]) to communicate with other I²S slave or master. The serial bus clock I2S_BCLK is permanently generated by the master even through there is no transferring data bit at the moment. The word select signal I2S_LRCLK is also generated by the master and it indicates the beginning of a new data word and the targeted audio channel. Both the I2S_LRCLK and the transmitting data change synchronously to the falling edges of I2S_BCLK.

In some applications, especially for Audio-ADC or Audio-DAC, a master clock signal, I2S_MCLK, is required with a fixed phase relation to the I2S_BCLK. The I2S_MCLK is enabled by MCLKEN (I2S_CTL0[15]). In Master mode, the I2S_MCLK, I2S_BCLK, I2S_LRCLK is output to device slave. And if in slave mode, the I2S_MCLK is output to master, and I2S_BCLK or I2S_LRCLK is input from master.

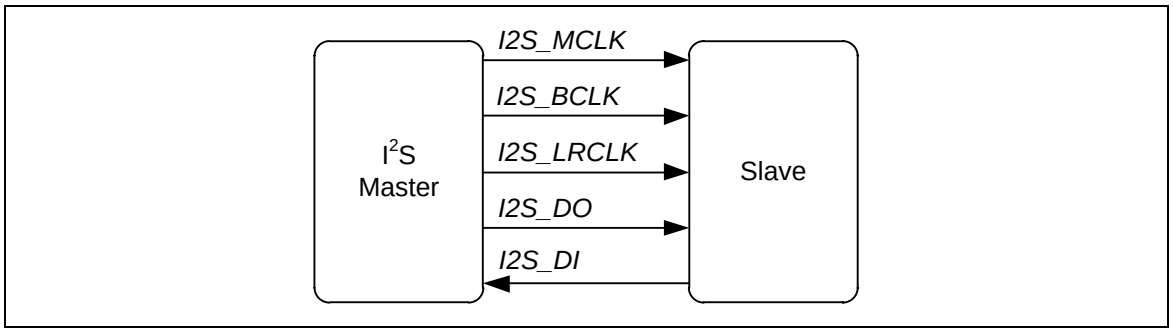


Figure 6.17-3 Master mode Interface Block Diagram

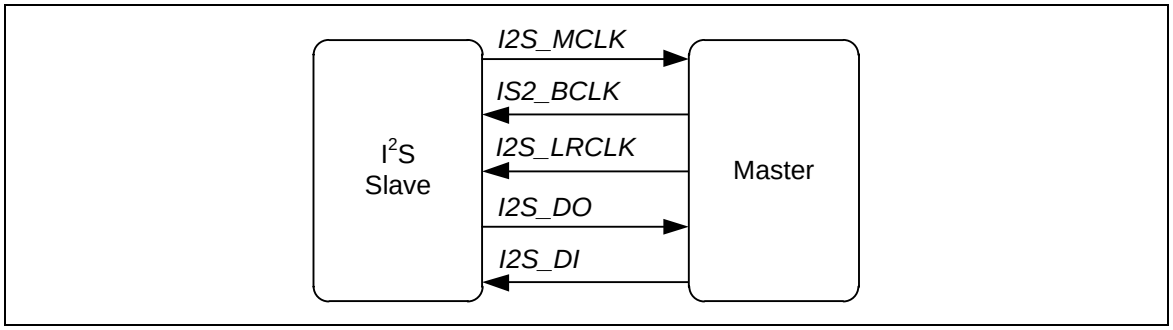


Figure 6.17-4 Slave mode Interface Block Diagram

6.17.5.3 I²S Operation

The I²S controller supports MSB-justified, LSB-justified, and I²S Philips standard data format. The I2S_LRCLK signal indicates which audio channel is in transferring. The bit count of an audio channel is defined by CHWIDTH (I2S_CTL0[29:28]), and the bit-width of data word in an audio channel is determined by DATWIDTH (I2S_CTL0[5:4]). If CHWIDTH (I2S_CTL0[29:28]) is less than DATWIDTH (I2S_CTL0[5:4]), the hardware will set the channel bit-width to be same as data bit-width. However, there will be redundant zero bits in each audio channel if CHWIDTH (I2S_CTL0[29:28]) is greater than DATWIDTH (I2S_CTL0[5:4]).

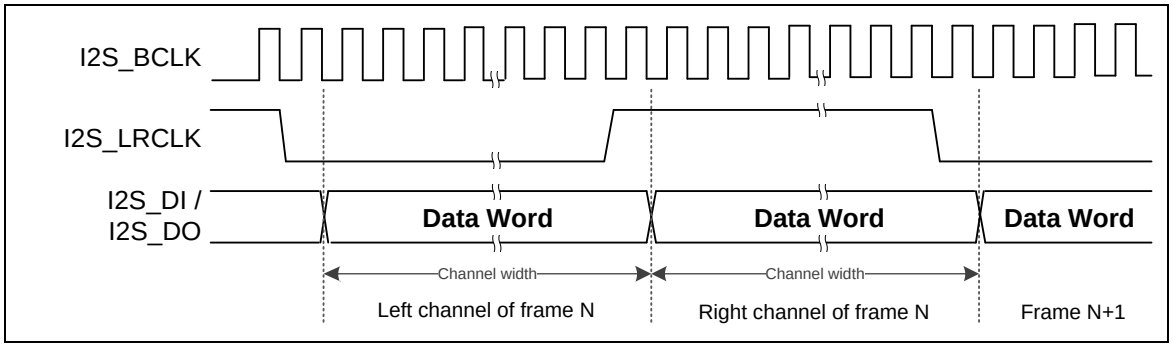


Figure 6.17-5 I²S Channel Width and Data Width (CHWIDTH≦DATWIDTH)

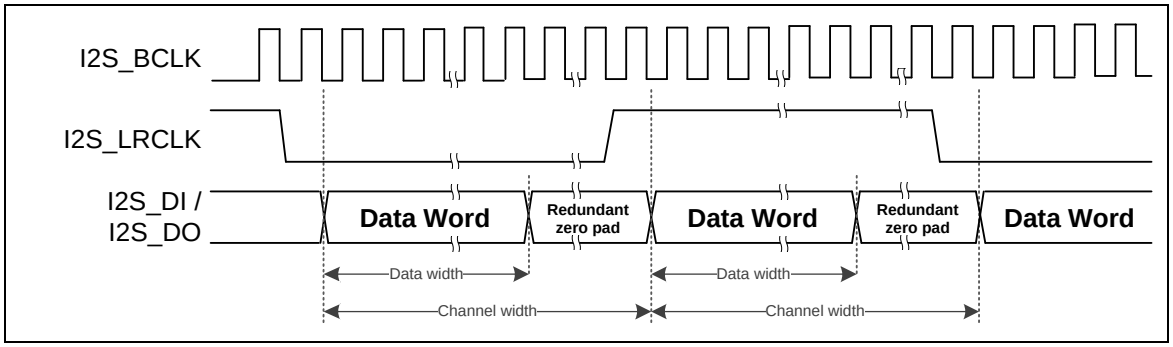


Figure 6.17-6 I²S Channel Width and Data Width (CHWIDTH > DATWIDTH)

The transferring data sequence is always started from the MSB (most significance bit) to the LSB (least significance bit). As the Figure 6.17-7, transmitting data are read at rising edge of I2S_BCLK and sent out at falling edge of I2S_BCLK in I²S protocol. In I²S data format, the MSB is sent and latched at the next falling edge of I2S_BCLK cycle after the transition of I2S_LRCLK. In MSB justified data format, the I2S_LRCLK changes the polarity at the transmitting of the first data bit (MSB) in each audio channel. In LSB justified data format, the LSB is sent and latched at the last I2S_BCLK cycle of an audio channel. The MSB justified and LSB justified data format of I²S

protocol can be selected by FORMAT (I2S_CTL0[26:24]).

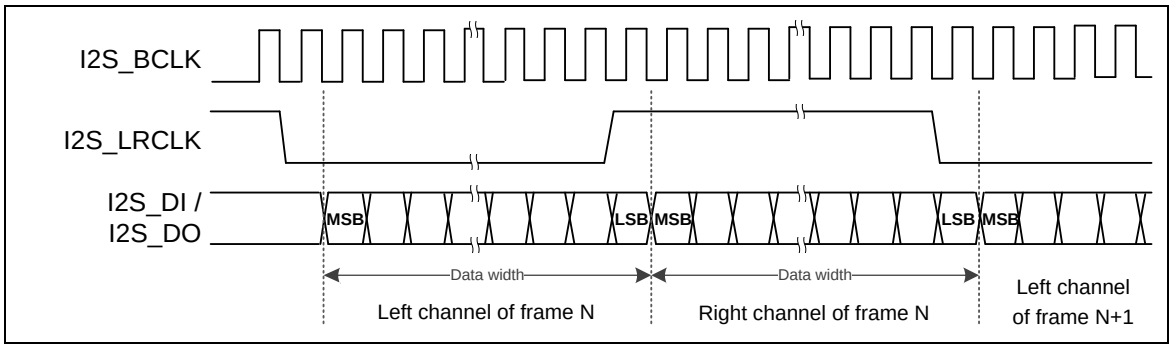


Figure 6.17-7 I^{2}S Data Format Timing Diagram (FORMAT = 0x0 ; CHWIDTH≦DATWIDTH)

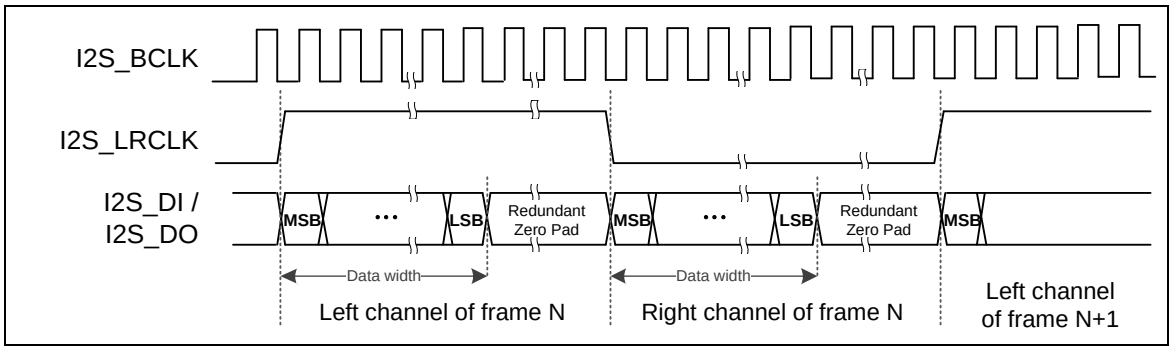


Figure 6.17-8 MSB Justified Data Format (FORMAT = 0x1 ; CHWIDTH > DATWIDTH)

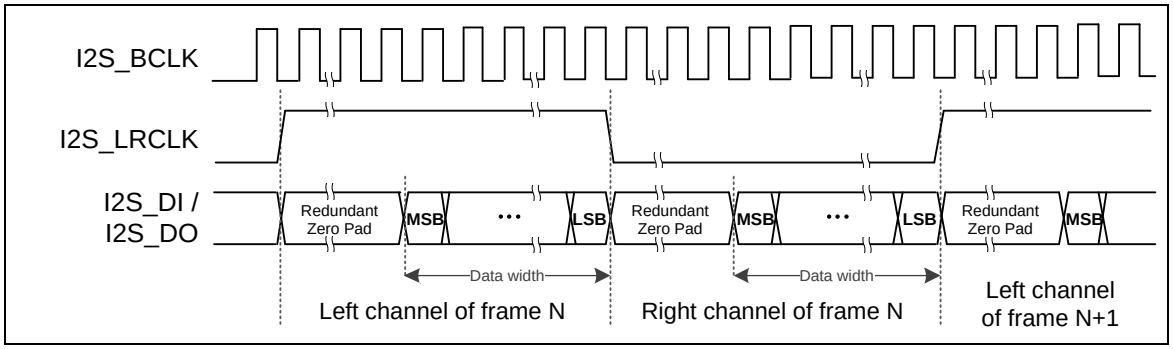


Figure 6.17-9 LSB Justified Data Format (FORMAT = 0x2 ; CHWIDTH > DATWIDTH)

The I^{2}S controller also supports PCM audio transmission which can be selected by FORMAT (I2S_CTL0[26:24]). In PCM protocol, the function of I2S_LRCLK is simply to identify the beginning of an audio sample (or audio frame) and it is always indicated by the rising edge of the pulse. Therefore, the I2S_LRCLK in PCM protocol may be also called “frame start” or “frame sync” signal. In master device, there are two common representations for the width of the frame start pulse which can use PCMSYNC (I2S_CTL0[27]) to choose: One is equivalent to the period of a channel width and the other is equivalent to a single period of the I2S_BCLK.

Same as I^{2}S protocol, the DATWIDTH (I2S_CTL0[5:4]) and CHWIDTH (I2S_CTL0[29:28]) can be used to configure the data bit-width and channel bit-width in PCM protocol. Besides, FORMAT (I2S_CTL0[26:24]) can also be used to select the different data formats of PCM standard mode, PCM with MSB justified, and PCM with LSB justified data format.

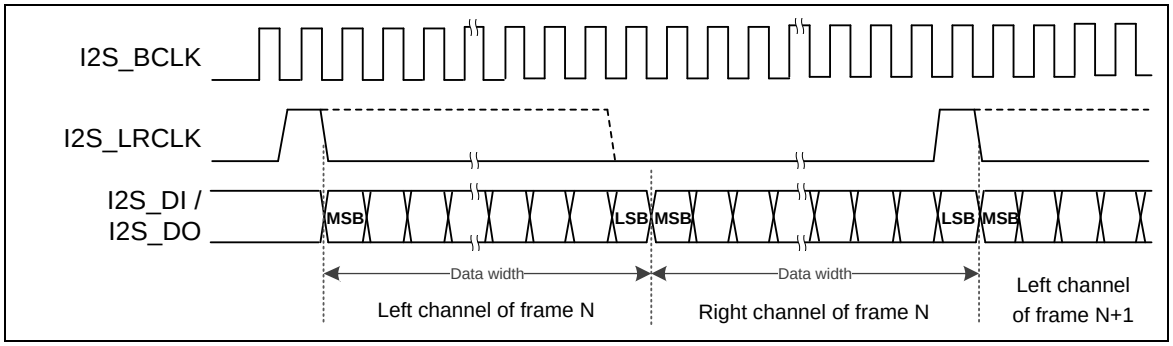


Figure 6.17-10 Standard PCM Audio Timing Diagram (FORMAT = 0x4 ; CHWIDTH≦DATWIDTH)

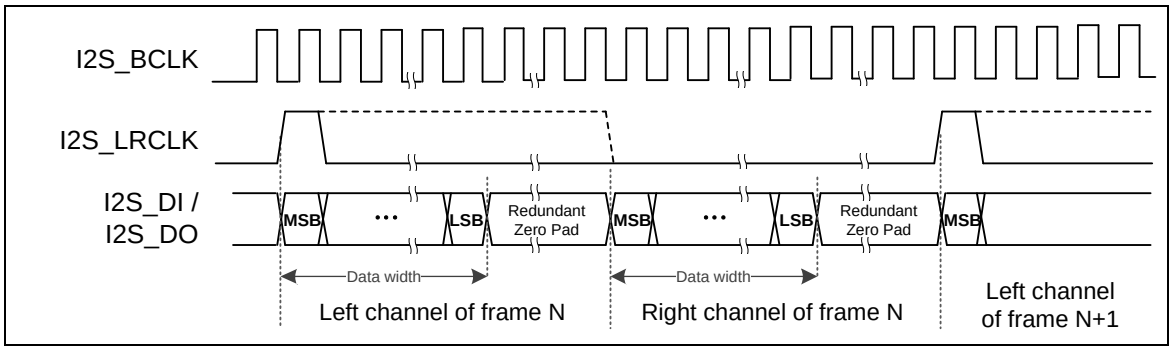


Figure 6.17-11 PCM with MSB Justified Data Format (FORMAT = 0x5 ; CHWIDTH > DATWIDTH)

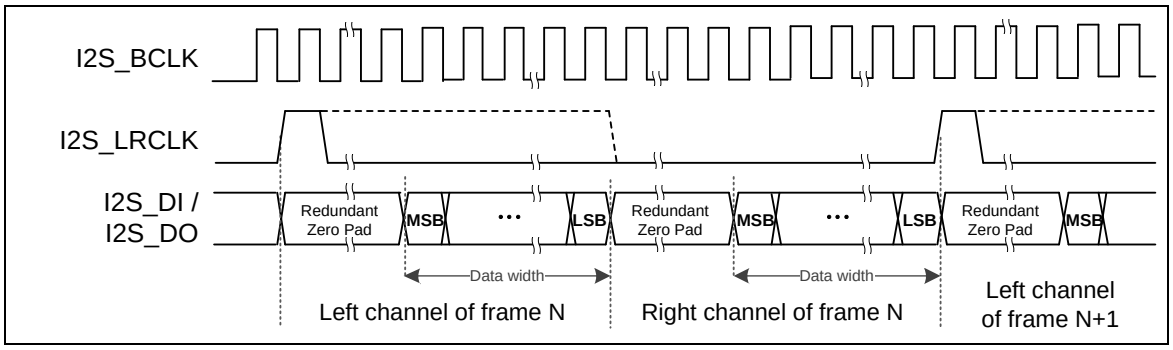


Figure 6.17-12 PCM with LSB Justified Data Format (FORMAT = 0x6 ; CHWIDTH > DATWIDTH)

6.17.5.4 TDM Multi-channel transmission

The PCM mode in this I^{2}S controller also supports TDM transmission. The Time Division Multiplexed (TDM) method allows multiple channels of audio data to be transmitted on a single data line. The TDM interface is similar to the 2-channel PCM audio interface with the exception that more audio channels are transmitted within a sample frame which is defined by a period of the I2S_LRCLK. The channel number of TDM interface is typically 4, 6, or 8 and it is selected by TDMCHNUM (I2S_CTL0[31:30]).

Same as previous I^{2}S and PCM descriptions, each channel block is comprised of the audio data word followed by a sufficient number of zero data bits to complete one channel block. The bit-width of data word and channel block are defined by DATWIDTH (I2S_CTL0[5:4]) and CHWIDTH (I2S_CTL0[29:28]) respectively. Note that the TDM PCM mode supports 16-bit, 24-bit, 32-bit audio data word (excluding 8-bit data), and the hardware will set the bit-width of transmitting data as 16-bit if DATWIDTH (I2S_CTL0[5:4]) is 0x0. The pulse width of frame start

signal is also selected by PCMSYNC (I2S_CTL0[27]).

The examples of 6-channel TDM transmission with 24-bit audio data in 32-bit channel block are shown in Figure 6.17-13. In 2-channel audio interface, we may call the first and second audio channels as left-channel and right-channel (or channel0 and channel1). In TDM multi-channel application, we call the first and second audio channels as channel0 and channel1.

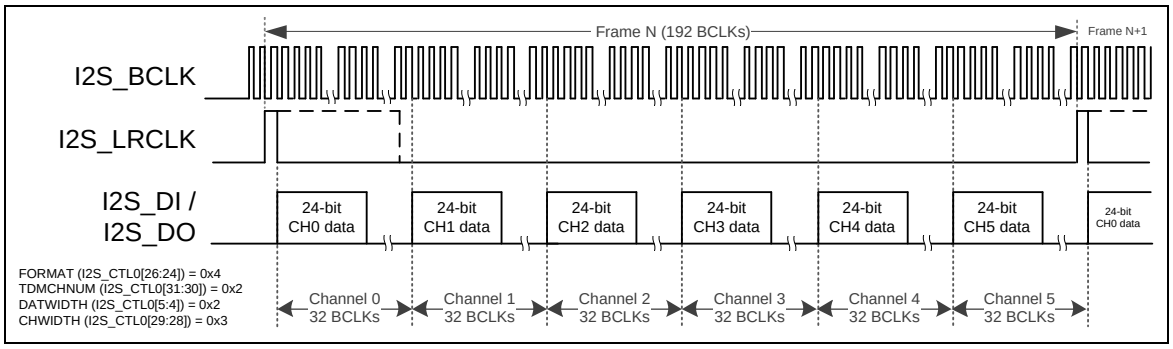


Figure 6.17-13 TDM 6-channel audio format with 24-bit data in 32-bit channel block (PCM standard data format; FORMAT=0x4)

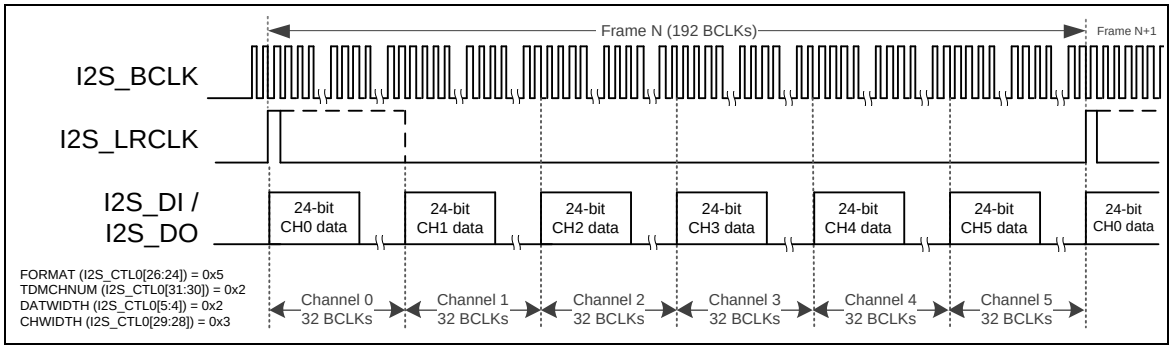


Figure 6.17-14 TDM 6-channel audio format with 24-bit data in 32-bit channel block (PCM with MSB justified; FORMAT=0x5)

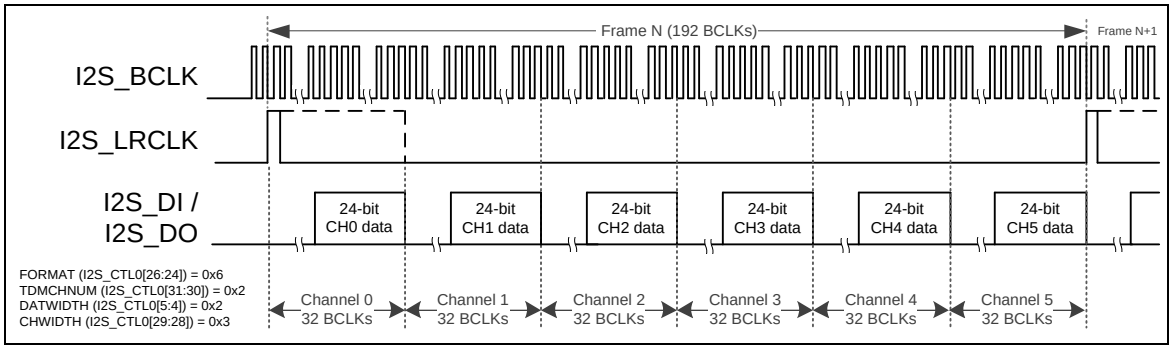


Figure 6.17-15 TDM 6-channel audio format with 24-bit data in 32-bit channel block (PCM with LSB justified; FORMAT=0x6)

6.17.5.5 Zero Crossing

When playing the audio by I^{2}S controller, the output transmitting data comes from the memory by PDMA or by CPU. However, there may be some pop noise which induces the uncomfortable hearing if the playing sound volume is changed greatly by user. The zero-crossing event of audio data means the playing sound is relatively silent at the moment. Therefore, the zero-cross interrupt can be used for the indication of gain level adjustment in order to prevent the huge variance of sound volume.

If zero-cross detection of individual audio channel is enabled by the corresponding control bit from CH0ZCEN to CH7ZCEN (I2S_CTL1[0] to I2S_CTL1[7]), the hardware will detect the next transferring data word of the corresponding audio channel whether it is zero or its MSB has been changed. If zero value or MSB (sign bit) changing of the transmitting audio data has been detected while zero-cross detection is enabled, the hardware will set the corresponding status bit from CH0ZCIF to CH7ZCIF (I2S_STATUS1[0] to I2S_STATUS1[7]) for the audio channel and then keep the output audio data silent (all data bit zero) automatically until the corresponding event status bit is cleared by software.

Therefore, if user wants to modify the audio playing gain, users can enable the zero crossing interrupt function, CH0ZCIEN to CH7ZCIEN (I2S_IEN[16:23]) , to indicate the zero crossing time and to change the audio gain. This will reduce the pop noise.

6.17.5.6 PDMA Mode

The I^{2}S function can use PDMA function for transmitting or receiving data access. If the PDMA function of transmitting data is enabled by TXPDMAEN (I2S_CTL0[20]), the I^{2}S controller will generate the request signal and then get transmitting audio data from memory by PDMA IP automatically while TX FIFO is not full. If the PDMA function of receiving data is enabled by RXPDMAEN (I2S_CTL0[19]), the I^{2}S controller will generate the request signal and then the receiving data will be moved into memory by PDMA hardware automatically while the RX FIFO is not empty. Therefore, using PDMA function will save the CPU loading to service other functions.

6.17.5.7 I^{2}S Interrupt Sources

The I^{2}S controller supports zero-cross interrupt of individual audio channel, transmit FIFO threshold level interrupt, transmit FIFO overflow interrupt and transmit FIFO underflow interrupt in transmit operation. In receive operation, it supports receive FIFO threshold level interrupt, receive FIFO overflow interrupt and receive FIFO underflow interrupt. When I^{2}S interrupt occurs, user can check I2STXINT (I2S_STATUS0[2]) and I2SRXINT (I2S_STATUS0[1]) flags to recognize the interrupt sources.

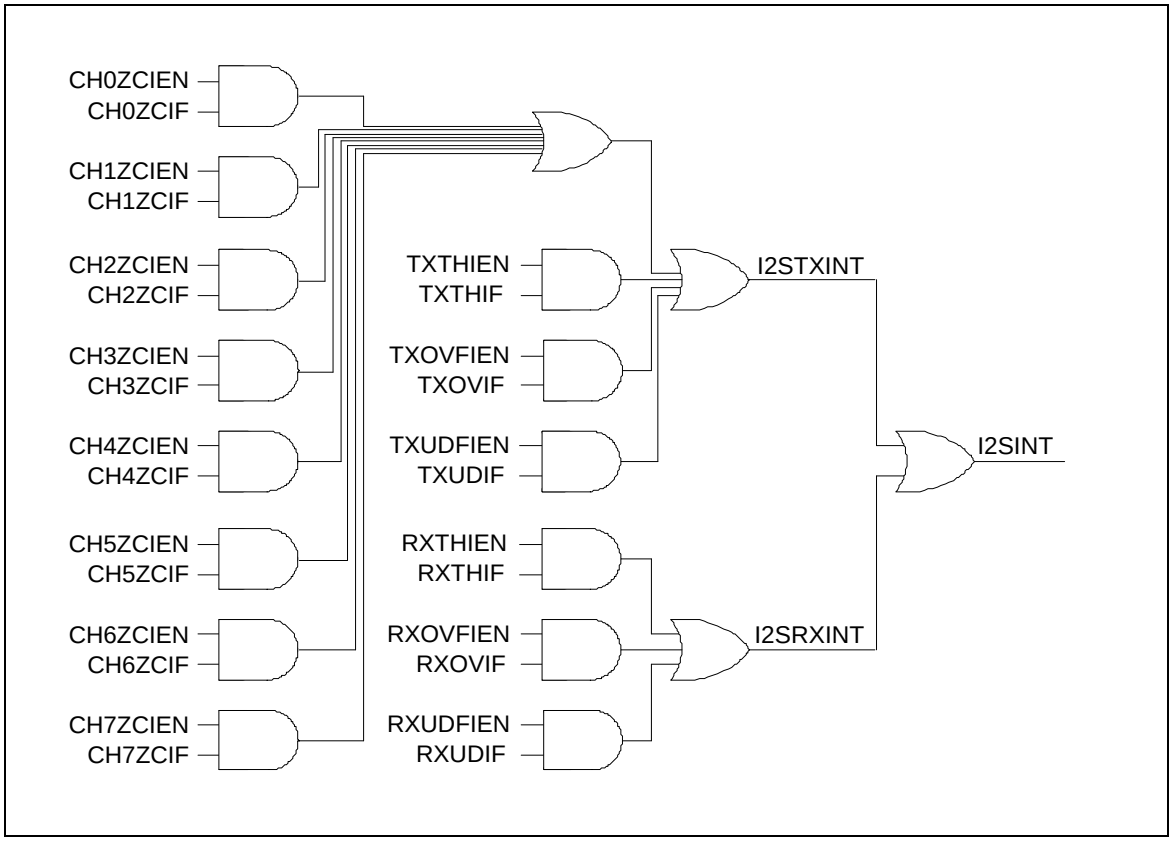


Figure 6.17-16 I^2S Interrupts

6.17.5.8 FIFO Operation

In 2-channel I^2S or PCM protocol, the bit-width of audio data in a channel block can be 8, 16, 24, or 32 bits. The memory arrangements of audio data for various settings are shown in Figure 6.17-17.

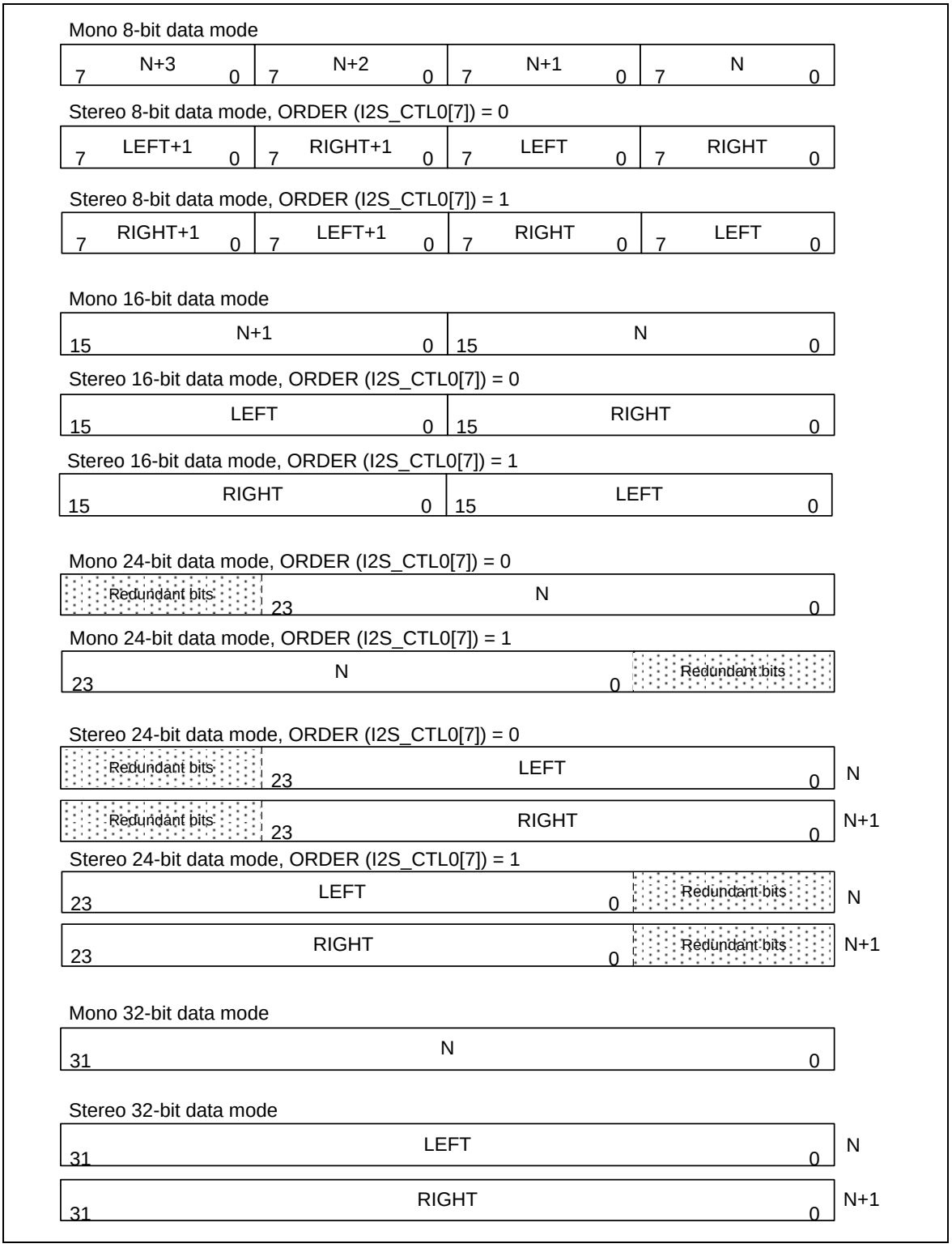


Figure 6.17-17 FIFO Contents for Various 2-channel Audio Modes

In 4-channel TDM PCM data format, the bit-width of audio data in a channel block can be 16, 24, or 32 bits. The memory arrangements of audio data for various settings are shown in Figure 6.17-18.

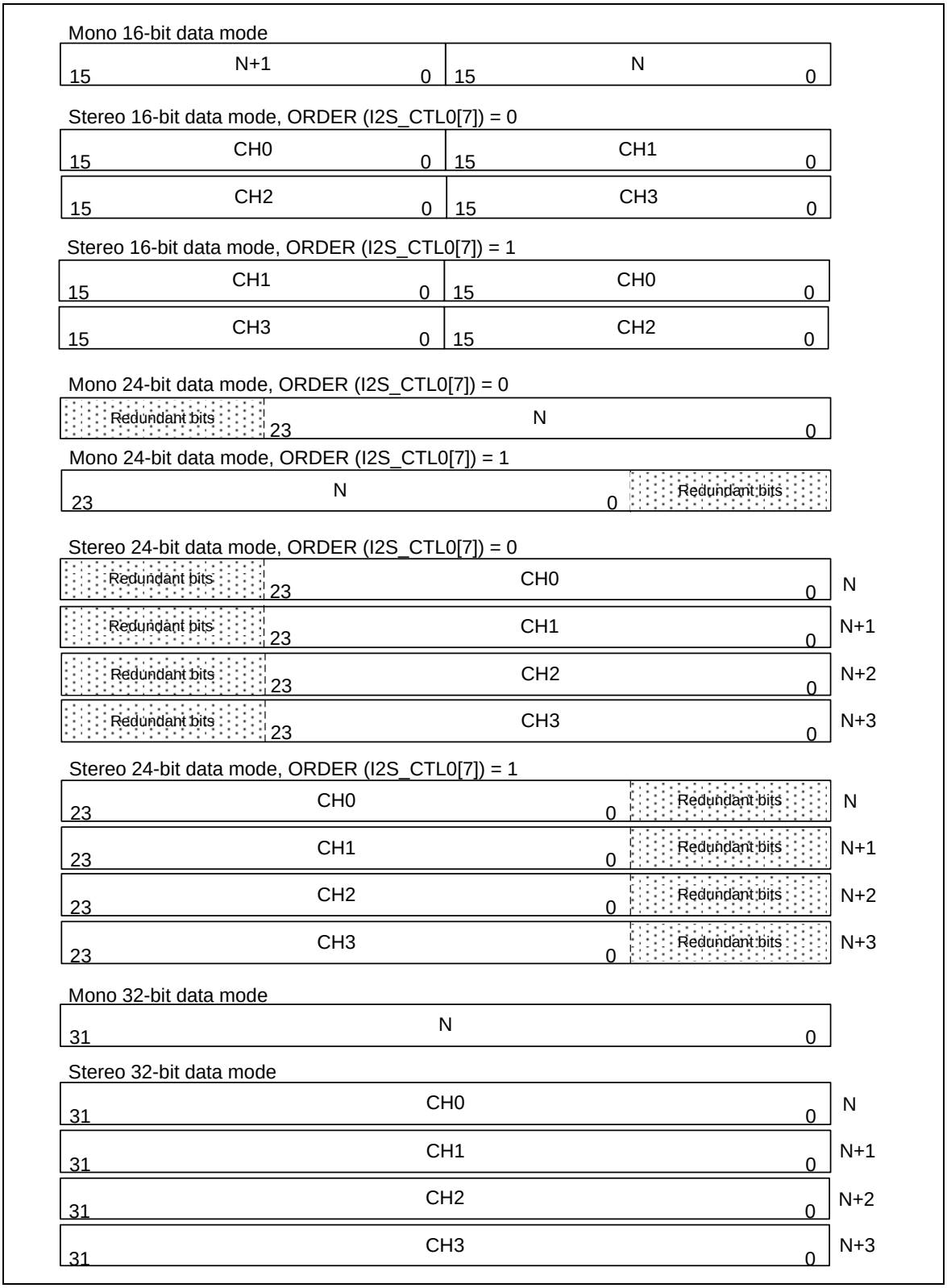


Figure 6.17-18 FIFO Contents for Various 4-channel Audio Modes

In 6-channel TDM PCM data format, the bit-width of audio data in a channel block can be 16, 24, or 32 bits. The memory arrangements of audio data for various settings are shown Figure

6.17-19. In 16-bit audio data transmission, ORDER (I2S_CTL0[7]) can be used to swap the audio data of even and odd channels which are stored in transmitting and receiving FIFO. In 24-bit audio data transmission, ORDER (I2S_CTL0[7]) can be also used to select the left-alignment or right-alignment formula of audio data which is stored in 32-bit FIFO entries.

The FIFO content of 8-channel TDM PCM data format is similar to 6-channel and it can be analogized easily.

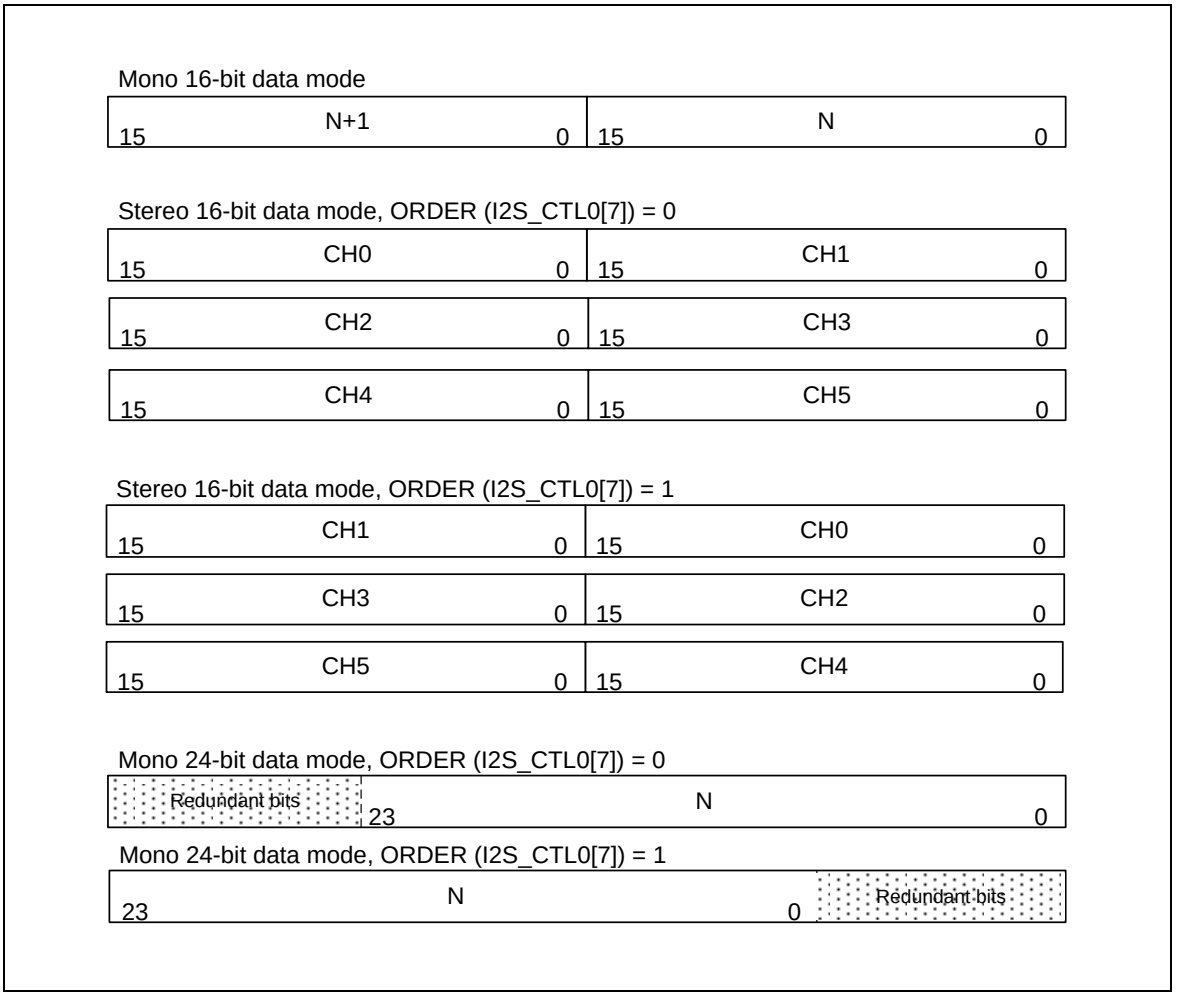


Figure 6.17-19 FIFO Contents for Various 6-channel Audio Modes (Part-1)

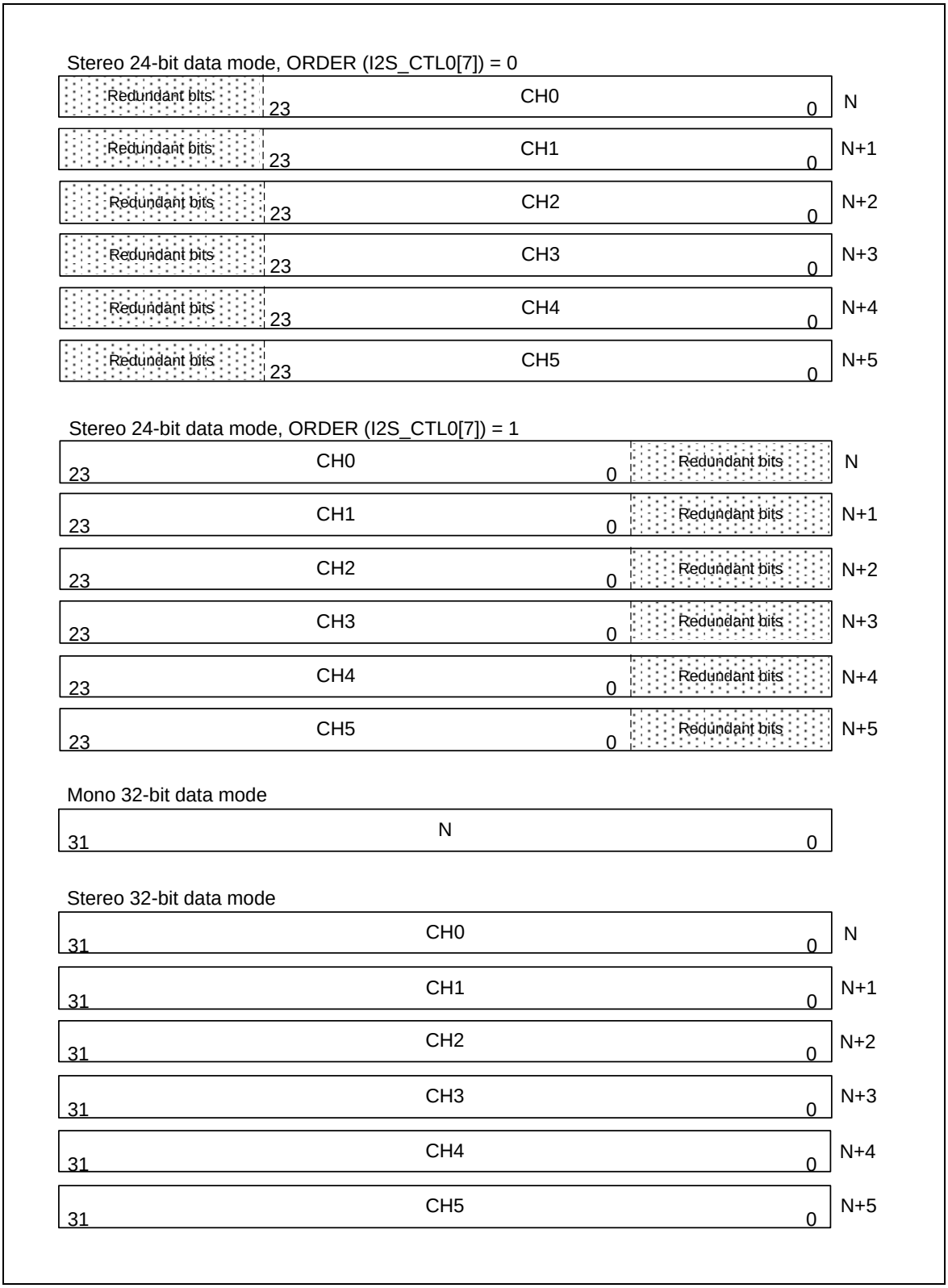


Figure 6.17-20 FIFO Contents for Various 6-channel Audio Modes (Part-2)

6.17.6 Register Map

R: Read only, **W**: Write only, **R/W**: Both read and write

Register	Offset	R/W	Description	Reset Value
I2S Base Address **I2S_BA = 0x4004_8000**				
I2S_CTL0	I2S_BA+0x00	R/W	I^2S Control Register 0	0x0000_0000
I2S_CTL1	I2S_BA+0x20	R/W	I^2S Control Register 1	0x0000_0000
I2S_CLKDIV	I2S_BA+0x04	R/W	I^2S Clock Divider Register	0x0000_0000
I2S_IEN	I2S_BA+0x08	R/W	I^2S Interrupt Enable Register	0x0000_0000
I2S_STATUS0	I2S_BA+0x0C	R/W	I^2S Status Register 0	0x0014_1000
I2S_STATUS1	I2S_BA+0x24	R/W	I^2S Status Register 1	0x0000_0000
I2S_TXFIFO	I2S_BA+0x10	W	I^2S Transmit FIFO Register	0x0000_0000
I2S_RXFIFO	I2S_BA+0x14	R	I^2S Receive FIFO Register	0x0000_0000

Note:

1. Any register not listed here is reserved and must not be written. The result of a read operation on these bits is undefined.
2. The reserved register fields that listed in register description must be written to their reset value. Writing reserved fields with other than reset values may produce undefined results.

6.17.7 Register Description

I2S Control Register 0 (I2S_CTL0)

Register	Offset	R/W	Description	Reset Value
I2S_CTL0	I2S_BA+0x00	R/W	I^2S Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
TDMCHNUM		**CHWIDTH**		**PCMSYNC**	**FORMAT**		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
RXLCH	**Reserved**	**RXPDMAEN**	**TXPDMAEN**	**RXFBCLR**	**TXFBCLR**	**FLZCDEN**	**FRZCDEN**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
MCLKEN	**Reserved**						**SLAVE**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ORDER	**MONO**	**DATWIDTH**		**MUTE**	**RXEN**	**TXEN**	**I2SEN**

Bits	Description	
[31:30]	**TDMCHNUM**	**TDM Channel Number** This bit fields are used to define the TDM channel number in one audio frame while PCM mode (FORMAT[2] = 1). 00 = 2 channels in audio frame. 01 = 4 channels in audio frame. 10 = 6 channels in audio frame. 11 = 8 channels in audio frame.
[29:28]	**CHWIDTH**	**Channel Width** This bit fields are used to define the length of audio channel. If CHWIDTH < DATWIDTH, the hardware will set the real channel length as the bit-width of audio data which is defined by DATWIDTH. 00 = The bit-width of each audio channel is 8-bit. 01 = The bit-width of each audio channel is 16-bit. 10 = The bit-width of each audio channel is 24-bit. 11 = The bit-width of each audio channel is 32-bit.
[27]	**PCMSYNC**	**PCM Synchronization Pulse Length Selection** This bit field is used to select the high pulse length of frame synchronization signal in PCM protocol 0 = One BCLK period. 1 = One channel period. **Note:** This bit is only available in master mode

[26:24]	**FORMAT**	**Data Format Selection** 000 = I^2S standard data format. 001 = I^2S with MSB justified. 010 = I^2S with LSB justified. 011 = Reserved. Do not use. 100 = PCM standard data format. 101 = PCM with MSB justified. 110 = PCM with LSB justified. 111 = Reserved. Do not use.
[23]	**RXLCH**	**Receive Left Channel Enable Control** When monaural format is selected (MONO = 1), I^2S will receive channel1 data if RXLCH is set to 0, and receive channel0 data if RXLCH is set to 1. 0 = Receives channel1 data in MONO mode. 1 = Receives channel0 data in MONO mode.
[22]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[21]	**RXPDMAEN**	**Receive PDMA Enable Control** 0 = Receiver PDMA function Disabled. 1 = Receiver PDMA function Enabled.
[20]	**TXPDMAEN**	**Transmit PDMA Enable Control** 0 = Transmit PDMA function Disabled. 1 = Transmit PDMA function Enabled.
[19]	**RXFBCLR**	**Receive FIFO Buffer Clear** 0 = No Effect. 1 = Clear RX FIFO. **Note1:** Write 1 to clear receive FIFO, internal pointer is reset to FIFO start point, and RXCNT (I2S_STATUS1[20:16]) returns 0 and receive FIFO becomes empty. **Note2:** This bit is cleared by hardware automatically, read it return zero.
[18]	**TXFBCLR**	**Transmit FIFO Buffer Clear** 0 = No Effect. 1 = Clear TX FIFO. **Note1:** Write 1 to clear transmit FIFO, internal pointer is reset to FIFO start point, and TXCNT (I2S_STATUS1[12:8]) returns 0 and transmit FIFO becomes empty but data in transmit FIFO is not changed. **Note2:** This bit is clear by hardware automatically, read it return zero.
[17]	**FLZCDEN**	**Force Left Channel Zero Cross Data Option Bit** If this bit set to 1, when channel (Ch0,Ch2,Ch4,Ch6) data sign bit changes or next shift data bits are all 0 then the channel ZCIF flag in I2S_STATUS1 register is set to 1 and channel data will force zero. This function is only available in transmit operation. 0 = Keep channel (Ch0,Ch2,Ch4,Ch6) data , when zero crossing flag on. 1 = Force channel (Ch0,Ch2,Ch4,Ch6) data to zero, when zero crossing flag on.
[16]	**FRZCDEN**	**Force Right Channel Zero Cross Data Option Bit** If this bit set to 1, when channel (Ch1,Ch3,Ch5,Ch7) data sign bit changes or next shift data bits are all 0 then the channel ZCIF flag in I2S_STATUS1 register is set to 1 and channel data will force zero. This function is only available in transmit operation. 0 = Keep channel (Ch1,Ch3,Ch5,Ch7) data , when zero crossing flag on. 1 = Force channel (Ch1,Ch3,Ch5,Ch7) data to zero, when zero crossing flag on.

[15]	**MCLKEN**	**Master Clock Enable Control** If MCLKEN is set to 1, I^{2}S controller will generate master clock on I2S_MCLK pin for external audio devices. 0 = Master clock Disabled. 1 = Master clock Enabled.
[14:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8]	**SLAVE**	**Slave Mode Enable Control** 0 = Master mode. 1 = Slave mode. **Note:** I^{2}S can operate as master or slave. For Master mode, I2S_BCLK and I2S_LRCLK pins are output mode and send out bit clock to Audio CODEC chip. In Slave mode, I2S_BCLK and I2S_LRCLK pins are input mode and I2S_BCLK and I2S_LRCLK signals are received from outer Audio CODEC chip.
[7]	**ORDER**	**Stereo Data Order in FIFO** In 8-bit/16-bit data width, this bit is used to select whether the even or odd channel data is stored in higher byte. In 24-bit data width, this is used to select the left/right alignment method of audio data which is stored in data memory consisted of 32-bit FIFO entries. 0 = Even channel data at high byte in 8-bit/16-bit data width. LSB of 24-bit audio data in each channel is aligned to right side in 32-bit FIFO entries. 1 = Even channel data at low byte. MSB of 24-bit audio data in each channel is aligned to left side in 32-bit FIFO entries.
[6]	**MONO**	**Monaural Data Control** 0 = Data is stereo format. 1 = Data is monaural format. **Note:** when chip records data, RXLCH (I2S_CTL0[23]) indicates which channel data will be saved if monaural format is selected.
[5:4]	**DATWIDTH**	**Data Width** This bit field is used to define the bit-width of data word in each audio channel 00 = The bit-width of data word is 8-bit. 01 = The bit-width of data word is 16-bit. 10 = The bit-width of data word is 24-bit. 11 = The bit-width of data word is 32-bit.
[3]	**MUTE**	**Transmit Mute Enable Control** 0 = Transmit data is shifted from buffer. 1 = Send zero on transmit channel.
[2]	**RXEN**	**Receive Enable Control** 0 = Data receiving Disabled. 1 = Data receiving Enabled.
[1]	**TXEN**	**Transmit Enable Control** 0 = Data transmission Disabled. 1 = Data transmission Enabled.
[0]	**I2SEN**	**I^{2}S Controller Enable Control** 0 = I^{2}S controller Disabled. 1 = I^{2}S controller Enabled.

I2S Control Register 1 (I2S_CTL1)

Register	Offset	R/W	Description	Reset Value
I2S_CTL1	I2S_BA+0x20	R/W	I^2S Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						**PB16ORD**	**PBWIDTH**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved				**RXTH**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**TXTH**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CH7ZCEN	**CH6ZCEN**	**CH5ZCEN**	**CH4ZCEN**	**CH3ZCEN**	**CH2ZCEN**	**CH1ZCEN**	**CH0ZCEN**

Bits	Description	
[31:26]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[25]	**PB16ORD**	**FIFO Read/Write Order in 16-bit Width of Peripheral Bus** When PBWIDTH = 1, the data FIFO will be increased or decreased by two peripheral bus access. This bit is used to select the order of FIFO access operations to meet the 32-bit transmitting/receiving FIFO entries. 0 = Low 16-bit read/write access first. 1 = High 16-bit read/write access first. **Note:** This bit is available while PBWIDTH = 1.
[24]	**PBWIDTH**	**Peripheral Bus Data Width Selection** This bit is used to choice the available data width of APB bus. It must be set to 1 while PDMA function is enable and it is set to 16-bit transmission mode 0 = 32 bits data width. 1 = 16 bits data width. **Note1:** If PBWIDTH=1, the low 16 bits of 32-bit data bus are available. **Note2:** If PBWIDTH=1, the transmitting FIFO level will be increased after two FIFO write operations. **Note3:** If PBWIDTH=1, the receiving FIFO level will be decreased after two FIFO read operations.
[23:20]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

[19:16]	**RXTH**	**Receive FIFO Threshold Level** 0000 = 1 data word in receive FIFO. 0001 = 2 data words in receive FIFO. 0010 = 3 data words in receive FIFO. …. 1110 = 15 data words in receive FIFO. 1111 = 16 data words in receive FIFO. **Note:** When received data word number in receive buffer is higher than threshold level then RXTHIF (I2S_STATUS0[10]) flag is set.
[15:12]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[11:8]	**TXTH**	**Transmit FIFO Threshold Level** 0000 = 0 data word in transmit FIFO. 0001 = 1 data word in transmit FIFO. 0010 = 2 data words in transmit FIFO. .... 1110 = 14 data words in transmit FIFO. 1111 = 15 data words in transmit FIFO. **Note:** If remain data word number in transmit FIFO is equal to or lower than threshold level then TXTHIF (I2S_STATUS0[18]) flag is set.
[7]	**CH7ZCEN**	**Channel7 Zero-cross Detect Enable Control** 0 = channel7 zero-cross detect Disabled. 1 = channel7 zero-cross detect Enabled. **Note1:** This bit is available while multi-channel PCM mode and TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3. **Note2:** If this bit is set to 1, when channel7 data sign bit change or next shift data bits are all zero then CH7ZCIF (I2S_STATUS1[7]) flag is set to 1. **Note3:** If CH7ZCIF Flag is set to 1, the channel7 will be mute.
[6]	**CH6ZCEN**	**Channel6 Zero-cross Detect Enable Control** 0 = channel6 zero-cross detect Disabled. 1 = channel6 zero-cross detect Enabled. **Note1:** This bit is available while multi-channel PCM mode and TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3. **Note2:** If this bit is set to 1, when channel6 data sign bit change or next shift data bits are all zero then CH6ZCIF(I2S_STATUS1[6]) flag is set to 1. **Note3:** If CH6ZCIF Flag is set to 1, the channel6 will be mute.
[5]	**CH5ZCEN**	**Channel5 Zero-cross Detect Enable Control** 0 = channel5 zero-cross detect Disabled. 1 = channel5 zero-cross detect Enabled. **Note1:** This bit is available while multi-channel PCM mode and TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3. **Note2:** If this bit is set to 1, when channel5 data sign bit change or next shift data bits are all zero then CH5ZCIF(I2S_STATUS1[5]) flag is set to 1. **Note3:** If CH5ZCIF Flag is set to 1, the channel5 will be mute.

[4]	**CH4ZCEN**	**Channel4 Zero-cross Detect Enable Control** 0 = channel4 zero-cross detect Disabled. 1 = channel4 zero-cross detect Enabled. **Note1:** This bit is available while multi-channel PCM mode and TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3. **Note2:** If this bit is set to 1, when channel4 data sign bit change or next shift data bits are all zero then CH4ZCIF(I2S_STATUS1[4]) flag is set to 1. **Note3:** If CH4ZCIF Flag is set to 1, the channel4 will be mute.
[3]	**CH3ZCEN**	**Channel3 Zero-cross Detect Enable Control** 0 = channel3 zero-cross detect Disabled. 1 = channel3 zero-cross detect Enabled. **Note1:** This bit is available while multi-channel PCM mode and TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3. **Note2:** If this bit is set to 1, when channel3 data sign bit change or next shift data bits are all zero then CH3ZCIF(I2S_STATUS1[3]) flag is set to 1. **Note3:** If CH3ZCIF Flag is set to 1, the channel3 will be mute.
[2]	**CH2ZCEN**	**Channel2 Zero-cross Detect Enable Control** 0 = channel2 zero-cross detect Disabled. 1 = channel2 zero-cross detect Enabled. **Note1:** This bit is available while multi-channel PCM mode and TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3. **Note2:** If this bit is set to 1, when channel2 data sign bit change or next shift data bits are all zero then CH2ZCIF(I2S_STATUS1[2]) flag is set to 1. **Note3:** If CH2ZCIF Flag is set to 1, the channel2 will be mute.
[1]	**CH1ZCEN**	**Channel1 Zero-cross Detect Enable Control** 0 = channel1 zero-cross detect Disabled. 1 = channel1 zero-cross detect Enabled. **Note1:** Channel1 also means right audio channel while I2S (FORMAT[2]=0) or 2-channel PCM mode. **Note2:** If this bit is set to 1, when channel1 data sign bit change or next shift data bits are all zero then CH1ZCIF(I2S_STATUS1[1]) flag is set to 1. **Note3:** If CH1ZCIF Flag is set to 1, the channel1 will be mute.
[0]	**CH0ZCEN**	**Channel0 Zero-cross Detection Enable Control** 0 = channel0 zero-cross detect Disabled. 1 = channel0 zero-cross detect Enabled. **Note1:** Channel0 also means left audio channel while I2S (FORMAT[2]=0) or 2-channel PCM mode. **Note2:** If this bit is set to 1, when channel0 data sign bit change or next shift data bits are all zero then CH0ZCIF(I2S_STATUS1[0]) flag is set to 1. **Note3:** If CH0ZCIF Flag is set to 1, the channel0 will be mute.

I2S Clock Divider (I2S_CLKDIV)

Register	Offset	R/W	Description	Reset Value
I2S_CLKDIV	I2S_BA+0x04	R/W	I^2S Clock Divider Register	0x0000_0000

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	
[31:18]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[17:8]	**BCLKDIV**	**Bit Clock Divider** The I^2S controller will generate bit clock in Master mode. Software can program these bit fields to generate sampling rate clock frequency. F_BCLK= F_I2SCLK / (2*(BCLKDIV + 1)). **Note:** F_BCLK is the frequency of BCLK and F_I2SCLK is the frequency of I2S_CLK
[7]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[6:0]	**MCLKDIV**	**Master Clock Divider** If chip external crystal frequency is (2xMCLKDIV)*256fs then software can program these bits to generate 256fs clock frequency to audio codec chip. If MCLKDIV is set to 0, MCLK is the same as external clock input. For example, sampling rate is 24 kHz and chip external crystal clock is 12.288 MHz, set MCLKDIV = 1. F_MCLK = F_I2SCLK/(2x(MCLKDIV)) (When MCLKDIV is >= 1). F_MCLK = F_I2SCLK (When MCLKDIV is set to 0). **Note:** F_MCLK is the frequency of MCLK, and F_I2SCLK is the frequency of the I2S_CLK

I2S Interrupt Enable Register (I2S_IEN)

Register	Offset	R/W	Description	Reset Value
I2S_IEN	I2S_BA+0x08	R/W	I^2S Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
CH7ZCIEN	CH6ZCIEN	CH5ZCIEN	CH4ZCIEN	CH3ZCIEN	CH2ZCIEN	CH1ZCIEN	CH0ZCIEN
15	14	13	12	11	10	9	8
Reserved					TXTHIEN	TXOVFIEN	TXUDFIEN
7	6	5	4	3	2	1	0
Reserved					RXTHIEN	RXOVFIEN	RXUDFIEN

Bits	Description	
[31:24]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[23]	**CH7ZCIEN**	**Channel7 Zero-cross Interrupt Enable Control** 0 = Interrupt Disabled. 1 = Interrupt Enabled. **Note1:** Interrupt occurs if this bit is set to 1 and channel7 zero-cross **Note2:** This bit is available while multi-channel PCM mode and TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3.
[22]	**CH6ZCIEN**	**Channel6 Zero-cross Interrupt Enable Control** 0 = Interrupt Disabled. 1 = Interrupt Enabled. **Note1:** Interrupt occurs if this bit is set to 1 and channel6 zero-cross **Note2:** This bit is available while multi-channel PCM mode and TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3.
[21]	**CH5ZCIEN**	**Channel5 Zero-cross Interrupt Enable Control** 0 = Interrupt Disabled. 1 = Interrupt Enabled. **Note1:** Interrupt occurs if this bit is set to 1 and channel5 zero-cross **Note2:** This bit is available while multi-channel PCM mode and TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3.
[20]	**CH4ZCIEN**	**Channel4 Zero-cross Interrupt Enable Control** 0 = Interrupt Disabled. 1 = Interrupt Enabled. **Note1:** Interrupt occurs if this bit is set to 1 and channel4 zero-cross **Note2:** This bit is available while multi-channel PCM mode and TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3.

[19]	**CH3ZCIEN**	**Channel3 Zero-cross Interrupt Enable Control** 0 = Interrupt Disabled. 1 = Interrupt Enabled. **Note1:** Interrupt occurs if this bit is set to 1 and channel3 zero-cross **Note2:** This bit is available while multi-channel PCM mode and TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3.
[18]	**CH2ZCIEN**	**Channel2 Zero-cross Interrupt Enable Control** 0 = Interrupt Disabled. 1 = Interrupt Enabled. **Note1:** Interrupt occurs if this bit is set to 1 and channel2 zero-cross **Note2:** This bit is available while multi-channel PCM mode and TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3.
[17]	**CH1ZCIEN**	**Channel1 Zero-cross Interrupt Enable Control** 0 = Interrupt Disabled. 1 = Interrupt Enabled. **Note1:** Interrupt occurs if this bit is set to 1 and channel1 zero-cross **Note2:** Channel1 also means right audio channel while I2S (FORMAT[2]=0) or 2-channel PCM mode.
[16]	**CH0ZCIEN**	**Channel0 Zero-cross Interrupt Enable Control** 0 = Interrupt Disabled. 1 = Interrupt Enabled. **Note1:** Interrupt occurs if this bit is set to 1 and channel0 zero-cross **Note2:** Channel0 also means left audio channel while I2S (FORMAT[2]=0) or 2-channel PCM mode.
[15:11]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[10]	**TXTHIEN**	**Transmit FIFO Threshold Level Interrupt Enable Control** 0 = Interrupt Disabled. 1 = Interrupt Enabled. **Note:** Interrupt occurs if this bit is set to 1 and data words in transmit FIFO is equal to or lower than TXTH (I2S_CTL1[11:8]).
[9]	**TXOVFIEN**	**Transmit FIFO Overflow Interrupt Enable Control** 0 = Interrupt Disabled. 1 = Interrupt Enabled. **Note:** Interrupt occurs if this bit is set to 1 and TXOVIF (I2S_STATUS0[17]) flag is set to 1
[8]	**TXUDFIEN**	**Transmit FIFO Underflow Interrupt Enable Control** 0 = Interrupt Disabled. 1 = Interrupt Enabled. **Note:** Interrupt occur if this bit is set to 1 and TXUDIF (I2S_STATUS0[16]) flag is set to 1.
[7:3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**RXTHIEN**	**Receive FIFO Threshold Level Interrupt Enable Control** 0 = Interrupt Disabled. 1 = Interrupt Enabled. **Note:** When data word in receive FIFO is higher than RXTH (I2S_CTL1[19:16]) and the RXTHIF (I2S_STATUS0[10]) bit is set to 1. If RXTHIEN bit is enabled, interrupt occur.

[1]	**RXOVFIEN**	**Receive FIFO Overflow Interrupt Enable Control** 0 = Interrupt Disabled. 1 = Interrupt Enabled. **Note:** Interrupt occurs if this bit is set to 1 and RXOVIF (I2S_STATUS0[9]) flag is set to 1
[0]	**RXUDFIEN**	**Receive FIFO Underflow Interrupt Enable Control** 0 = Interrupt Disabled. 1 = Interrupt Enabled. **Note:** If software reads receive FIFO when it is empty then RXUDIF (I2S_STATUS0[8]) flag is set to 1.

I2S Status Register 0 (I2S_STATUS0)

Register	Offset	R/W	Description	Reset Value
I2S_STATUS0	I2S_BA+0x0C	R/W	I²S Status Register 0	0x0014_1000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved		TXBUSY	TXEMPTY	TXFULL	TXTHIF	TXOVIF	TXUDIF
15	14	13	12	11	10	9	8
Reserved			RXEMPTY	RXFULL	RXTHIF	RXOVIF	RXUDIF
7	6	5	4	3	2	1	0
Reserved		DATACH			I2STXINT	I2SRXINT	I2SINT

Bits	Description	
[31:22]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[21]	**TXBUSY**	**Transmit Busy (Read Only)** 0 = Transmit shift buffer is empty. 1 = Transmit shift buffer is busy. **Note:** This bit is cleared to 0 when all data in transmit FIFO and shift buffer is shifted out. And set to 1 when 1st data is load to shift buffer.
[20]	**TXEMPTY**	**Transmit FIFO Empty (Read Only)** This bit reflect data word number in transmit FIFO is zero 0 = Not empty. 1 = Empty.
[19]	**TXFULL**	**Transmit FIFO Full (Read Only)** This bit reflect data word number in transmit FIFO is 16 0 = Not full. 1 = Full.
[18]	**TXTHIF**	**Transmit FIFO Threshold Interrupt Flag (Read Only)** 0 = Data word(s) in FIFO is higher than threshold level. 1 = Data word(s) in FIFO is equal or lower than threshold level. **Note:** When data word(s) in transmit FIFO is equal to or lower than threshold value set in TXTH (I2S_CTL1[11:8]) the TXTHIF bit becomes to 1. It keeps at 1 till TXCNT (I2S_STATUS1[12:8]) is higher than TXTH (I2S_CTL1[11:8]) after software write TXFIFO register.
[17]	**TXOVIF**	**Transmit FIFO Overflow Interrupt Flag** 0 = No overflow. 1 = Overflow. **Note1:** Write data to transmit FIFO when it is full and this bit set to 1 **Note2:** Write 1 to clear this bit to 0.

[16]	**TXUDIF**	**Transmit FIFO Underflow Interrupt Flag** 0 = No underflow. 1 = Underflow. **Note1:** This bit will be set to 1 when shift logic hardware read data from transmitting FIFO and the filling data level in transmitting FIFO is not enough for one audio frame. **Note2:** Write 1 to clear this bit to 0.
[15:13]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[12]	**RXEMPTY**	**Receive FIFO Empty (Read Only)** 0 = Not empty. 1 = Empty. **Note:** This bit reflects data words number in receive FIFO is zero
[11]	**RXFULL**	**Receive FIFO Full (Read Only)** 0 = Not full. 1 = Full. **Note:** This bit reflects data words number in receive FIFO is 16.
[10]	**RXTHIF**	**Receive FIFO Threshold Interrupt Flag (Read Only)** 0 = Data word(s) in FIFO is not higher than threshold level. 1 = Data word(s) in FIFO is higher than threshold level. **Note:** When data word(s) in receive FIFO is higher than threshold value set in RXTH (I2S_CTL1[19:16]) the RXTHIF bit becomes to 1. It keeps at 1 till RXCNT (I2S_STATUS1[20:16]) is not higher than RXTH (I2S_CTL1[19:16]) after software read RXFIFO register.
[9]	**RXOVIF**	**Receive FIFO Overflow Interrupt Flag** 0 = No overflow occur. 1 = Overflow occur. **Note1:** When receive FIFO is full and receive hardware attempt to write data into receive FIFO then this bit is set to 1, data in 1st buffer is overwrote. **Note2:** Write 1 to clear this bit to 0.
[8]	**RXUDIF**	**Receive FIFO Underflow Interrupt Flag** 0 = No underflow occur. 1 = Underflow occur. **Note1:** When receive FIFO is empty, and software reads the receive FIFO again. This bit will be set to 1, and it indicates underflow situation occurs. **Note2:** Write 1 to clear this bit to zero
[7:6]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[5:3]	**DATACH**	**Transmission Data Channel (Read Only)** This bit fields are used to indicate which audio channel is current transmit data belong. 000 = channel0 (means left channel while 2-channel I2S/PCM mode). 001 = channel1 (means right channel while 2-channel I2S/PCM mode). 010 = channel2 (available while 4-channel TDM PCM mode). 011 = channel3 (available while 4-channel TDM PCM mode). 100 = channel4 (available while 6-channel TDM PCM mode). 101 = channel5 (available while 6-channel TDM PCM mode). 110 = channel6 (available while 8-channel TDM PCM mode). 111 = channel7 (available while 8-channel TDM PCM mode).

[2]	**I2STXINT**	**I²S Transmit Interrupt (Read Only)** 0 = No transmit interrupt. 1 = Transmit interrupt.
[1]	**I2SRXINT**	**I²S Receive Interrupt (Read Only)** 0 = No receive interrupt. 1 = Receive interrupt.
[0]	**I2SINT**	**I²S Interrupt Flag (Read Only)** 0 = No I²S interrupt. 1 = I²S interrupt. **Note:** It is wire-OR of I2STXINT and I2SRXINT bits.

I2S Status Register 1 (I2S_STATUS1)

Register	Offset	R/W	Description	Reset Value
I2S_STATUS1	I2S_BA+0x24	R/W	I^2S Status Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved			**RXCNT**				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			**TXCNT**				
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CH7ZCIF	**CH6ZCIF**	**CH5ZCIF**	**CH4ZCIF**	**CH3ZCIF**	**CH2ZCIF**	**CH1ZCIF**	**CH0ZCIF**

Bits	Description	
[31:21]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[20:16]	**RXCNT**	**Receive FIFO Level (Read Only)** These bits indicate the number of available entries in receive FIFO 00000 = No data. 00001 = 1 word in receive FIFO. 00010 = 2 words in receive FIFO. …. 01110 = 14 words in receive FIFO. 01111 = 15 words in receive FIFO. 10000 = 16 words in receive FIFO. Others are reserved. Do not use.
[15:13]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[12:8]	**TXCNT**	**Transmit FIFO Level (Read Only)** These bits indicate the number of available entries in transmit FIFO 00000 = No data. 00001 = 1 word in transmit FIFO. 00010 = 2 words in transmit FIFO. …. 01110 = 14 words in transmit FIFO. 01111 = 15 words in transmit FIFO. 10000 = 16 words in transmit FIFO. Others are reserved. Do not use.

[7]	**CH7ZCIF**	**Channel7 Zero-cross Interrupt Flag** It indicates channel7 next sample data sign bit is changed or all data bits are zero. 0 = No zero-cross in channel7. 1 = Channel7 zero-cross is detected. **Note1:** Write 1 to clear this bit to 0. **Note2:** This bit is available while multi-channel PCM mode and TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3.
[6]	**CH6ZCIF**	**Channel6 Zero-cross Interrupt Flag** It indicates channel6 next sample data sign bit is changed or all data bits are zero. 0 = No zero-cross in channel6. 1 = Channel6 zero-cross is detected. **Note1:** Write 1 to clear this bit to 0. **Note2:** This bit is available while multi-channel PCM mode and TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3.
[5]	**CH5ZCIF**	**Channel5 Zero-cross Interrupt Flag** It indicates channel5 next sample data sign bit is changed or all data bits are zero. 0 = No zero-cross in channel5. 1 = Channel5 zero-cross is detected. **Note1:** Write 1 to clear this bit to 0. **Note2:** This bit is available while multi-channel PCM mode and TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3.
[4]	**CH4ZCIF**	**Channel4 Zero-cross Interrupt Flag** It indicates channel4 next sample data sign bit is changed or all data bits are zero. 0 = No zero-cross in channel4. 1 = Channel4 zero-cross is detected. **Note1:** Write 1 to clear this bit to 0. **Note2:** This bit is available while multi-channel PCM mode and TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3.
[3]	**CH3ZCIF**	**Channel3 Zero-cross Interrupt Flag** It indicates channel3 next sample data sign bit is changed or all data bits are zero. 0 = No zero-cross in channel3. 1 = Channel3 zero-cross is detected. **Note1:** Write 1 to clear this bit to 0. **Note2:** This bit is available while multi-channel PCM mode and TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3.
[2]	**CH2ZCIF**	**Channel2 Zero-cross Interrupt Flag** It indicates channel2 next sample data sign bit is changed or all data bits are zero. 0 = No zero-cross in channel2. 1 = Channel2 zero-cross is detected. **Note1:** Write 1 to clear this bit to 0. **Note2:** This bit is available while multi-channel PCM mode and TDMCHNUM (I2S_CTL0[31:30]) = 0x1, 0x2, 0x3.
[1]	**CH1ZCIF**	**Channel1 Zero-cross Interrupt Flag** It indicates channel1 next sample data sign bit is changed or all data bits are zero. 0 = No zero-cross in channel1. 1 = Channel1 zero-cross is detected. **Note1:** Write 1 to clear this bit to 0. **Note2:** Channel1 also means right audio channel while I2S (FORMAT[2]=0) or 2-channel PCM mode.

[0]	**CH0ZCIF**	**Channel0 Zero-cross Interrupt Flag** It indicates channel0 next sample data sign bit is changed or all data bits are zero. 0 = No zero-cross in channel0. 1 = Channel0 zero-cross is detected. **Note1:** Write 1 to clear this bit to 0. **Note2:** Channel0 also means left audio channel while I2S (FORMAT[2]=0) or 2-channel PCM mode.

I2S Transmit FIFO (I2S_TXFIFO)

Register	Offset	R/W	Description	Reset Value
I2S_TXFIFO	I2S_BA+0x10	W	I^2S Transmit FIFO Register	0x0000_0000

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

Bits	Description	
[31:0]	**TXFIFO**	**Transmit FIFO Bits** I^2S contains 16 words (16x32 bit) data buffer for data transmit. Write data to this register to prepare data for transmit. The remaining word number is indicated by TXCNT (I2S_STATUS1[12:8]).

I2S Receive FIFO (I2S_RXFIFO)

Register	Offset	R/W	Description	Reset Value
I2S_RXFIFO	I2S_BA+0x14	R	I^2S Receive FIFO Register	0x0000_0000

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

Bits	Description	
[31:0]	**RXFIFO**	**Receive FIFO Bits** I^2S contains 16 words (16x32 bit) data buffer for data receive. Read this register to get data in FIFO. The remaining data word number is indicated by RXCNT (I2S_STATUS1[20:16]).

6.18 USB 1.1 Device Controller (USBD)

6.18.1 Overview

There is one set of USB 2.0 full-speed device controller and transceiver in this device. It is compliant with USB 2.0 full-speed device specification and supports control/bulk/interrupt/isochronous transfer types.

In this device controller, there are two main interfaces: 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 1Kbytes 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_BUFSEGx).

There are 12 endpoints in this controller. Each of the endpoint can be configured as IN or OUT endpoint. All the operations including Control, Bulk, Interrupt and Isochronous transfer are implemented in this block. The block of “Endpoint Control” is also used to manage the data sequential synchronization, endpoint states, current start address, transaction status, and data buffer status for each endpoint.

There are four different interrupt events in this controller. They are the 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_EPSTS0 and USBD_EPSTS1) 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.*

6.18.2 Features

- 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 12 endpoints for configurable Control/Bulk/Interrupt/Isochronous transfer types and maximum 1Kbytes buffer size
- Provides remote wake-up capability

6.18.3 Block Diagram

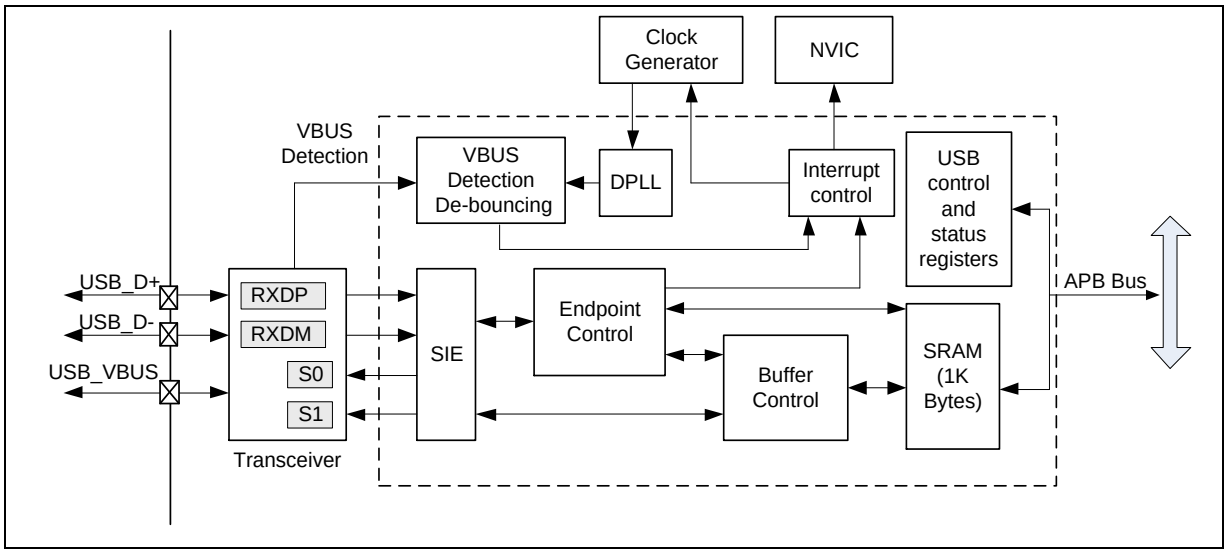


Figure 6.18-1 USB Block Diagram

6.18.4 Basic Configuration

The role of USB frame is determined by USBROLE (SYS_USBPHY[1:0]). The internal USB 3.3V LDO can be enabled by LDO33EN (SYS_USBPHY[8]). These two configurations are write-protection bits. Before writing to these bits, user must disable the register protection function. Refer to the description of SYS_REGLCTL register for details. The USBD clock source is derived from PLL. User has to set the PLL related configurations before USB device controller is enabled. 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.

- Clock source configuration
  - Setting PLL controller (CLK_PLLCTL).
  - Select the clock divider number of USBD peripheral clock on USBDIV (CLK_CLKDIV0[7:4]).
  - Enable USBD peripheral clock in USBDCKEN (CLK_APBCLK0[27]).
- Reset configuration
  - Reset USBD controller in USBDRST (SYS_IPRST1[27]).
- Pin configuration

Group	Pin Name	GPIO	MFP
USB	USB_D+	PB.13	MFP1
	USB_D-	PB.14	MFP1
	USB_VBUS	PB.15	MFP1

Note: If GPIO configure as USB function, the register PB_MODE[31:26] should be set to 0 and PUSEL15 (PB_PUSEL[31:30]) should be set to 2.

6.18.5 Functional Description

6.18.5.1 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
- 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

6.18.5.2 Endpoint Control

This controller supports 12 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.

6.18.5.3 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.

6.18.5.4 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.

6.18.5.5 Interrupt control

This USB provides 1 interrupt vector with 4 interrupt events (NEVWK, VBUSDET, USB and BUS). 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 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. The Figure 6.18-2 is the control flow of wake-up interrupt.

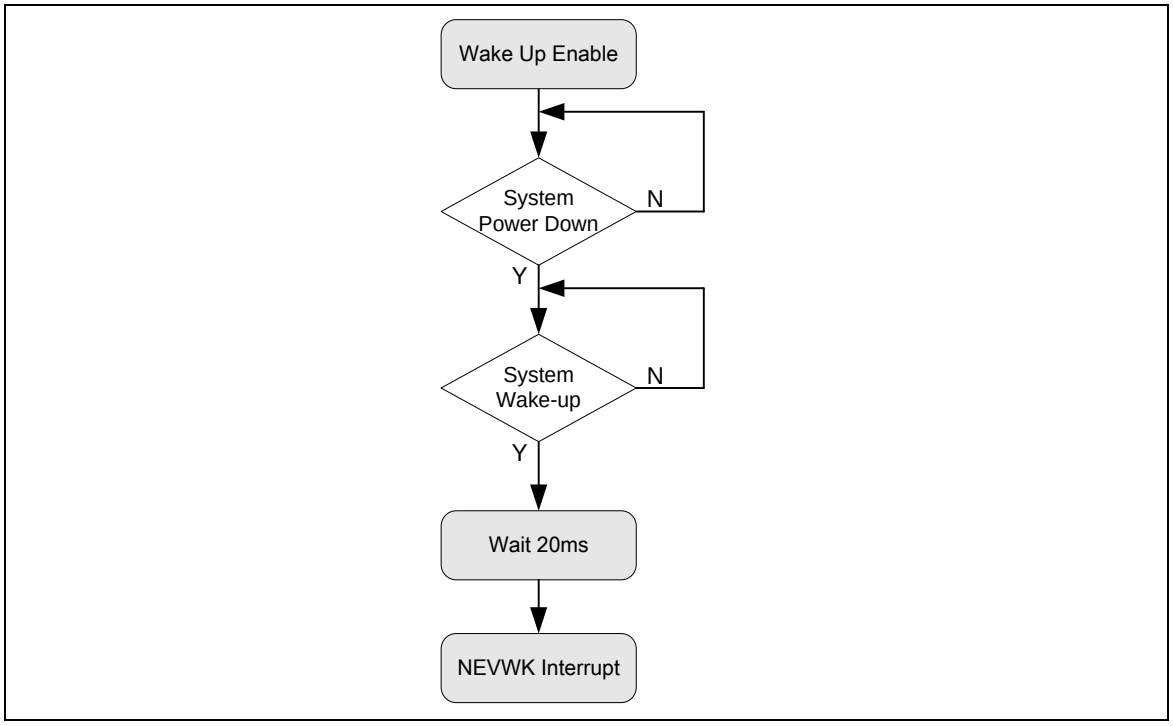


Figure 6.18-2 NEVWK Interrupt Operation Flow

The USB interrupt is used to notify users of any USB event on the bus, and user can read EPSTS (USBD_EPSTS0 and USBD_EPSTS1) and EPEVT11~0 (USBD_INTSTS[27:16]) 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.

6.18.5.6 Power Saving

User can write 0 to USBD_ATTR[4] to disable PHY under special circumstances, like suspend, to conserve power.

6.18.5.7 Buffer Control

There is 1Kbytes SRAM in the controller and the 12 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_MXPLDx register.

Figure 6.18-3 depicts the starting address for each endpoint according the content of USBD_BUFSEGx and USBD_MXPLDx registers. If the USBD_BUFSEG0 is programmed as 0x08h and USBD_MXPLD0 is set as 0x40h, the SRAM size of endpoint 0 is start from USBD_BA+0x108h and end in USBD_BA+0x148h. (**Note:** The USBD SRAM base is USBD_BA+0x100h).

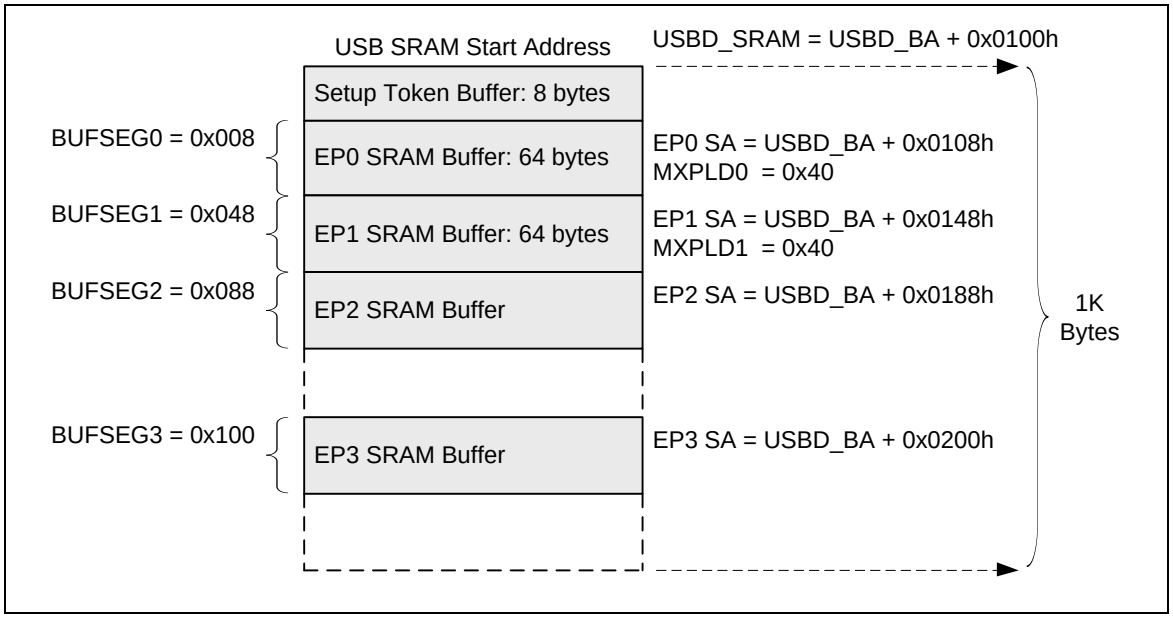


Figure 6.18-3 Endpoint SRAM Structure

6.18.5.8 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_MXPLDx 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.

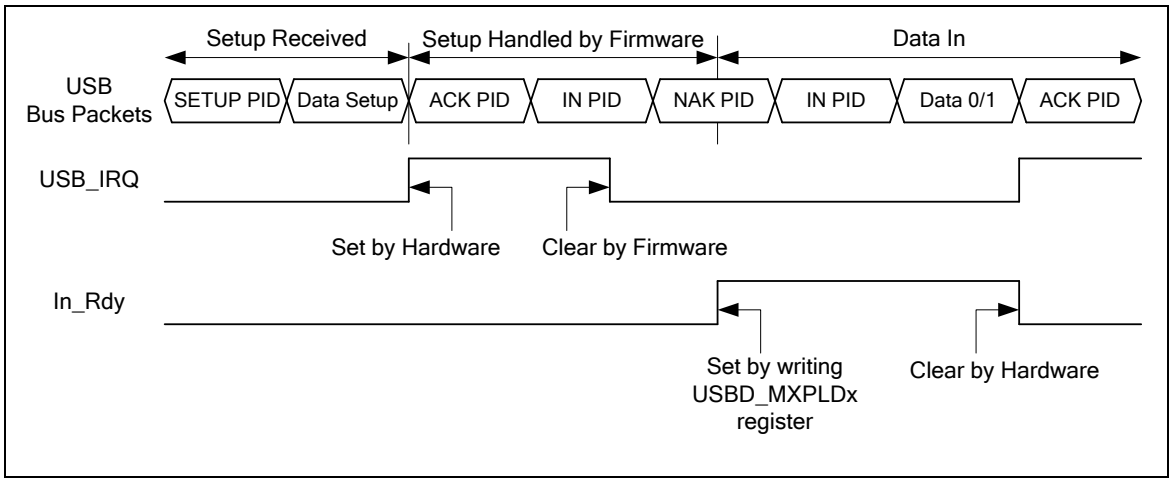


Figure 6.18-4 Setup Transaction Followed by Data IN Transaction

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_MXPLDx 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_MXPLDx register needs to be written by firmware to assert the signal “Out_Rdy” again to accept the next transaction.

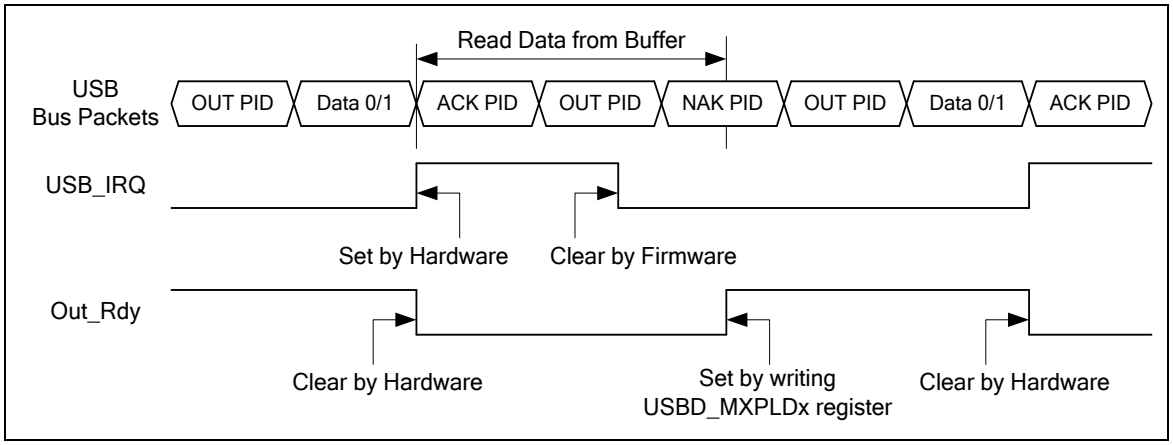


Figure 6.18-5 Data Out Transfer

6.18.6 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 = 0x400C_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_EPSTS0	USBD_BA+0x020	R	USB Device Endpoint Status Register 0	0x0000_0000
USBD_EPSTS1	USBD_BA+0x024	R	USB Device Endpoint Status Register 1	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

Register	Offset	R/W	Description	Reset Value
USBD Base Address: **USBD_BA = 0x400C_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
USBD_BUFSEG8	USBD_BA+0x580	R/W	Endpoint 8 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD8	USBD_BA+0x584	R/W	Endpoint 8 Maximal Payload Register	0x0000_0000
USBD_CFG8	USBD_BA+0x588	R/W	Endpoint 8 Configuration Register	0x0000_0000
USBD_CFGP8	USBD_BA+0x58C	R/W	Endpoint 8 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG9	USBD_BA+0x590	R/W	Endpoint 9 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD9	USBD_BA+0x594	R/W	Endpoint 9 Maximal Payload Register	0x0000_0000
USBD_CFG9	USBD_BA+0x598	R/W	Endpoint 9 Configuration Register	0x0000_0000
USBD_CFGP9	USBD_BA+0x59C	R/W	Endpoint 9 Set Stall and Clear In/Out Ready Control Register	0x0000_0000

Register	Offset	R/W	Description	Reset Value
USBD Base Address: **USBD_BA = 0x400C_0000**				
USBD_BUFSEG10	USBD_BA+0x5A0	R/W	Endpoint 10 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD10	USBD_BA+0x5A4	R/W	Endpoint 10 Maximal Payload Register	0x0000_0000
USBD_CFG10	USBD_BA+0x5A8	R/W	Endpoint 10 Configuration Register	0x0000_0000
USBD_CFGP10	USBD_BA+0x5AC	R/W	Endpoint 10 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG11	USBD_BA+0x5B0	R/W	Endpoint 11 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD11	USBD_BA+0x5B4	R/W	Endpoint 11 Maximal Payload Register	0x0000_0000
USBD_CFG11	USBD_BA+0x5B8	R/W	Endpoint 11 Configuration Register	0x0000_0000
USBD_CFGP11	USBD_BA+0x5BC	R/W	Endpoint 11 Set Stall and Clear In/Out Ready Control Register	0x0000_0000

Note:

1. Any register not listed here is reserved and must not be written. The result of a read operation on these bits is undefined.
2. The reserved register fields that listed in register description must be written to their reset value. Writing reserved fields with other than reset values may produce undefined results.

Memory Type	Address	Size	Description
USBD_BA = 0x400C_0000			
USBD_SRAM	USBD_BA+0x100 ~ USBD_BA+0x4FF	1024 Bytes	The SRAM is used for the entire endpoints buffer. Refer to section 6.18.5.7 for the endpoint SRAM structure and its description.

6.18.7 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. Any values read should be ignored. When writing to this field always write with reset value.
[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_EPSTS0 and USBD_EPSTS1, so that the USB interrupt event will not be asserted. 1 = IN NAK status will be updated to USBD_EPSTS0 and USBD_EPSTS1 and the USB interrupt event will be asserted, when the device responds NAK after receiving IN token.
[14:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8]	**WKEN**	**Wake-up Function Enable Bit** 0 = USB wake-up function Disabled. 1 = USB wake-up function Enabled.
[7:5]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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			EPEVT11	EPEVT10	EPEVT9	EPEVT8
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 writing 1 to USBD_INTSTS[31].
[30:28]	Reserved	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[27]	EPEVT11	**Endpoint 11's USB Event Status** 0 = No event occurred in endpoint 11. 1 = USB event occurred on Endpoint 11, check USBD_EPSTS1[15:12] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[27] or USBD_INTSTS[1].
[26]	EPEVT10	**Endpoint 10's USB Event Status** 0 = No event occurred in endpoint 10. 1 = USB event occurred on Endpoint 10, check USBD_EPSTS1[11 :8] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[26] or USBD_INTSTS[1].
[25]	EPEVT9	**Endpoint 9's USB Event Status** 0 = No event occurred in endpoint 9. 1 = USB event occurred on Endpoint 9, check USBD_EPSTS1[7 :4] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[25] or USBD_INTSTS[1].
[24]	EPEVT8	**Endpoint 8's USB Event Status** 0 = No event occurred in endpoint 8. 1 = USB event occurred on Endpoint 8, check USBD_EPSTS1[3 :0] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[24] or USBD_INTSTS[1].
[23]	EPEVT7	**Endpoint 7's USB Event Status** 0 = No event occurred in endpoint 7. 1 = USB event occurred on Endpoint 7, check USBD_EPSTS0[31:28] to know which kind of USB event was occurred, cleared by writing 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_EPSTS0[27:24] to know which kind of USB event was occurred, cleared by writing 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_EPSTS0[23:20] to know which kind of USB event was occurred, cleared by writing 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_EPSTS0[19:16] to know which kind of USB event was occurred, cleared by writing 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_EPSTS0[15:12] to know which kind of USB event was occurred, cleared by writing 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_EPSTS0[11:8] to know which kind of USB event was occurred, cleared by writing 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_EPSTS0[7:4] to know which kind of USB event was occurred, cleared by writing 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_EPSTS0[3:0] to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[16] or USBD_INTSTS[1].
[15:5]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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 writing 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 writing 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 EPSTS (USBD_EPSTS0 and USBD_EPSTS1) to know which kind of USB event was occurred, cleared by writing 1 to USBD_INTSTS[1] or EPEVT11~0 (USBD_INTSTS[27:16] and SETUP (USBD_INTSTS[31]).
[0]	**BUSIF**	**BUS Interrupt Status** The BUS event means that there is one of the suspend or the resume function in the bus. 0 = No BUS event occurred. 1 = Bus event occurred; check USBD_ATTR[3:0] to know which kind of bus event was occurred, cleared by writing 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. Any values read should be ignored. When writing to this field always write with reset value.
[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
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
OV	Reserved						

Bits	Description	
[31:8]	Reserved	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.

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							
15	14	13	12	11	10	9	8
Reserved					BYTEM	Reserved	DPPUEN
7	6	5	4	3	2	1	0
USBEN	Reserved	RWAKEUP	PHYEN	TOUT	RESUME	SUSPEND	USBRST

Bits	Description	
[31:11]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.
[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	
[31:9]	Reserved	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8:3]	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 + {STBUFSEG, 3'b000} Where the USBD_SRAM address = USBD_BA+0x100h. **Note:** It is used for SETUP token only.
[2:0]	Reserved	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

USB Endpoint Status Register 0 (USBD_EPSTS0)

Register	Offset	R/W	Description	Reset Value
USBD_EPSTS0	USBD_BA+0x020	R	USB Device Endpoint Status Register 0	0x0000_0000

31	30	29	28	27	26	25	24
EPSTS7				EPSTS6			
23	22	21	20	19	18	17	16
EPSTS5				EPSTS4			
15	14	13	12	11	10	9	8
EPSTS3				EPSTS2			
7	6	5	4	3	2	1	0
EPSTS1				EPSTS0			

Bits	Description	
[31:28]	**EPSTS7**	**Endpoint 7 Status** These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[27:24]	**EPSTS6**	**Endpoint 6 Status** These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[23:20]	**EPSTS5**	**Endpoint 5 Status** These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[19:16]	**EPSTS4**	**Endpoint 4 Status** These bits are used to indicate the current status of this endpoint 0000 = In ACK.

		0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[15:12]	**EPSTS3**	**Endpoint 3 Status** These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[11:8]	**EPSTS2**	**Endpoint 2 Status** These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[7:4]	**EPSTS1**	**Endpoint 1 Status** These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[3:0]	**EPSTS0**	**Endpoint 0 Status** These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.

USB Endpoint Status Register 1 (USBD_EPSTS1)

Register	Offset	R/W	Description	Reset Value
USBD_EPSTS1	USBD_BA+0x024	R	USB Device Endpoint 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
EPSTS11				EPSTS10			
7	6	5	4	3	2	1	0
EPSTS9				EPSTS8			

Bits	Description	
[31:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:12]	**EPSTS11**	**Endpoint 11 Status** These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[11:8]	**EPSTS10**	**Endpoint 10 Status** These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[7:4]	**EPSTS9**	**Endpoint 9 Status** These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[3:0]	**EPSTS8**	**Endpoint 8 Status**

		These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0011 = Setup ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.

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

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

Bits	Description	
[31:11]	Reserved	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.
[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 (USBD_BUFSEGx)

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
USBD_BUFSEG8	USBD_BA+0x580	R/W	Endpoint 8 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG9	USBD_BA+0x590	R/W	Endpoint 9 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG10	USBD_BA+0x5A0	R/W	Endpoint 10 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG11	USBD_BA+0x5B0	R/W	Endpoint 11 Buffer Segmentation Register	0x0000_0000

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

Bits	Description	
[31:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8:3]	**BUFSEG**	**Endpoint Buffer Segmentation** It is used to indicate the offset address for each endpoint with the USB SRAM starting address The effective starting address of the endpoint is USBD_SRAM address + { BUFSEG, 3'b000} Where the USBD_SRAM address = USBD_BA+0x100h. Refer to the section 6.18.5.7 for the endpoint SRAM structure and its description.
[2:0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

USB Maximal Payload Register (USBD_MXPLDx)

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
USBD_MXPLD8	USBD_BA+0x584	R/W	Endpoint 8 Maximal Payload Register	0x0000_0000
USBD_MXPLD9	USBD_BA+0x594	R/W	Endpoint 9 Maximal Payload Register	0x0000_0000
USBD_MXPLD10	USBD_BA+0x5A4	R/W	Endpoint 10 Maximal Payload Register	0x0000_0000
USBD_MXPLD11	USBD_BA+0x5B4	R/W	Endpoint 11 Maximal Payload Register	0x0000_0000

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

Bits	Description	
[31:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8:0]	**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 (USBD_CFGx)

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
USBD_CFG8	USBD_BA+0x588	R/W	Endpoint 8 Configuration Register	0x0000_0000
USBD_CFG9	USBD_BA+0x598	R/W	Endpoint 9 Configuration Register	0x0000_0000
USBD_CFG10	USBD_BA+0x5A8	R/W	Endpoint 10 Configuration Register	0x0000_0000
USBD_CFG11	USBD_BA+0x5B8	R/W	Endpoint 11 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. Any values read should be ignored. When writing to this field always write with reset value.
[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. Any values read should be ignored. When writing to this field always write with reset value.
[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 (USBD_CFGPx)

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
USBD_CFGP8	USBD_BA+0x58C	R/W	Endpoint 8 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP9	USBD_BA+0x59C	R/W	Endpoint 9 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP10	USBD_BA+0x5AC	R/W	Endpoint 10 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP11	USBD_BA+0x5BC	R/W	Endpoint 11 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. Any values read should be ignored. When writing to this field always write with reset value.
[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_MXPLDx 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.

6.19 Digital Microphone Inputs (DMIC)

6.19.1 Overview

Using the dual channel digital PDM (Pulse Density Modulation) microphone interface (DMIC_CLK0, DMIC_DAT0, DMIC_CLK1 and DMIC_DAT1 pins) that are handled four digital PDM microphone inputs. Both DMIC_DAT0 and DMIC_DAT1 inputs are able to handle two digital microphones by selecting them alternately for each half of the clock cycle.

6.19.2 Features

The digital microphone interface use two wires (DMIC_DATn and DMIC_CLKn) to receive information from digital microphones. The main features of DMIC includes:

- Provides one 32-level FIFO data buffers for receiving.
- Generates interrupt requests when buffer levels cross a programmable boundary.
- Supports PDMA transfer.
- Supports up to four channel digital microphones.
- Both digital PDM microphone inputs can be used simultaneously.

6.19.3 Block Diagram

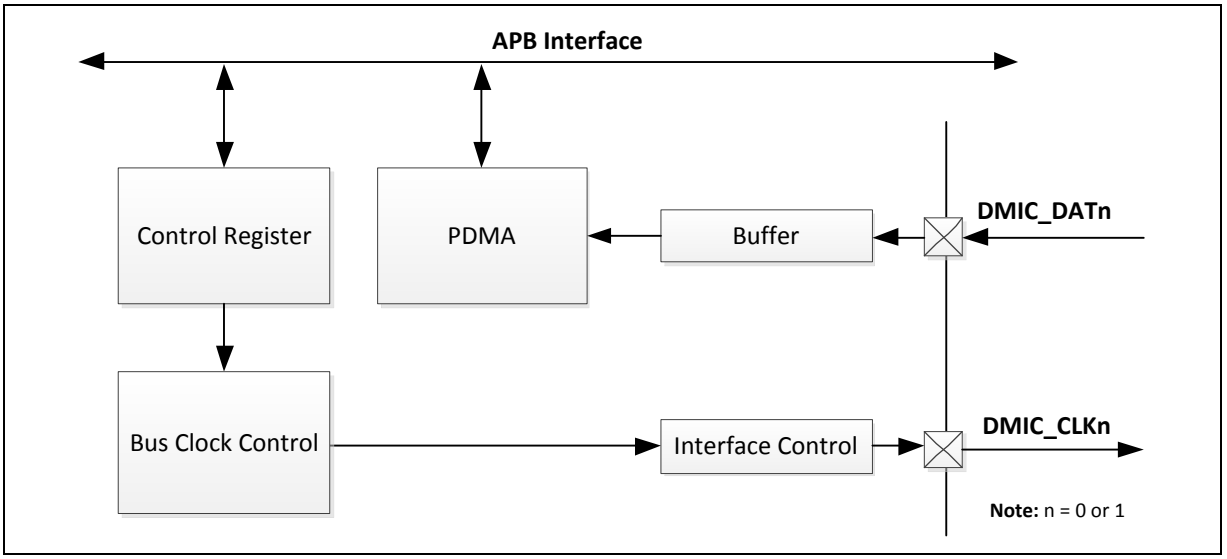


Figure 6.19-1 DMIC Block Diagram

6.19.4 Basic Configuration

- Clock source configuration
  - The source of DMIC peripheral clock is selected by DMICSEL (CLK_CLKSEL2[11:10]).
  - The DMIC peripheral clock is enabled by DMICCKEN (CLK_APBCLK0[15]).
- Reset configuration
  - The DMIC module is reset by DMICRST (SYS_IPRST1[15]).

- Pin configuration

Group	Pin Name	GPIO	MFP
DMIC	DMIC_DAT0	PA.0	MFP3
		PB.5	MFP5
		PD.6	MFP4
	DMIC_CLK0	PA.1	MFP3
		PB.6	MFP5
		PD.5	MFP4
	DMIC_DAT1	PA.2	MFP3
		PB.3	MFP3
		PD.4	MFP4
	DMIC_CLK1	PA.3	MFP3
		PB.4	MFP3
		PD.3	MFP4

6.19.5 Functional Description

6.19.5.1 DMIC Woking Main Clock Generator

The DMIC module has four clock sources selected by DMICSEL (CLK_CLKSEL2[11:10]). The frequency of the DMIC working main clock (DMIC_MCLK) must be less than 30MHz. The DMIC clock control diagram is shown in Figure 6.19-2.

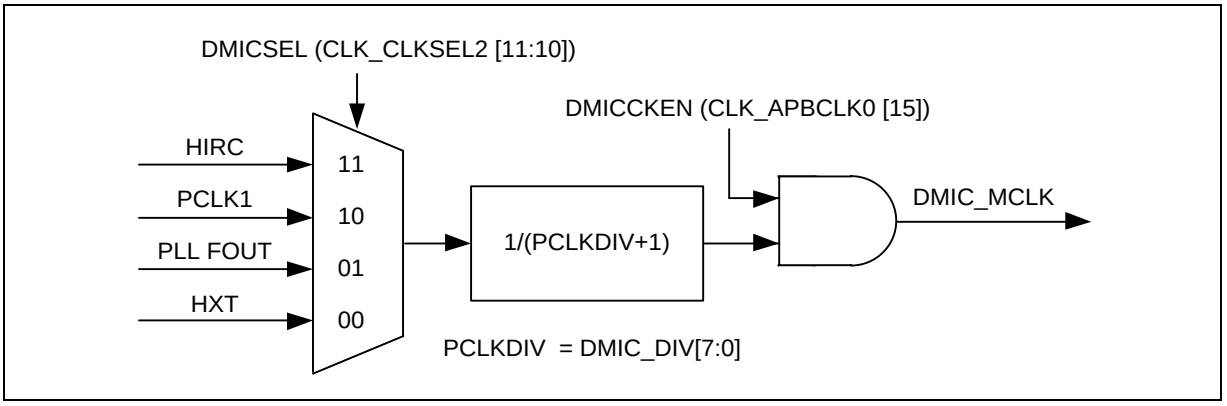


Figure 6.19-2 DMIC Clock Control Diagram

6.19.5.2 DMIC Bus Clock Generator

To generate DMIC bus clock (DMIC_CLK) based on DMIC working main clock (DMIC_MCLK)

F_DMIC_CLK = F_DMIC_MCLK/(1 + MCLKDIV).

Where F_DMIC_CLK is the frequency of DMIC_CLK and F_DMIC_MCLK is the frequency of DMIC_MCLK.

6.19.5.3 Determine DMIC Bus Clock and DMIC Working Main Clock

Determine DMIC working main clock (DMIC_MCLK) by:

F_DMIC_MCLK = Fs * K;

K must be 500 or 256.

Where Fs is sample rate, and F_DMIC_MCLK is the frequency of DMIC_MCLK.

Determine DMIC bus clock (DMIC_CLK) by:

F_DMIC_CLK = Fs * OSR.

Where F_DMIC_CLK is the frequency of DMIC_CLK, and OSR is over sampling rate that is controlled by register OSR (DMIC_CTL[6:4]). The example for DMIC clock and OSR configuring is shown in Table 6.19.5-1.

Fs (Sample Rate)	DMIC_MCLK	DMIC_CLK	OSR (DMIC_CTL[6:4])
48 KHz	24 MHz	2.4 MHz	100 (Down Sample 50)
48 KHz	12.288MHz	3.072 MHz	001 (Down Sample 64)
16 KHz	8 MHz	1.6 MHz	100 (Down Sample 100)
16 KHz	4.096 MHz	2.048 MHz	010 (Down Sample 128)

Table 6.19.5-1 Example for DMIC bus clock and OSR configuring

Note: DMIC_CLK outputs select between the stereo microphones depending on the phase of the clock that has a maximum frequency of 3.25 MHz.

6.19.5.4 DMIC Inputs

NPCA121 series supports up to four channel digital microphones. Two channel of PDM data are multiplexed on the DMIC_DAT0 pin and the other two channel are multiplexed on the DMIC_DAT1 pin. The digital microphones are clocked by DMIC_CLK0 and DMIC_CLK1 pin. The general connection is shown in Figure 6.19-3. The PDM data of channel 0 and channel 1 are latched on DMIC_DAT0 pin, and the PDM data of channel 2 and channel 3 are latched on DMC_DAT1 pin. Each channel can be enabled by register CHENn (DMIC_CTL[3:0]) (n = 0, 1, 2 and 3). User can decide rising or falling edge of DMIC bus clock to latch PDM data for each channel by using register LCHEDGE01 (DMIC_CTL[8]) and LCHEDGE23 (DMIC_CTL[9]). Figure 6.19-4 shows a timing diagram of the two digital microphones sharing DMIC_DAT0 pin.

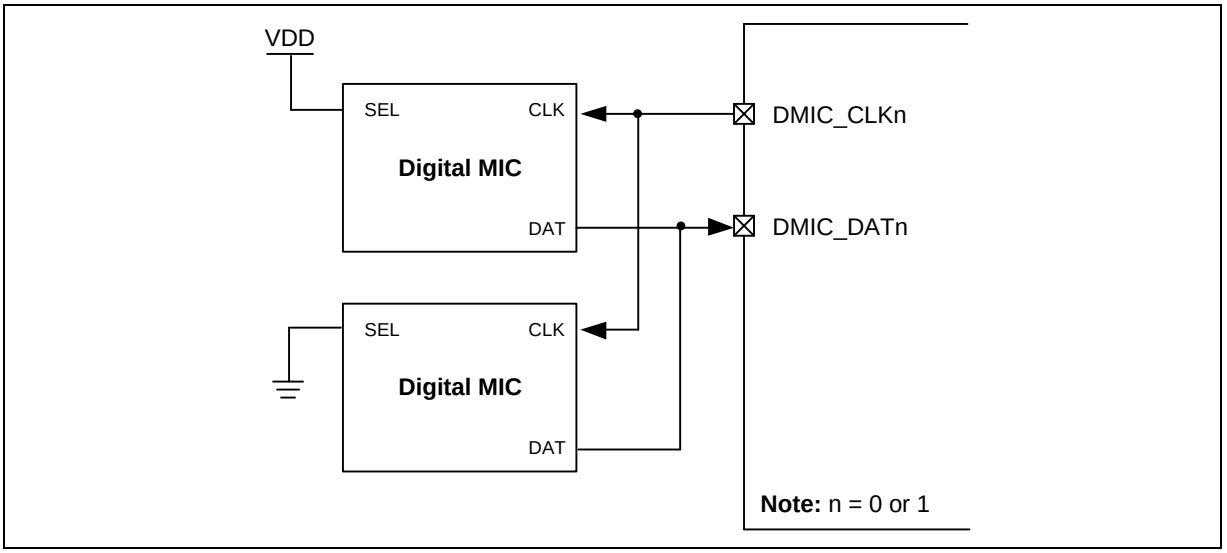


Figure 6.19-3 Typical connection to two digital microphones sharing a common data line

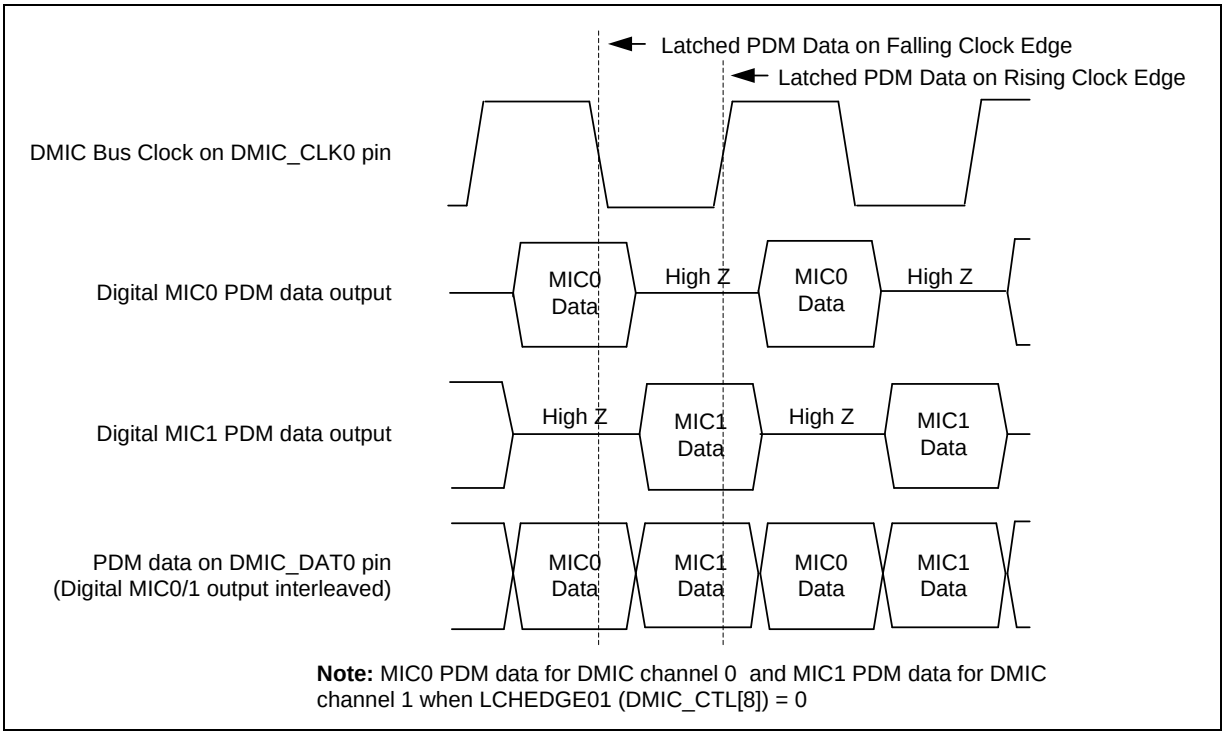


Figure 6.19-4 Digital Microphone Interface Timing Diagram

6.19.5.5 FIFO Operation

FIFO bits is 24 bits. NPCA121 series supports four channel digital microphone inputs, each channel can be enabled by register CHENn (DMIC_CTL[3:0]) (n = 0, 1, 2 and 3). The memory arrangements of PCM data for various settings are shown in Figure 6.19-5.

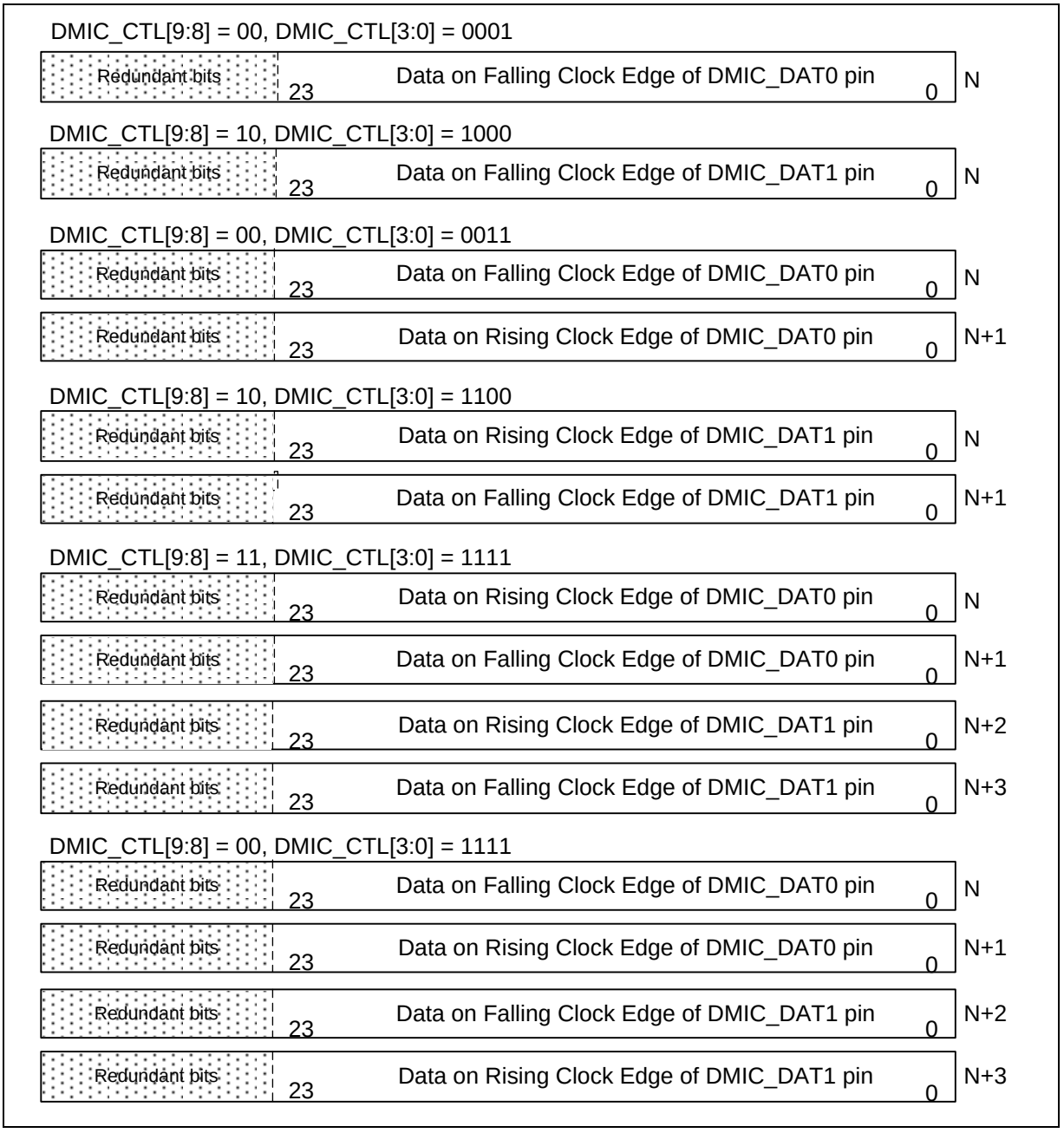


Figure 6.19-5 DMIC FIFO Contents for Various Settings

6.19.5.6 Peripheral DMA Request

Normal use of the DMIC is with PDMA. In this mode DMIC requests PDMA service whenever there is space in FIFO. PDMA channel will copy data from a streaming buffer to the DMIC and alert the CPU when buffer is empty. In this way an entire buffer of data can be sent to DMIC without any CPU intervention.

6.19.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
DMIC Base Address: **DMIC_BA = 0x4006_3000**				
DMIC_CTL	DMIC_BA+0x00	R/W	DMIC Control Register	0xB7CD_0000
DMIC_DIV	DMIC_BA+0x04	R/W	DMIC Clock Divider Register	0x0000_0307
DMIC_STATUS	DMIC_BA+0x08	R	DMIC Status Register	0x0000_0002
DMIC_PDMACTL	DMIC_BA+0x0C	R/W	DMIC PDMA Control Register	0x0000_0000
DMIC_FIFO	DMIC_BA+0x10	W	DMIC FIFO Data Output Register	0x0000_0000

Note:

1. Any register not listed here is reserved and must not be written. The result of a read operation on these bits is undefined.
2. The reserved register fields that listed in register description must be written to their reset value. Writing reserved fields with other than reset values may produce undefined results.

6.19.7 Register Description

DMIC Control Register (DMIC_CTL)

Register	Offset	R/W	Description	Reset Value
DMIC_CTL	DMIC_BA+0x00	R/W	DMIC Control Register	0xB7CD_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						LCHEDGE23	LCHEDGE01
7	6	5	4	3	2	1	0
Reserved	OSR			CHEN3	CHEN2	CHEN1	CHEN0

Bits	Description	
[31:10]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[9]	**LCHEDGE23**	**Channel 23 Data Latch Edge** The data of DMIC channel 2 and channel 3 is latched on DMIC_DATA1 pin. This bit is used to select the data of DMIC channel 2 and channel 3 is latched on rising or falling edge of DMIC_CLK (DMIC bus clock). 0 = The data of channel 2 is latched on falling edge of DMIC_CLK. The data of channel 3 is latched on rising edge of DMIC_CLK. 1 = The data of channel 2 is latched on rising edge of DMIC_CLK. The data of channel 3 is latched on falling edge of DMIC_CLK.
[8]	**LCHEDGE01**	**Channel 01 Data Latch Edge** The data of DMIC channel 0 and channel 1 is latched on DMIC_DATA0 pin. This bit is used to select the data of DMIC channel 0 and channel 1 is latched on rising or falling edge of DMIC_CLK (DMIC bus clock). 0 = The data of channel 0 is latched on falling edge of DMIC_CLK. The data of channel 1 is latched on rising edge of DMIC_CLK. 1 = The data of channel 0 is latched on rising edge of DMIC_CLK. The data of channel 1 is latched on falling edge of DMIC_CLK.
[7]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[6:4]	**OSR**	**OSR Setting** 000 = Down sample 32 001 = Down sample 64 010 = Down sample 128 011 = Down sample 256 100 = Down sample 100 or 50 Others = Reserved. Do not use.
[3]	**CHEN3**	**Channel 3 Enable Bit**

		Set this bit to 1 to enable DMIC channel 3 operation. 0 = Channel 3 Disabled. 1 = Channel 3 Enabled.
[2]	**CHEN2**	**Channel 2 Enable Bit** Set this bit to 1 to enable DMIC channel 2 operation. 0 = Channel 2 Disabled. 1 = Channel 2 Enabled.
[1]	**CHEN1**	**Channel 1 Enable Bit** Set this bit to 1 to enable DMIC channel 1 operation. 0 = Channel 1 Disabled. 1 = Channel 1 Enabled.
[0]	**CHEN0**	**Channel 0 Enable Bit** Set this bit to 1 to enable DMIC channel 0 operation. 0 = Channel 0 Disabled. 1 = Channel 0 Enabled.

DMIC Clock Divider Register (DMIC_DIV)

Register	Offset	R/W	Description	Reset Value
DMIC_DIV	DMIC_BA+0x04	R/W	DMIC Clock Divider Register	0x0000_0307

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

Bits	Description	
[31:24]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[23:22]	**FCLR**	**FIFO Clear** 11 = Clear the FIFO. Others = Reserved. Do not use. **Note 1:** To clear the FIFO, need to write FCLR (DMIC_DIV[23:22]) to 11b, and can read the EMPTY (DMIC_STATUS[1]) bit to make sure that the FIFO has been cleared. **Note 2:** This field is auto cleared by hardware.
[21]	**THIE**	**FIFO Threshold Interrupt** 0 = FIFO threshold interrupt Disabled 1 = FIFO threshold interrupt Enabled.
[20:16]	**TH**	**FIFO Threshold Level** If the valid data count of the FIFO data buffer is more than or equal to TH (DMIC_DIV[20:16]) setting, the THIF (DMIC_STATUS[2]) bit will set to 1, else the THIF (DMIC_STATUS[2]) bit will be cleared to 0.
[15:8]	**MCLKDIV**	**Divider to generate the DMIC Bus Clock** The value in this field is the frequency divider for generating the DMIC bus clock. The frequency is obtained according to the following equation. F_DMIC_CLK = (F_DMIC_MCLK)/(1 + MCLKDIV) where F_DMIC_MCLK is the frequency of DMIC working main clock (DMIC_MCLK) and F_DMIC_CLK is the frequency of DMIC bus clock (DMIC_CLK).
[7:0]	**PCLKDIV**	**Divider to generate the DMIC Working Main Clock** The value in this field is the frequency divider for generating the DMIC working main clock. The frequency is obtained according to the following equation. F_DMIC_MCLK = (F_DMIC_CLK_SRC)/(1 + PCLKDIV) where F_DMIC_CLK_SRC is the frequency of DMIC module clock source, which is defined in the clock control register DMICSEL (CLK_CLKSEL2[11:10]) and F_DMIC_MCLK is the frequency of DMIC working main clock (DMIC_MCLK).

DMIC Status Register (DMIC_STATUS)

Register	Offset	R/W	Description	Reset Value
DMIC_STATUS	DMIC_BA+0x08	R	DMIC Status Register	0x0000_0002

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

Bits	Description	
[31:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8:4]	**FIFOPTR**	**FIFO Pointer (Read Only)** The FULL (DMIC_STATUS[0]) and FIFOPTR (DMIC_STATUS[8:4]) indicates the field that the valid data count within the DMIC FIFO buffer. The maximum value shown in FIFOPTR (DMIC_STATUS[8:4]) is 31. When the using level of DMIC FIFO buffer equal to 32, The FULL (DMIC_STATUS[0]) is set to 1.
[3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**THIF**	**FIFO Threshold Interrupt Status (Read Only)** 0 = The valid data count within the FIFO data buffer is less than the setting value of TH (DMIC_DIV[20:16]). 1 = The valid data count within the FIFO data buffer is more than or equal to the setting value of TH (DMIC_DIV[20:16]).
[1]	**EMPTY**	**FIFO Empty Indicator (Read Only)** 0 = FIFO is not empty. 1 = FIFO is empty.
[0]	**FULL**	**FIFO Full Indicator (Read Only)** 0 = FIFO is not full. 1 = FIFO is full.

DMIC PDMA Control Register(DMIC_PDMACTL)

Register	Offset	R/W	Description	Reset Value
DMIC_PDMACTL	DMIC_BA+0x0C	R/W	DMIC 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**

Bits	Description	
[31:1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**PDMAEN**	**PDMA Transfer Enable Bit** 0 = PDMA data transfer Disabled. 1 = PDMA data transfer Enabled.

DMIC FIFO Data Output Register (DMIC_FIFO)

Register	Offset	R/W	Description	Reset Value
DMIC_FIFO	DMIC_BA+0x10	W	DMIC FIFO Data Output Register	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[23:0]	**FIFO**	**FIFO Data Output Register** DMIC contains 32 words (32x32 bit) data buffer for data receive. A read to this register pushes data out from FIFO data buffer and decrements the read pointer. This is the address that PDMA reads audio data from. The remaining data word number is indicated by FIFOPTR (DMIC_STATUS[8:4]).

6.20 Voice Active Detection (VAD)

6.20.1 Overview

The Voice Active Detection (VAD) analyses the PCM data from DMIC channel 0, and it consists of a SINC filter, a biquad filter and a VAD module. The idea of the VAD is to calculate the short term signal power and long term signal power of the input signal, and then compare the short term power with the short term power threshold. Moreover, the deviation of the short term power and long term power can be calculated and compared with the threshold deviation. Based on these two results, which can determine if the input signal is voice or not.

VAD can be active during idle mode and therefore provide lowest power operation, compared with a software based implementation.

6.20.2 Features

- Configuration detect levels.
- Supports idle mode wake-up function.
- Supports auto switch DMIC path when CPU wake-up by VAD.
- Generates interrupt requests when voice detected.

6.20.3 Block Diagram

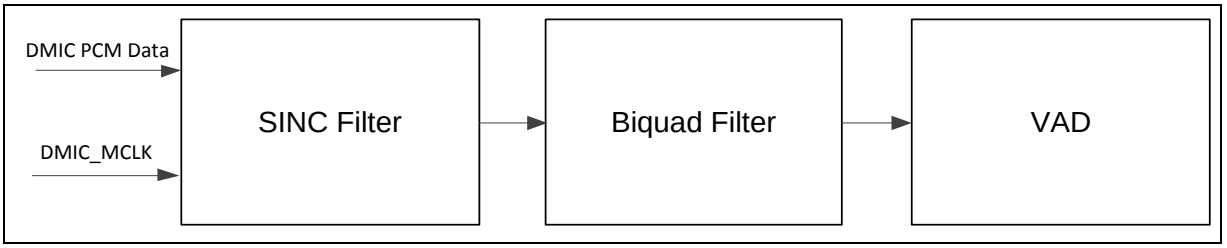


Figure 6.20-1 VAD Block Diagram

6.20.4 Basic Configuration

DMIC module must be enabled with normal operate configuration due to VAD analyses the PCM data that from DMIC channel 0.

6.20.5 Functional Description

6.20.5.1 VAD Clock Control

The VAD clock control diagram is shown in Figure 6.20-2.

When VAD enabled (VADEN (VAD_SINCCTL[31] = 1) and voice detected (ACTIVE (VAD_STATUS0[31]) = 1), the first channel of DMIC will be automatically enabled (DMIC_CTL[3:0] = 1) and the DMIC bus clock (DMIC_CLK) will switch from the VAD path to the DMIC path.

To switch back to the VAD path again, user needs to disable the channels of DMIC (DMIC_CTL[3:0] = 0) and clear VAD active flag and set VAD path switch by set register ACTCL (VAD_SINCCTL[30]) and SW (VAD_SINCCTL[29]) to 1.

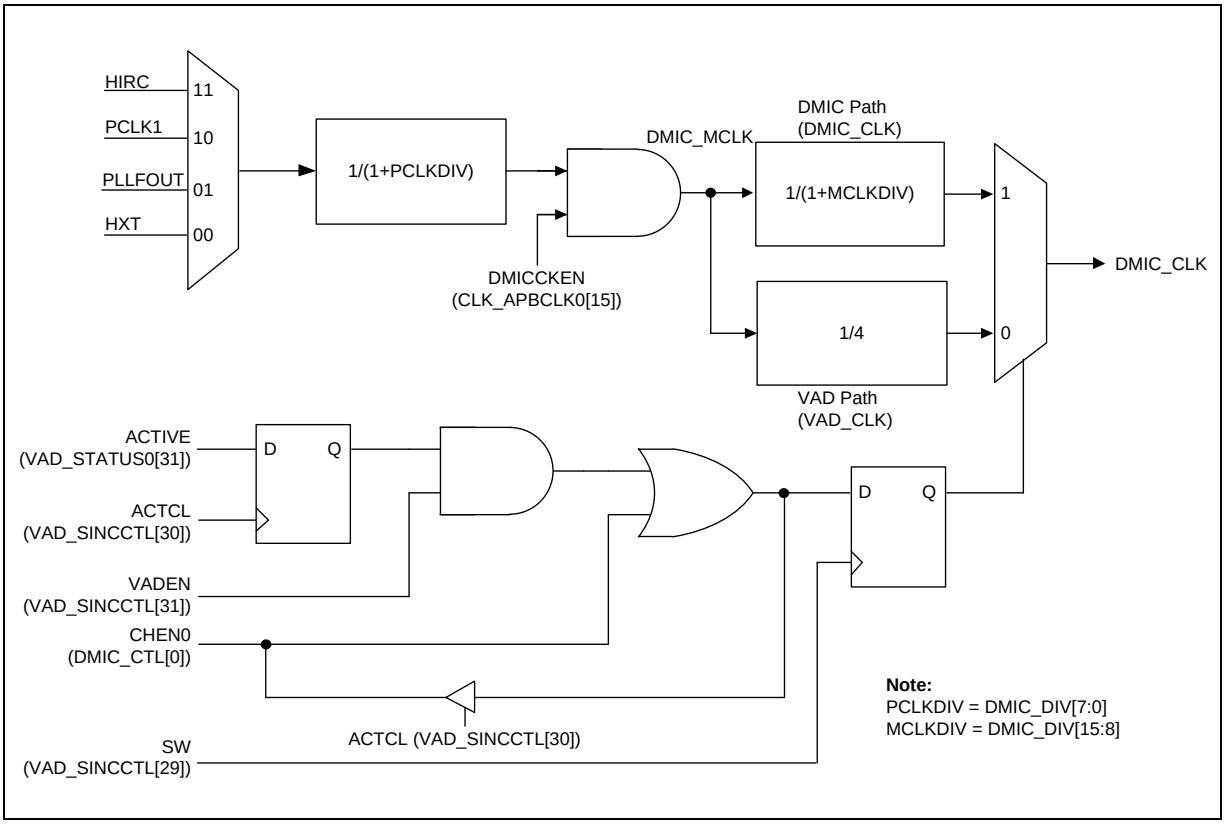


Figure 6.20-2 VAD Clock Control Diagram

6.20.5.2 VAD Data Control

The VAD data diagram is shown in Figure 6.20-3.

When VAD enabled (VADEN (VAD_SINCCTL[31]) = 1) and voice detected (ACTIVE (VAD_STATUS0[31]) = 1), the PCM data can be sent to SRAM via PDMA to do key word detection by software solution. And user can stop sending PCM data to SRAM by setting register DATAOFF (VAD_SINCCTL[28]).

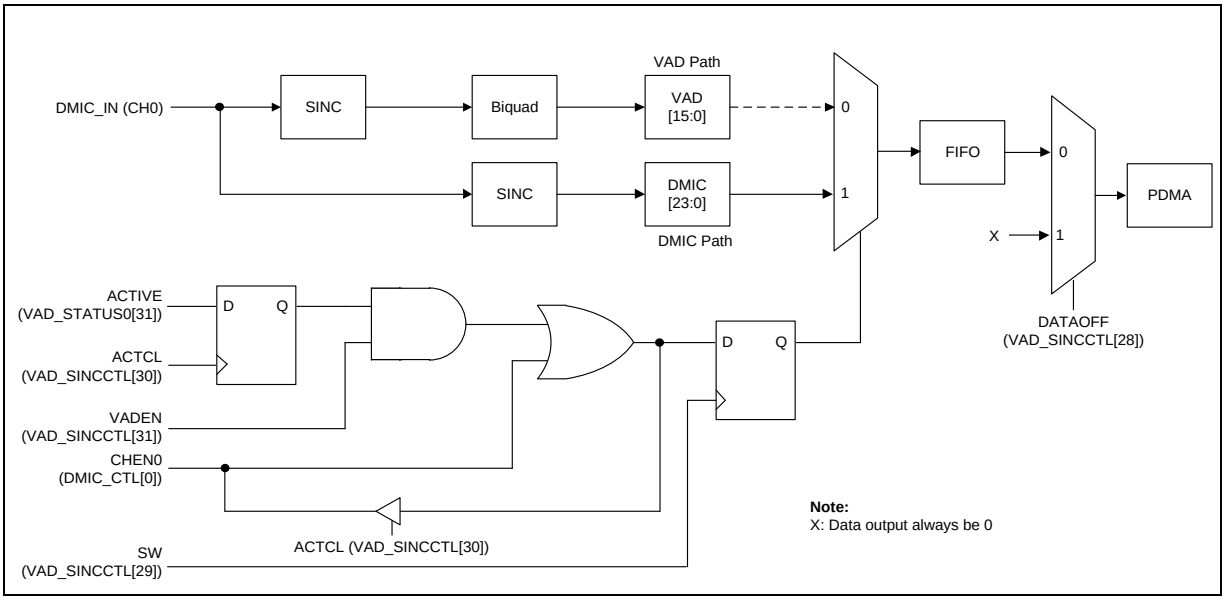


Figure 6.20-3 VAD Data Diagram

6.20.5.3 SINC Filter

For the SINC filter, it has three over sampling rate (OSR) configuration that controlled by register SINCOSR (VAD_SINC_CTL[11:8]): OSR48, OSR64, OSR96. For the three OSR options, the DMIC bus clock (DMIC_CLK) will be (Fs x 48) kHz, (Fs x 64) kHz or (Fs x 96) kHz, where Fs is sample rate. The DMIC_CLK will be controlled by VAD module when VADEN (VAD_SINCCTL[31]) is enabled. The frequency of DMIC_CLK is obtained according to the following equation.

F_DMIC_CLK = F_DMIC_MCLK / 4

where F_DMIC_CLK is the frequency of DMIC_CLK and F_DMIC_MCLK is the frequency of DMIC_MCLK.

The frequency of DMIC working main clock (DMIC_MCLK) should be F_DMIC_CLK x 4.

6.20.5.4 Biquad Filter

The biquad filter is a second-order recursive linear filter with two poles and two zeros. Its transfer function is the Z-domain consists of two quadratic functions:

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

Each Biquad Coefficient (a1, a2, b0, b1 and b2) has 16 bits in Sxx.13 format where

1. S is the sign bit (1 bit)
2. xx are integers (2bits)
3. 13 fractional bits (13 bits)

6.20.5.5 VAD Configuration

VAD analyses the PCM data from DMIC channel 0. In order to use the VAD function, the parameters need to be set correctly. First it's to set the attack time. For the attack time setting, the bigger we set, the faster the energy we calculated. So the default value of the LTAT (VAD_CTL0[19:16]) (Long term attack time) is 0x7, and the STAT (VAD_CLT0 [7:0]) (short term attack time) is 0xCC. If you want to calculate the energy faster, you can set it bigger, and then the waveform of the energy will be more similar with the input. But the STAT (VAD_CLT0[7:0]) should be always bigger than the LTAT (VAD_CTL0[19:16]).

Then we need to set the threshold of the energy, including the short term energy threshold, long term energy threshold and the deviation energy threshold. The threshold setting is based on the input level. The bigger the input level, the larger the threshold setting. If the threshold is set too high, then the VAD cannot detect the voice, and if the threshold is set too low, then it may have some wrong detection.

In order to use the VAD detection, user can have some example such as some actual voice file to tune the parameters.

6.20.5.6 VAD Decision Tree

Figure 6.20-4 illustrates the operation flow of VAD.

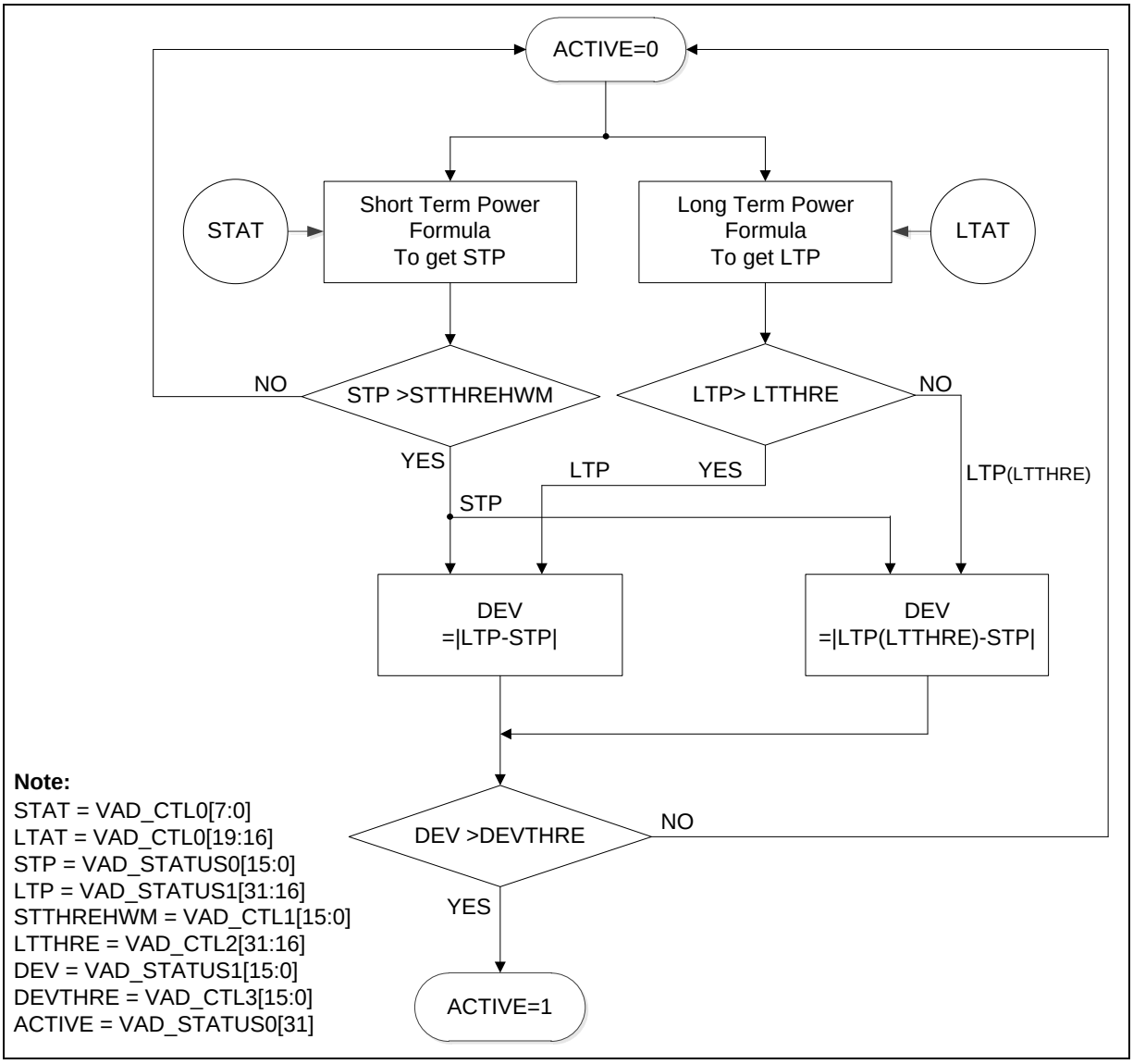


Figure 6.20-4 VAD Decision Tree

6.20.5.7 Tuning VAD Performance

This section provides a tuning guide for VAD performance. The VAD calculates power of an input signal to determine whether any voice is present or not. The calculations include a combination of long-term / short-term power and deviation power.

Step 1: Set Short Term and Long Term Power Attack Time

Short term power attack time should be properly set such that the short-term power reflects the human voices. Long term power attack time should be set such that the long-term power represents the ambient condition. In general, lower setting means slower attack or long responding time; whereas higher setting means faster attack or short responding time. Suggested default attack time setting:

Short term power attack time: STAT (VAD_CTL0[7:0]) = 0xAA. If slower attack is desired, e.g., if the VAD is triggered frequently by many non-voice sudden sounds, 0x99 may be used. If faster attack is desired, e.g., if the VAD appears to suffer long latency when clear voices are present, 0xCC or 0xBB may be used.

STAT (VAD_CTL0[7:0])	0x99	0xAA	0xBB	0xCC
Attack Time	16 ms	8 ms	4 ms	2 ms

Table 6.20.5-1 Short Term Power Attack Time Selection

Long term power attack time: LTAT (VAD_CTL0[19:16]) = 0x5. If slower attack is desired, e.g., if the target environment is relatively stable, 0x4 may be used. If faster attack is desired, e.g., if the target environment has non-stationary background noise, 0x6 or 0x7 may be used.

LTAT (VAD_CTL0[19:16])	0x4	0x5	0x6	0x7
Attack Time	512 ms	256 ms	128 ms	64 ms

Table 6.20.5-2 Long Term Power Attack Time Selection

Step 2: Ambient Noise Only Power Measurement

Read out STP (VAD_STATUS0[15:0]) for short term power calculation and DEV (VAD_STATUS1[15:0]) for deviation and LTP (VAD_STATUS1[31:16]) for long term power calculation.

Set short term power threshold STTHREHWM (VAD_CTL1[15:0]) higher than STP (VAD_STATUS0[15:0]) and deviation threshold DEVTHRE (VAD_CTL3[15:0]) higher than DEV (VAD_STATUS1[15:0]) and long term power threshold LTTHRE (VAD_CTL2[31:16]) higher than LTP (VAD_STATUS1[31:16]) in ambient condition, then checked ACTIVE (VAD_STATUS0[31]) should be 0.

At this point, VAD has been adjusted to eliminate false triggers from ambient noise. However, to minimize false triggers from occasional background noise (i.e. paper shuffling, footsteps, music, etc.), short term and deviation threshold need to increase the focus of the VAD sensitivity to human voice.

Step 3: Human Voice Power Measurement (constantly playing)

Read out STP (VAD_STATUS0[15:0]) for short term power calculation and DEV (VAD_STATUS1[15:0]) for deviation

Play the sound file, and tune the volume louder than the volume of normal people talking. This is to observe VAD distinction between ambient noise and human voice.

Set short term power threshold STTHREHWM (VAD_CTL1[15:0]) lower than STP (VAD_STATUS0[15:0]) and deviation threshold DEVTHRE (VAD_CTL3 [15:0]) lower than DEV (VAD_STATUS1[15:0]) in human voice condition, then checked ACTIVE (VAD_STATUS0[31]) should be 1.

Step 4: Set short term power threshold with a value right below short term signal power values obtained from step 3

Set short term power threshold STTHREHWM (VAD_CTL1[15:0]) lower than short term signal power STP (VAD_STATUS0[15:0]) read out values.

Step 5: Set Deviation Threshold with a value right below Deviation values obtained from

step 3

Set deviation threshold DEVTHRE (VAD_CTL3[15:0]) lower than deviation power DEV (VAD_STATUS1[15:0]) read out values.

Short term power threshold STTHREHWM (VAD_CTL1) and deviation threshold DEVTHRE (VAD_CTL3[15:0]) can be fine-tuned to achieve the sensitivity to desired human speaking volume.

To reduce sensitivity to background noise, these actions are recommended to perform in certain combinations only: increase speech trigger alone, or increase speech trigger and speech window together to reduce sensitivity to background noise. Increasing speech window alone will not help reduce sensitivity to background noise.

The Reference Table of power threshold as Table 6.20.5-3.

Power Threshold	0x7FFF	0x2879	0x0CCC	0x040C	0x0147
dB	0 dB	10 dB	20 dB	30 dB	40 dB
Power Threshold	0x0067	0x0020	0x000A	0x0003	0x0001
dB	50 dB	60 dB	70 dB	80 dB	90 dB

Table 6.20.5-3 Power Threshold Reference

6.20.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
VAD Base Address: **VAD_BA = 0x4006_3100**				
VAD_SINCCTL	VAD_BA+0x00	R/W	VAD SINC Filter Control Register	0x0000_0008
VAD_BIQCTL0	VAD_BA+0x04	R/W	VAD Biquad Filter Control Register 0	0x0000_0000
VAD_BIQCTL1	VAD_BA+0x08	R/W	VAD Biquad Filter Control Register 1	0x0000_0000
VAD_BIQCTL2	VAD_BA+0x0C	R/W	VAD Biquad Filter Control Register 2	0x0000_0000
VAD_CTL0	VAD_BA+0x10	R/W	VAD Control Register 0	0x0007_00CC
VAD_CTL1	VAD_BA+0x14	R/W	VAD Control Register 1	0x0000_7FFF
VAD_CTL2	VAD_BA+0x18	R/W	VAD Control Register 2	0x0000_0000
VAD_CTL3	VAD_BA+0x1C	R/W	VAD Control Register 3	0x0000_7FFF
VAD_STATUS0	VAD_BA+0x20	R	VAD Status Read-Back Register 0	0x0000_0000
VAD_STATUS1	VAD_BA+0x24	R	VAD Status Read-Back Register 1	0x0000_0000

Note:

1. Any register not listed here is reserved and must not be written. The result of a read operation on these bits is undefined.
2. The reserved register fields that listed in register description must be written to their reset value. Writing reserved fields with other than reset values may produce undefined results.

6.20.7 Register Description

VAD SINC Filter Control Register (VAD_SINCCTL)

Register	Offset	R/W	Description	Reset Value
VAD_SINCCTL	VAD_BA+0x00	R/W	VAD SINC Filter Control Register	0x0000_0008

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

Bits	Description	
[31]	**VADEN**	**VAD Enable Control** 0 = VAD Disabled. 1 = VAD Enabled. **Note 1:** When set this bit to 1, CHEN0 (DMIC_CTL[0]) will be set to 1 and CHEN1 (DMIC_CTL[1]), CHEN2 (DMIC_CTL[2]) and CHEN3 (DMIC_CTL[3]) will be set to 0 automatically. **Note 2:** When set this bit to 1, DMIC_CLK is generated by VAD module.
[30]	**ACTCL**	**VAD Active Flag Clear** 0 = No effect. 1 = Clear ACTIVE(VAD_STATUS0[31]). **Note:** After ACTIVE(VAD_STATUS0[31]) is cleared, user need to set set this bit to 0.
[29]	**SW**	**VAD Path Switch Control** After the ACTIVE(VAD_STATUS0[31]) goes high, it will automatically switch to the DMIC path. When the CPU is entering idle mode, write 1 to switch back to the VAD path. **Note 1:** After switch back VAD path, user need to set this bit to 0. **Note 2:** User need to set DMIC_CTL[3:0] to 1 and clear ACTIVE (VAD_STATUS0[31]) before set this bit 1.
[28]	**DATAOFF**	**VAD Sending Data to SRAM Control** When the ACTIVE (VAD_STATUS0[31]) goes high, the data will be transferred to SRAM to store which can be used for keyword detection later. After some time, if user needs to stop sending data to SRAM, write this bit to 1.
[27:12]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[11:8]	**SINCOSR**	**VAD SINC Filter OSR Setting** 000 = Down sample 48 001 = Down sample 64

		010 = Down sample 96 Others = Reserved. Do not use.
[7:0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

VAD Biquad Filter Control Register 0 (VAD_BIQCTL0)

Register	Offset	R/W	Description	Reset Value
VAD_BIQCTL0	VAD_BA+0x04	R/W	VAD Biquad Filter Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
BIQA2							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
BIQA2							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
BIQA1							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BIQA1							

Bits	Description	
[31:16]	**BIQA2**	**VAD Biquad Filter Coefficient** Biquad Filter Coefficient a2, in 3 intergers + 13 fractional bits.
[15:0]	**BIQA1**	**VAD Biquad Filter Coefficient** Biquad Filter Coefficient a1, in 3 intergers + 13 fractional bits

VAD Biquad Filter Control Register 1 (VAD_BIQCTL1)

Register	Offset	R/W	Description	Reset Value
VAD_BIQCTL1	VAD_BA+0x08	R/W	VAD Biquad Filter Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
BIQB1							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
BIQB1							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
BIQB0							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BIQB0							

Bits	Description	
[31:16]	**BIQB1**	**VAD Biquad Filter Coefficient** Biquad Filter Coefficient b1, in 3 intergers + 13 fractional bits.
[15:0]	**BIQB0**	**VAD Biquad Filter Coefficient** Biquad Filter Coefficient b0, in 3 intergers + 13 fractional bits.

VAD Biquad Filter Control Register 2 (VAD_BIQCTL2)

Register	Offset	R/W	Description	Reset Value
VAD_BIQCTL2	VAD_BA+0x0C	R/W	VAD Biquad Filter Control Register 2	0x0000_0000

31	30	29	28	27	26	25	24
BIQEN	**Reserved**						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
BIQB2							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BIQB2							

Bits	Description	
[31]	**BIQEN**	**VAD Biquad Filter Enable Bit** 0 = VAD Biquad Filter Disabled. 1 = VAD Biquad Filter Enabled.
[30:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**BIQB2**	**VAD Biquad Filter Coefficient** Biquad Filter Coefficient b2, in 3 intergers + 13 fractional bits.

VAD Control Register 0 (VAD_CTL0)

Register	Offset	R/W	Description	Reset Value
VAD_CTL0	VAD_BA+0x10	R/W	VAD Control Register 0	0x0007_00CC

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

Bits	Description	
[31:20]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[19:16]	**LTAT**	**Long Term Power Attack Time** Slow attack (e.g., 0x5): less sensitive to environment change. Fast attack (e.g., 0x8): more sensitive to environment change.
[15:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7:0]	**STAT**	**Short Term Power Attack Time** Slow attack (e.g., 0x99): slow responding to voice, but more stable. Fast attack (e.g., 0xCC): fast responding to voice, but more sensitive to other sounds. Suggested default attack time setting: Long term power attack time (0x5), Short term power attack time (0xAA). The “Short Term Power”, in order to detect the instant power of the voices, requires faster attack time, while “Long Term Power”, in order to get the averaged power of the background environment, requires slower attack time to maintain its stability. So the Short term power attack time should be always bigger than the Long term power attack time.

VAD Control Register 1 (VAD_CTL1)

Register	Offset	R/W	Description	Reset Value
VAD_CTL1	VAD_BA+0x14	R/W	VAD Control Register 1	0x0000_7FFF

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

Bits	Description	
[31:16]	**STTHRELWM**	**Short Term Power Threshold Lower Limit** To check if the incoming signal is small enough so that VAD status can be terminated.
[15:0]	**STTHREHWM**	**Short Term Power Threshold Upper Limit** To check if the incoming signal is big enough to be ready for VAD activation.

VAD Control Register 2 (VAD_CTL2)

Register	Offset	R/W	Description	Reset Value
VAD_CTL2	VAD_BA+0x18	R/W	VAD Control Register 2	0x0000_0000

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

Bits	Description	
[31:16]	**LTTHRE**	**Long Term Power Threshold** To check the background energy, also serve as the lower limit of long term power. When the long term power value is lower than the threshold, it will be set to the threshold value for VAD decision.
[15:0]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.

VAD Control Register 3 (VAD_CTL3)

Register	Offset	R/W	Description	Reset Value
VAD_CTL3	VAD_BA+0x1C	R/W	VAD Control Register 3	0x0000_7FFF

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

Bits	Description	
[31:16]	**HOT**	**Hang Over time** Hang Over time setting, means how many clocks (CLKSD) of the ACTIVE (VAD_STATUS0[31]) staying high when the calculation is no longer bigger than the threshold
[15:0]	**DEVTHRE**	**Deviation Threshold** To check if the incoming signal is substantially bigger than its background. This may work to exclude breath sound as it is slowly varying, but not other sounds (e.g., footsteps, hand claps) with sudden amplitude increase. Small: easy to trigger, good for far-field pick-up, but requiring quiet environment. Large: good for handheld applications, but requiring louder voice to trigger.

VAD Status Register 0 (VAD_STATUS0)

Register	Offset	R/W	Description	Reset Value
VAD_STATUS0	VAD_BA+0x20	R	VAD Status Read-Back Register 0	0x0000_0000

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

Bits	Description	
[31]	**ACTIVE**	**VAD Activation Flag (Read Only)** When the voice active event occurs, this bit will be set to 1. 0 = No effect. 1 = Voice detected. **Note:** When wake-up from idle mode, user need to set CHENn DMIC_CTL[3:0] for DMIC path normal operation.
[30:16]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[15:0]	**STP**	**Short Term Signal Power (Read Only)** This field shows the short term signal power value.

VAD Status Register 1 (VAD_STATUS1)

Register	Offset	R/W	Description	Reset Value
VAD_STATUS1	VAD_BA+0x24	R	VAD Status Read-Back Register 1	0x0000_0000

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

Bits	Description	
[31:16]	**LTP**	**Long Term Signal Power (Read Only)** This field shows the long term signal power value.
[15:0]	**DEV**	**Deviation (Read Only)** This field shows deviation of the Long Term Signal Power and Short Term Signal Power.

6.21 Audio DPWM Modulator (DPWM)

6.21.1 Overview

The DPWM modulator is sigma-delta modulator which is for class D amplifer. NPCA121 series has 3 DPWM modulator and each one can provide 2 differential pins.

6.21.2 Features

- Differential Audio PWM Output (DPWM)
- Support left channel,right channel and sub-woofer channel.
- Support sample rates from 16~96kHz.
- Programeable biquad filter with 10 band
- PDMA data channel for streaming of PCM audio data.
- Support the single precision floating point for input data and BIQ coefficient
- Provides one 32-level FIFO data buffers for transmitting.

6.21.3 Block Diagram

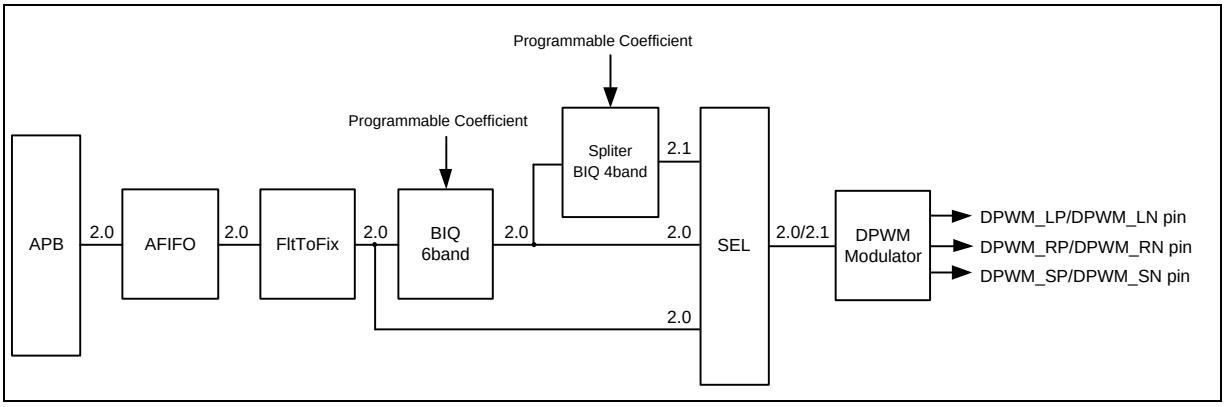


Figure 6.21-1 DPWM Block Diagram

6.21.4 Basic Configuration

- Clock source configuration
  - The source of DPWM peripheral clock is selected by DPWMSEL (CLK_CLKSEL2[13:12]).
  - The DPWM peripheral clock is enabled by DPWMCKEN (CLK_APBCLK1[6]).
- Reset configuration
  - The DPWM module is reset by DPWMRST (SYS_IPRST2[6]).
- Pin configuration

Group	Pin Name	GPIO	MFP
DPWM	DPWM_LN	PA.4	MFP3
		PC.12	MFP3
		PD.0	MFP5

	DPWM_LP	PA.5	MFP3
		PC.13	MFP3
		PD.1	MFP5
	DPWM_RN	PA.10	MFP3
		PC.10	MFP3
		PD.5	MFP5
	DPWM_RP	PA.11	MFP3
		PC.11	MFP3
		PD.6	MFP5
	DPWM_SN	PA.13	MFP3
		PC.14	MFP3
		PD.8	MFP5
	DPWM_SP	PA.14	MFP3
		PC.15	MFP3
		PD.9	MFP5

6.21.5 Functional Description

The DPWM block receives audio data by writing PCM audio to the FIFO. FIFO is accessed through PDMA for ease of streaming. The audio stream is sampled by a zero-order hold and fed to an upsmaple filter. The signal is then modulated and sent to the driver stage through a non-overlap circuit. Master clock rate of the Delta-Sigma modulator is controlled by DPWM_CLK. This clock is generated by the internal system clock.

6.21.5.1 DPWM Clock Generation

The DPWM module has two clock sources selected by register DPWMSEL (CLK_CLKSEL2[13:12]). The DPWM clock control diagram is shown in Figure 6.21-2. Note that the frequency of DPWM_CLK must be 512 fs (Sample rate) or 500 fs according to the value of register CLKSET (DPWM_CTL[31]).

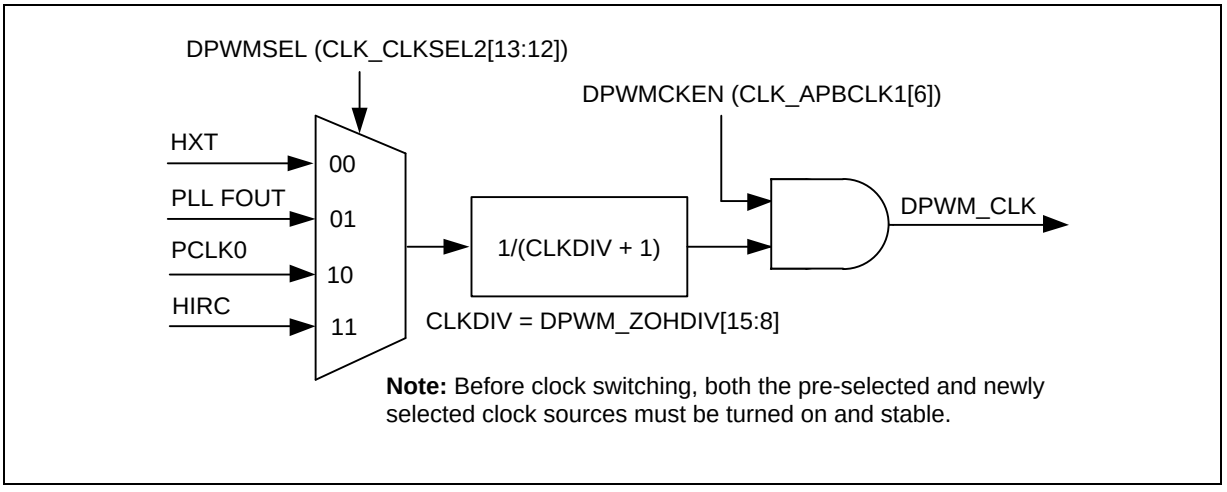


Figure 6.21-2 DPWM Clock Control Diagram

6.21.5.2 Determining Sample Rate

The sample rate at which the DPWM block consumes audio data is given by:

Fs = F_DPWM_CLK / (ZOHDIV * K)

K = 125, if CLKSET(DPWM_CTL[31]) is 1.

K = 128, if CLKSET(DPWM_CTL[31]) is 0.

Where F_DPWM_CLK is the frequency of DPWM_CLK.

6.21.5.3 BIQ Coefficient Operation and Generation

A coefficient programmable 10 band Biquad filter (20th-Order IIR filter) is available.

Each band biquad filter has the transfer function as H(z) and is implemented in Direct Form II Transpose structure as.

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

The biquad filter in each band has 5 user programmable coefficients (b0, b1 , b2, a1 and a2) and each band occupy 5 consecutive registers. The coefficient supports the single floating point and fixed point. If it is fixed point CPU write DPWM_FIFO[23:0] 24 bits format 3.21. If it is the single floating poing CPU write DPWM_FIFO[31:0] 32 bits. Set COEFFFLTEN (DPWM_COEFFCTL[1]) to 1, internal automatically transfer it to fixed point.

Fixed point Coefficient transferation from decimal to Hex:

[b0,b1,b2,a1,a2] * 2^21 --→ 24 bits hex

Fiflter mathmetical equation:

Parameter:

Fs = sample rate

Fc = center Splitter

Q = qualty factor

TK = tan(pi*Fc/Fs);

norm = 1/(1+TK/Q+TK^2)

G = 10^(peakgain/20)

A. Second order low pass filter coefficient

b0 = TK^2 *norm

b1 = 2 *b0

b2 = b0

a1 = 2*(TK^2-1)*norm

a2 = (1-TK/Q +TK^2) * norm

B. Second order high pass filter coefficient

b0 = norm

b1 = -2 *b0

b2 = b0

a1 = 2*(TK^2-1)*norm

a2 = (1-TK/Q +TK^2) * norm

C. Second order bandpass filter coefficient

b0 = TK/Q *norm

b1 = 0

b2 = -b0

a1 = 2*(TK^2-1)*norm

a2 = (1-TK/Q +TK^2) * norm

D. Second order notch filter coefficient

b0 = (1+TK^2) *norm

b1 = 2 *(TK^2-1) *norm

b2 = b0

a1 = b1

a2 = (1-TK/Q +TK^2) * norm

E. Second order peaking boost EQ coefficient

b0 = (1+G/Q*TK+K^2) *norm

b1 = 2 *(TK^2-1) *norm

b2 = (1-G/Q*TK +TK2)*norm

a1 = b1

a2 = (1-TK/Q +TK^2) * norm

F. Second order peaking cut EQ coefficient

b0 = (1+1/Q*TK+TK^2) *norm

b1 = 2 *(TK^2-1) *norm

b2 = (1-1/Q*TK +TK2)*norm

a1 = b1

a2 = (1-G/Q*TK +TK^2) * norm

G. Fourth order splitter coefficient

Q = 0.707

b0_1 = TK^2 *norm

b1_1 = 2 *b0

b2_1 = b0

a1_1 = 2*(TK^2-1)*norm

a2_1 = (1-TK/Q T+K^2) * norm

b0_2=T K^2 *norm

b1_2 = 2 *b0

b2_2 = b0

a1_2 = 2*(TK^2-1)*norm

a2_2 = (1-TK/Q +TK^2) * norm

b0_3 = norm

b1_3 = -2 *b0

b2_3 = b0

a1_3 = 2*(TK^2-1)*norm

a2_3 = (1-TK/Q +TK^2) * norm

b0_4 = norm

b1_4 = -2 *b0

b2_4 = b0

a1_4 = 2*(TK^2-1)*norm

a2_4 = (1-TK/Q +K^2) * norm

6.21.5.4 Splitter Configuring

The splitter is cross-over with LR4 to generate left, right and sub-woofer channel shared biquad filter 4 bands. Note that when configure coefficients, the biquad filter (BIQON (DPWM_CTL[21]) = 0) and splitter (SPLTON (DPWM_CTL[22]) = 0) must be turned off. Splitter uses 4 bands of coefficient.

To configure the splitter:

- Reset audio DPWM modulator by setting register DPWMRST (SYS_IPRST2[6]).
- Enable coefficient RAM programming mode by setting register PRGCOEFF (DPWM_COEFFCTL[0]) to “1”.
- Set 4 bands of splitter coefficient in register COEFFDAT (DPWM_COEFFn, n = 0~19). The coefficient uses band1 to band4.
- Disable coefficient RAM programming mode by setting register PRGCOEFF (DPWM_COEFFCTL[0]) to “0”.
- Set biquad filter band number to “4” in register BIQBANDNUM (DPWM_CTL[27:24]).
- Enable splitter by setting register SPLTON (DPWM_CTL[22]) to “1”.

The following is splitter frequency response and channel distribution. The splitter does frequency separation for left and right channels. Left and right channels keep high frequency signals and the low frequency signals go to sub-woofer.

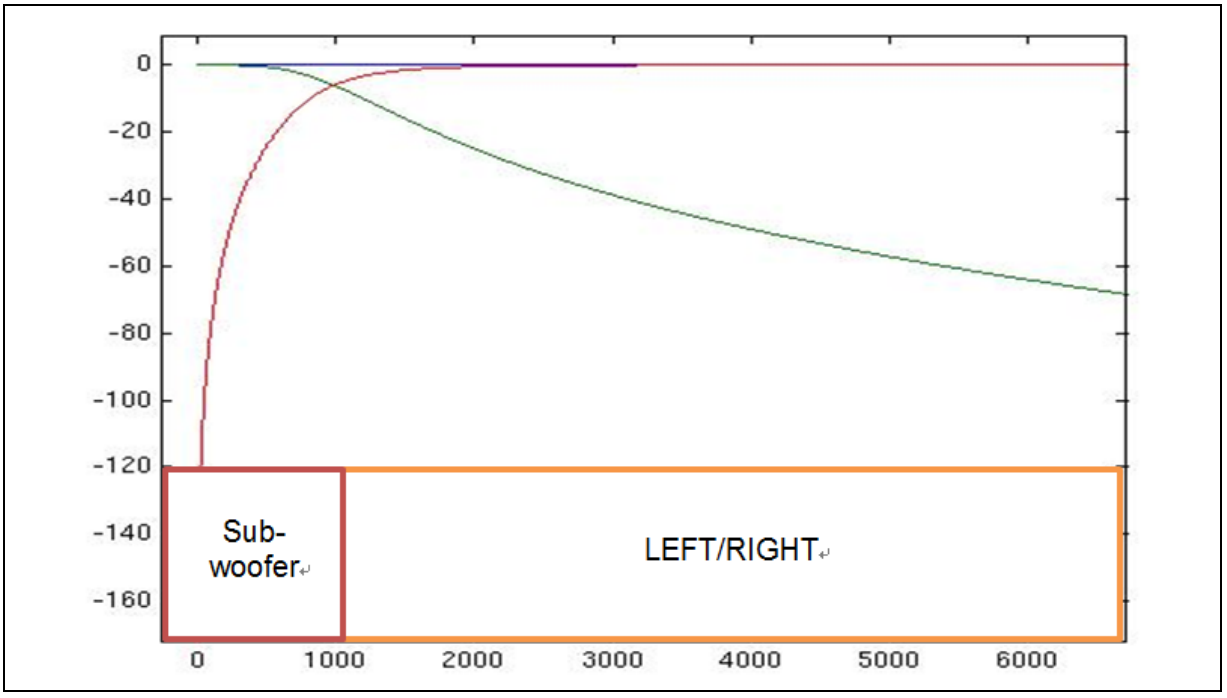


Figure 6.21-3 Splitter Frequency Response and Channel Distribution

6.21.5.5 Biquad Filter Configuring

A coefficient programmable 10 bands biquad filter (20th-Order IIR filter) is available. Note that if DPWM_CLK is 500fs and Fs is 48 kHz, bquad filter supports 9 bands.

To configure the biquad filter:

- Reset audio DPWM modulator by setting register DPWMRST (SYS_IPRST2[6]).
- Enable coefficient RAM programming mode by setting register PRGCOEFF (DPWM_COEFFCTL[0]) to “1”.
- Set biquad filter coefficient in register COEFFDAT (DPWM_COEFFn, if BIQBANDNUM = 6, n = 0 ~ (6 x 5 – 1)).
- Disable coefficient RAM programming mode by setting register PRGCOEFF (DPWM_COEFFCTL[0]) to “0”.
- Set biquad filter band number in register BIQBANDNUM (DPWM_CTL[27:24]).
- Enable biquad filter by setting register BIQON (DPWM_CTL[21]) to “1”.

6.21.5.6 Biquad filter and splitter Configuring

The splitter shares 4 bands biquad filter. The maximum number of biquad filter bands become 6, if biquad filter and splitter both are on. The bands of biquad filter are ahead then set splitter band when configure coefficient register. For example, if the total number of bands is 6, the biquad filter coefficients need to set in registers DPWM_COEFF0 to DPWM_COEFF9, and splitter coefficients need to set in registers DPWM_COEFF10 to DPWM_COEFF29.

To configure the biquad filter and splitter:

- Reset audio DPWM modulator by setting register DPWMRST (SYS_IPRST2[6]).
- Enable coefficient RAM programming mode by setting register PRGCOEFF (DPWM_COEFFCTL[0]) to “1”.

- Set coefficients in coefficient registers, biquad coefficient is first, last 4 band coefficients are for splitter.
- Disable coefficient RAM programming mode by setting register PRGCOEFF (DPWM_COEFFCTL[0]) to “0”.
- Set band number (biquad filter band number + 4 for Splitter) in register BIQBANDNUM (DPWM_CTL[27:24]).
- Enable biquad filter by setting register BIQON (DPWM_CTL[21]) to “1” and enable splitter by setting register SPLTON (DPWM_CTL[22]) to “1”.

6.21.5.7 FIFO Data Operation

FIFO bits is 32 bits. It supports the single floating point and fixed point, channel position is as below. The bit-width of audio data in a channel block can be 8, 16 or 24bits. The memory arrangements of audio data for various settings are shown in Figure 6.21-4.

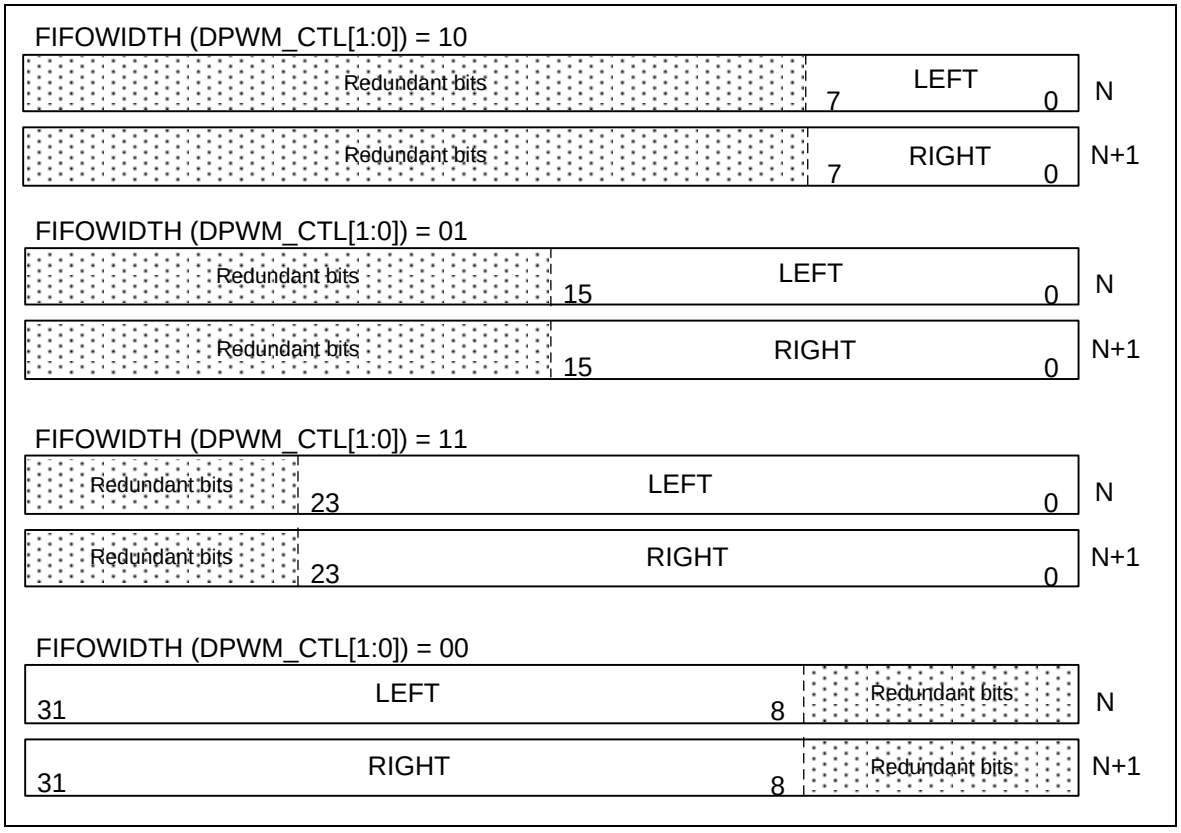


Figure 6.21-4 Audio DPWM FIFO Contents for Various Data Width

If it is fixed point set register FLTEN (DPWM_CTL[20]) to 0 and set FIFOWIDTH (DPWM_CTL[1:0]) for format. If it is the single floating point set FLTEN (DPWM_CTL[20]) to 1 and internal transfer it to fixed point. Set FLTINBIT (DPWM_CTL[19:17]) for integer format.

6.21.5.8 Peripheral DMA Request

Normal use of the audio DPWM is with PDMA. In this mode DPWM requests PDMA service whenever there is space in FIFO. PDMA channel will copy data from a streaming buffer to the DPWM FIFO and alert the CPU when buffer is empty. In this way an entire buffer of data can be sent to DPWM without any CPU intervention.

6.21.5.9 Configuring DPWM Modulator

To operate the DPWM modulator the following configuration is recommended:

- Enable DPWM clock source by register DPWMCKEN (CLK_APBCLK1[6]).
- Reset DPWM IP block by register DPWMRST (SYS_IPRST2[6]).
- Select sample rate based on current frequency of DPWM module.
- Setup PDMA channel to provide data to DPWM.
- Enable PDMA Request by using register PDMAEN (DPWM_PDMACTL[0]).
- Enable DPWM Modulator by using register DPWMEN (DPWM_CTL[6]).

6.21.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
DPWM Base Address: **DPWM_BA = 0x4006_4000**				
DPWM_CTL	DPWM_BA+0x00	R/W	DPWM Control Register	0x0000_0600
DPWM_STATUS	DPWM_BA+0x04	R	DPWM Status Register	0x0000_0002
DPWM_PDMACTL	DPWM_BA+0x08	R/W	DPWM PDMA Control Register	0x0000_0000
DPWM_FIFO	DPWM_BA+0x0C	W	DPWM FIFO Data Input Register	0x0000_0000
DPWM_ZOHDIV	DPWM_BA+0x10	R/W	DPWM Zero Order Hold Division Register	0x0000_0804
DPWM_FREQ	DPWM_BA+0x14	R/W	DPWM Output Signal Frequency Control Register	0x0000_0000
DPWM_COEFFCTL	DPWM_BA+0xFC	R/W	BIQ Coefficient Control	0x0000_0000
DPWM_COEFF0	DPWM_BA+0x100	R/W	Coefficient b0 Transfer function for band 1 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF1	DPWM_BA+0x104	R/W	Coefficient b1 Transfer function for band 1 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF2	DPWM_BA+0x108	R/W	Coefficient b2 Transfer function for band 1 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF3	DPWM_BA+0x10C	R/W	Coefficient a1 Transfer function for band 1 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF4	DPWM_BA+0x110	R/W	Coefficient a2 Transfer function for band 1 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF5	DPWM_BA+0x114	R/W	Coefficient b0 Transfer function for band 2 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF6	DPWM_BA+0x118	R/W	Coefficient b1 Transfer function for band 2 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF7	DPWM_BA+0x11C	R/W	Coefficient b2 Transfer function for band 2 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF8	DPWM_BA+0x120	R/W	Coefficient a1 Transfer function for band 2 fixed point – 3.21 format floating point – single precision point	0x0000_0000

Register	Offset	R/W	Description	Reset Value
DPWM Base Address: **DPWM_BA = 0x4006_4000**				
DPWM_COEFF9	DPWM_BA+0x124	R/W	Coefficient a2 Transfer function for band 2 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF10	DPWM_BA+0x128	R/W	Coefficient b0 Transfer function for band 3 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF11	DPWM_BA+0x12C	R/W	Coefficient b1 Transfer function for band 3 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF12	DPWM_BA+0x130	R/W	Coefficient b2 Transfer function for band 3 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF13	DPWM_BA+0x134	R/W	Coefficient a1 Transfer function for band 3 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF14	DPWM_BA+0x138	R/W	Coefficient a2 Transfer function for band 3 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF15	DPWM_BA+0x13C	R/W	Coefficient b0 Transfer function for band 4 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF16	DPWM_BA+0x140	R/W	Coefficient b1 Transfer function for band 4 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF17	DPWM_BA+0x144	R/W	Coefficient b2 Transfer function for band 4 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF18	DPWM_BA+0x148	R/W	Coefficient a1 Transfer function for band 4 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF19	DPWM_BA+0x14C	R/W	Coefficient a2 Transfer function for band 4 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF20	DPWM_BA+0x150	R/W	Coefficient b0 Transfer function for band 5 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF21	DPWM_BA+0x154	R/W	Coefficient b1 Transfer function for band 5 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF22	DPWM_BA+0x158	R/W	Coefficient b2 Transfer function for band 5	0x0000_0000

Register	Offset	R/W	Description	Reset Value
DPWM Base Address: **DPWM_BA = 0x4006_4000**				
			fixed point – 3.21 format floating point – single precision point	
DPWM_COEFF23	DPWM_BA+0x15C	R/W	Coefficient a1 Transfer function for band 5 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF24	DPWM_BA+0x160	R/W	Coefficient a2 Transfer function for band 5 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF25	DPWM_BA+0x164	R/W	Coefficient b0 Transfer function for band 6 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF26	DPWM_BA+0x168	R/W	Coefficient b1 Transfer function for band 6 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF27	DPWM_BA+0x16C	R/W	Coefficient b2 Transfer function for band 6 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF28	DPWM_BA+0x170	R/W	Coefficient a1 Transfer function for band 6 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF29	DPWM_BA+0x174	R/W	Coefficient a2 Transfer function for band 6 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF30	DPWM_BA+0x178	R/W	Coefficient b0 Transfer function for band 7 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF31	DPWM_BA+0x17C	R/W	Coefficient b1 Transfer function for band 7 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF32	DPWM_BA+0x180	R/W	Coefficient b2 Transfer function for band 7 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF33	DPWM_BA+0x184	R/W	Coefficient a1 Transfer function for band 7 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF34	DPWM_BA+0x188	R/W	Coefficient a2 Transfer function for band 7 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF35	DPWM_BA+0x18C	R/W	Coefficient b0 Transfer function for band 8 fixed point – 3.21 format	0x0000_0000

Register	Offset	R/W	Description	Reset Value
DPWM Base Address: **DPWM_BA = 0x4006_4000**				
			floating point – single precision point	
DPWM_COEFF36	DPWM_BA+0x190	R/W	Coefficient b1 Transfer function for band 8 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF37	DPWM_BA+0x194	R/W	Coefficient b2 Transfer function for band 8 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF38	DPWM_BA+0x198	R/W	Coefficient a1 Transfer function for band 8 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF39	DPWM_BA+0x19C	R/W	Coefficient a2 Transfer function for band 8 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF40	DPWM_BA+0x1A0	R/W	Coefficient b0 Transfer function for band 9 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF41	DPWM_BA+0x1A4	R/W	Coefficient b1 Transfer function for band 9 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF42	DPWM_BA+0x1A8	R/W	Coefficient b2 Transfer function for band 9 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF43	DPWM_BA+0x1AC	R/W	Coefficient a1 Transfer function for band 9 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF44	DPWM_BA+0x1B0	R/W	Coefficient a2 Transfer function for band 9 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF45	DPWM_BA+0x1B4	R/W	Coefficient b0 Transfer function for band 10 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF46	DPWM_BA+0x1B8	R/W	Coefficient b1 Transfer function for band 10 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF47	DPWM_BA+0x1BC	R/W	Coefficient b2 Transfer function for band 10 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF48	DPWM_BA+0x1C0	R/W	Coefficient a1 Transfer function for band 10 fixed point – 3.21 format floating point – single precision point	0x0000_0000

Register	Offset	R/W	Description	Reset Value
DPWM Base Address: **DPWM_BA = 0x4006_4000**				
DPWM_COEFF49	DPWM_BA+0x1C4	R/W	Coefficient a2 Transfer function for band 10 fixed point – 3.21 format floating point – single precision point	0x0000_0000

Note:

1. Any register not listed here is reserved and must not be written. The result of a read operation on these bits is undefined.
2. The reserved register fields that listed in register description must be written to their reset value. Writing reserved fields with other than reset values may produce undefined results.

6.21.7 Register Description

DPWM Control Register (DPWM_CTL)

Register	Offset	R/W	Description	Reset Value
DPWM_CTL	DPWM_BA+0x00	R/W	DPWM Control Register	0x0000_0600

31	30	29	28	27	26	25	24
CLKSET	**Reserved**	**FCLR**		**BIQBANDNUM**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	**SPLTON**	**BIQON**	**FLTEN**	**FLTINTBIT**			**TH**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TH				**THIE**	**Reserved**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DRVEN	**DPWMEN**	**Reserved**		**DEADTIME**	**Reserved**	**FIFOWIDTH**	

Bits	Description	
[31]	**CLKSET**	**Working Clock Selection** 0 = 512 fs working clock 1 = 500 fs working clock **Note:** For example, if the user want to get 48 kHz sample rate (fs), the frequency of DPWM_CLK need to be 24576 kHz when CLKSET = 0, the frequency of DPWM_CLK need to be 24000 kHz when CLKSET = 1.
[30]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[29:28]	**FCLR**	**FIFO Clear** 11 = Clear the FIFO. Others = Reserved. Do not use. **Note 1:** To clear the FIFO, need to write FCLR (DPWM_CTL[29:28]) to 11b, and can read the EMPTY (DPWM_STATUS[1]) bit to make sure that the FIFO has been cleared. **Note 2:** This field is auto cleared by hardware.
[27:24]	**BIQBANDNUM**	**BIQ Band Number Setting (Total 10 Bands)** This field represents the required number of bands. The minimum number is 1 and can up to 10 when user enables biquad filter or splitter.
[23]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[22]	**SPLTON**	**Splitter Enable Bit** 0 = 4-band splitter Disabled. 1 = 4-band splitter Enabled. **Note:** Splitter shared biquad filter 4 bands, the minimum number of BIQBANDNUM is 4, if splitter is enabled.
[21]	**BIQON**	**BIQ Enable Bit** 0 = Biquad filter Disabled. 1 = Biquad filter Enabled.

[20]	**FLTEN**	**Floating Point Format Enable Bit** 0 = Input data is fixed point. 1 = Input data is single precision point.
[19:17]	**FLTINTBIT**	**Floating Integer Bits Setting** 000 = Integer is 0, Data range +/- 0.999. 001 = Integer is 1, Data range +/- 1.9999. 010 = Integer is 2, Data range +/- 3.9999. Others = Reserved. Do not use.
[16:12]	**TH**	**FIFO Threshold Level** If the valid data count of the FIFO data buffer is less than or equal to TH (DPWM_CTL[16:12]) setting, the THIF (DPWM_STATUS[2]) will set to 1, else the THIF (DPWM_STATUS[2]) will be cleared to 0.
[11]	**THIE**	**FIFO Threshold Interrupt** 0 = FIFO threshold interrupt Disabled 1 = FIFO threshold interrupt Enabled.
[10:8]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[7]	**DRVEN**	**Driver Enable Bit** 0 = Audio DPWM driver Disabled. 1 = Audio DPWM driver Enabled.
[6]	**DPWMEN**	**Audio DPWM Modulator Enable** 0 = Audio DPWM modulator Disabled. 1 = Audio DPWM modulator Enabled.
[5:4]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[3]	**DEADTIME**	**Driver Dead Time Control.** Enabling this bit will insert an additional clock cycle deadtime into the switching of PMOS and NMOS driver transistors.
[2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1:0]	**FIFOWIDTH**	**FIFO Data Width** This bit field is used to define the bit-width of data word and valid bits in register DPWM_FIFO. 00 = The bit-width of data word is 24-bit, valid bits is DPWM_FIFO[31:8]. 01 = The bit-width of data word is 16-bit, valid bits is DPWM_FIFO[15:0]. 10 = The bit-width of data word is 8-bit, valid bits is DPWM_FIFO[7:0]. 11 = The bit-width of data word is 24-bit, valid bits is DPWM_FIFO[23:0]. **Note:** When FLTEN is “0”, FIFOWIDTH is for fixed point setting.

DPWM Status Register (DPWM_STATUS)

Register	Offset	R/W	Description	Reset Value
DPWM_STATUS	DPWM_BA+0x04	R	DPWM Status Register	0x0000_0002

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

Bits	Description	
[31:9]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[8:4]	**FIFOPTR**	**FIFO Pointer (Read Only)** The FULL (DPWM_STATUS[0]) and FIFOPTR (DPWM_STATUS[8:4]) indicates the field that the valid data count within the DPWM FIFO buffer. The maximum value shown in FIFOPTR is 31. When the using level of DPWM FIFO buffer equal to 32, The FULL (DPWM_STATUS[0]) is set to 1.
[3]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[2]	**THIF**	**FIFO Threshold Interrupt Status (Read Only)** 0 = The valid data count within the FIFO data buffer is more than the setting value of TH (DPWM_CTL[16:12]). 1 = The valid data count within the FIFO data buffer is less than or equal to the setting value of TH (DPWM_CTL[16:12]).
[1]	**EMPTY**	**FIFO Empty (Read Only)** 0 = FIFO is not empty. 1 = FIFO is empty.
[0]	**FULL**	**FIFO Full (Read Only)** 0 = FIFO is not full. 1 = FIFO is full.

DPWM PDMA Control Register (DPWM_PDMACTL)

Register	Offset	R/W	Description	Reset Value
DPWM_PDMACTL	DPWM_BA+0x08	R/W	DPWM 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**

Bits	Description	
[31:1]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[0]	**PDMAEN**	**PDMA Transfer Enable Bit** 0 = PDMA data transfer Disabled. 1 = PDMA data transfer Enabled.

DPWM FIFO Data Input Register (DPWM_FIFO)

Register	Offset	R/W	Description	Reset Value
DPWM_FIFO	DPWM_BA+0x0C	W	DPWM FIFO Data Input Register	0x0000_0000

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

Bits	Description	
[31:0]	**FIFO**	**FIFO Data Input Register** DPWM contains 32 words (32x32 bit) data buffer for data transmit. A write to this register pushes data onto the FIFO data buffer and increments the write pointer. This is the address that PDMA writes audio data to. The remaining word number is indicated by FIFOPTR (DPWM_STATUS[8:4]).

DPWM Zero Order Hold Division Register (DPWM_ZOHDIV)

Register	Offset	R/W	Description	Reset Value
DPWM_ZOHDIV	DPWM_BA+0x10	R/W	DPWM Zero Order Hold Division Register	0x0000_0804

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

Bits	Description	
[31:19]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[18:8]	**CLKDIV**	**Clock Divider** Divider to generate the DPWM_CLK F_DPWM_CLK = (F_DPWM_CLK_SRC) / (1 + CLKDIV) where F_ DPWM _CLK_SRC is the frequency of DPWM module clock source, which is defined in the clock control register DPWMSEL (CLK_CLKSEL2[13:12]) and F_DPWM_CLK is the frequency of DPWM module working clock (DPWM_CLK). **Note 1:** If fs is 48 kHz, the frequency of DPWM_CLK must be 24.576 MHz or 24 MHz according to the value of CLKSET (DPWM_CTL[31]). **Note 2:** If fs is 96 kHz, the frequency of DPWM_CLK must be 49.152 MHz or 48 MHz according to the value of CLKSET (DPWM_CTL[31]).
[7:0]	**ZOHDIV**	**Zero Order Hold, Down-sampling Divisor** The input sample rate of the DPWM is set by DPWM_CLK frequency and the divisor set in this register by the following formula: If CLKSET (DPWM_CTL[31]) is 0, K = 128. If CLKSET (DPWM_CTL[31]) is 1, K = 125. ZOHDIV = F_DPWM_CLK / (Fs * K). Where F_DPWM_CLK is the frequency of DPWM module working clock (DPWM_CLK) and Fs is sampling rate. **Note:** The value of ZOHDIV must be >= 4

DPWM Output Signal Frequency Control Register (DPWM_FREQ)

Register	Offset	R/W	Description	Reset Value
DPWM_FREQ	DPWM_BA+0x14	R/W	DPWM Output Signal Frequency 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					**STEPSEL**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**FREQSEL**	

Bits	Description	
[31:11]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[10:8]	**STEPSEL**	**Output Signal Frequency** 000 = Output signal frequency is 614 kHz. 001 = Output signal frequency is 512 kHz. 010 = Output signal frequency is 438 kHz. 011 = Output signal frequency is 384 kHz. 100 = Output signal frequency is 341 kHz. 101 = Output signal frequency is 307 kHz. Others = Reserved. Do not use.
[7:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1:0]	**FREQSEL**	**Output Signal FrequencySelection** 00 = Output signal frequency is 384 kHz. 01 = Output signal frequency is 307 kHz. 1X = Output signal frequency depends on STEPSEL (DPWM_FREQ[10:8]).

DPWM Coefficient Control Register (DPWM_COEFFCTL)

Register	Offset	R/W	Description	Reset Value
DPWM_COEFFCTL	DPWM_BA+0xFC	R/W	BIQ Coefficient 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						**COEFFFLTEN**	**PRGCOEFF**

Bits	Description	
[31:2]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[1]	**COEFFFLTEN**	**Coefficient Single Floating Point** 0 = Coefficient is fixed point 1 = Coefficient is single floating point
[0]	**PRGCOEFF**	**Coefficient Programming Control** 0 = Coefficient RAM is in normal mode. 1 = Coefficient RAM is under programming mode. **Note:** This bit must be truned off when BIQON (DPWM_CTL[21]) in on.

DPWM Coefficient Register (DPWM_COEFFn)

Register	Offset	R/W	Description	Reset Value
DPWM_COEFF0	DPWM_BA+0x100	R/W	Coefficient b0 Transfer function for band 1 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF1	DPWM_BA+0x104	R/W	Coefficient b1 Transfer function for band 1 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF2	DPWM_BA+0x108	R/W	Coefficient b2 Transfer function for band 1 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF3	DPWM_BA+0x10C	R/W	Coefficient a1 Transfer function for band 1 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF4	DPWM_BA+0x110	R/W	Coefficient a2 Transfer function for band 1 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF5	DPWM_BA+0x114	R/W	Coefficient b0 Transfer function for band 2 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF6	DPWM_BA+0x118	R/W	Coefficient b1 Transfer function for band 2 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF_7	DPWM_BA+0x11C	R/W	Coefficient b2 Transfer function for band 2 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF8	DPWM_BA+0x120	R/W	Coefficient a1 Transfer function for band 2 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF9	DPWM_BA+0x124	R/W	Coefficient a2 Transfer function for band 2 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF10	DPWM_BA+0x128	R/W	Coefficient b0 Transfer function for band 3 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF11	DPWM_BA+0x12C	R/W	Coefficient b1 Transfer function for band 3 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF12	DPWM_BA+0x130	R/W	Coefficient b2 Transfer function for band 3 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF13	DPWM_BA+0x134	R/W	Coefficient a1 Transfer function for band 3 fixed point – 3.21 format	0x0000_0000

			floating point – single precision point	
DPWM_COEFF14	DPWM_BA+0x138	R/W	Coefficient a2 Transfer function for band 3 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF15	DPWM_BA+0x13C	R/W	Coefficient b0 Transfer function for band 4 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF16	DPWM_BA+0x140	R/W	Coefficient b1 Transfer function for band 4 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF17	DPWM_BA+0x144	R/W	Coefficient b2 Transfer function for band 4 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF18	DPWM_BA+0x148	R/W	Coefficient a1 Transfer function for band 4 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF19	DPWM_BA+0x14C	R/W	Coefficient a2 Transfer function for band 4 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF20	DPWM_BA+0x150	R/W	Coefficient b0 Transfer function for band 5 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF21	DPWM_BA+0x154	R/W	Coefficient b1 Transfer function for band 5 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF22	DPWM_BA+0x158	R/W	Coefficient b2 Transfer function for band 5 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF23	DPWM_BA+0x15C	R/W	Coefficient a1 Transfer function for band 5 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF24	DPWM_BA+0x160	R/W	Coefficient a2 Transfer function for band 5 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF_25	DPWM_BA+0x164	R/W	Coefficient b0 Transfer function for band 6 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF26	DPWM_BA+0x168	R/W	Coefficient b1 Transfer function for band 6 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF27	DPWM_BA+0x16C	R/W	Coefficient b2 Transfer function for band 6 fixed point – 3.21 format floating point – single precision point	0x0000_0000

DPWM_COEFF28	DPWM_BA+0x170	R/W	Coefficient a1 Transfer function for band 6 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF29	DPWM_BA+0x174	R/W	Coefficient a2 Transfer function for band 6 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF30	DPWM_BA+0x178	R/W	Coefficient b0 Transfer function for band 7 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF31	DPWM_BA+0x17C	R/W	Coefficient b1 Transfer function for band 7 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF32	DPWM_BA+0x180	R/W	Coefficient b2 Transfer function for band 7 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF33	DPWM_BA+0x184	R/W	Coefficient a1 Transfer function for band 7 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF34	DPWM_BA+0x188	R/W	Coefficient a2 Transfer function for band 7 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF35	DPWM_BA+0x18C	R/W	Coefficient b0 Transfer function for band 8 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF36	DPWM_BA+0x190	R/W	Coefficient b1 Transfer function for band 8 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF37	DPWM_BA+0x194	R/W	Coefficient b2 Transfer function for band 8 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF38	DPWM_BA+0x198	R/W	Coefficient a1 Transfer function for band 8 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF39	DPWM_BA+0x19C	R/W	Coefficient a2 Transfer function for band 8 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF40	DPWM_BA+0x1A0	R/W	Coefficient b0 Transfer function for band 9 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF41	DPWM_BA+0x1A4	R/W	Coefficient b1 Transfer function for band 9 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF42	DPWM_BA+0x1A8	R/W	Coefficient b2 Transfer function for band 9	0x0000_0000

			fixed point – 3.21 format floating point – single precision point	
DPWM_COEFF43	DPWM_BA+0x1AC	R/W	Coefficient a1 Transfer function for band 9 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF44	DPWM_BA+0x1B0	R/W	Coefficient a2 Transfer function for band 9 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF45	DPWM_BA+0x1B4	R/W	Coefficient b0 Transfer function for band 10 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF46	DPWM_BA+0x1B8	R/W	Coefficient b1 Transfer function for band 10 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF47	DPWM_BA+0x1BC	R/W	Coefficient b2 Transfer function for band 10 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF48	DPWM_BA+0x1C0	R/W	Coefficient a1 Transfer function for band 10 fixed point – 3.21 format floating point – single precision point	0x0000_0000
DPWM_COEFF49	DPWM_BA+0x1C4	R/W	Coefficient a2 Transfer function for band 10 fixed point – 3.21 format floating point – single precision point	0x0000_0000

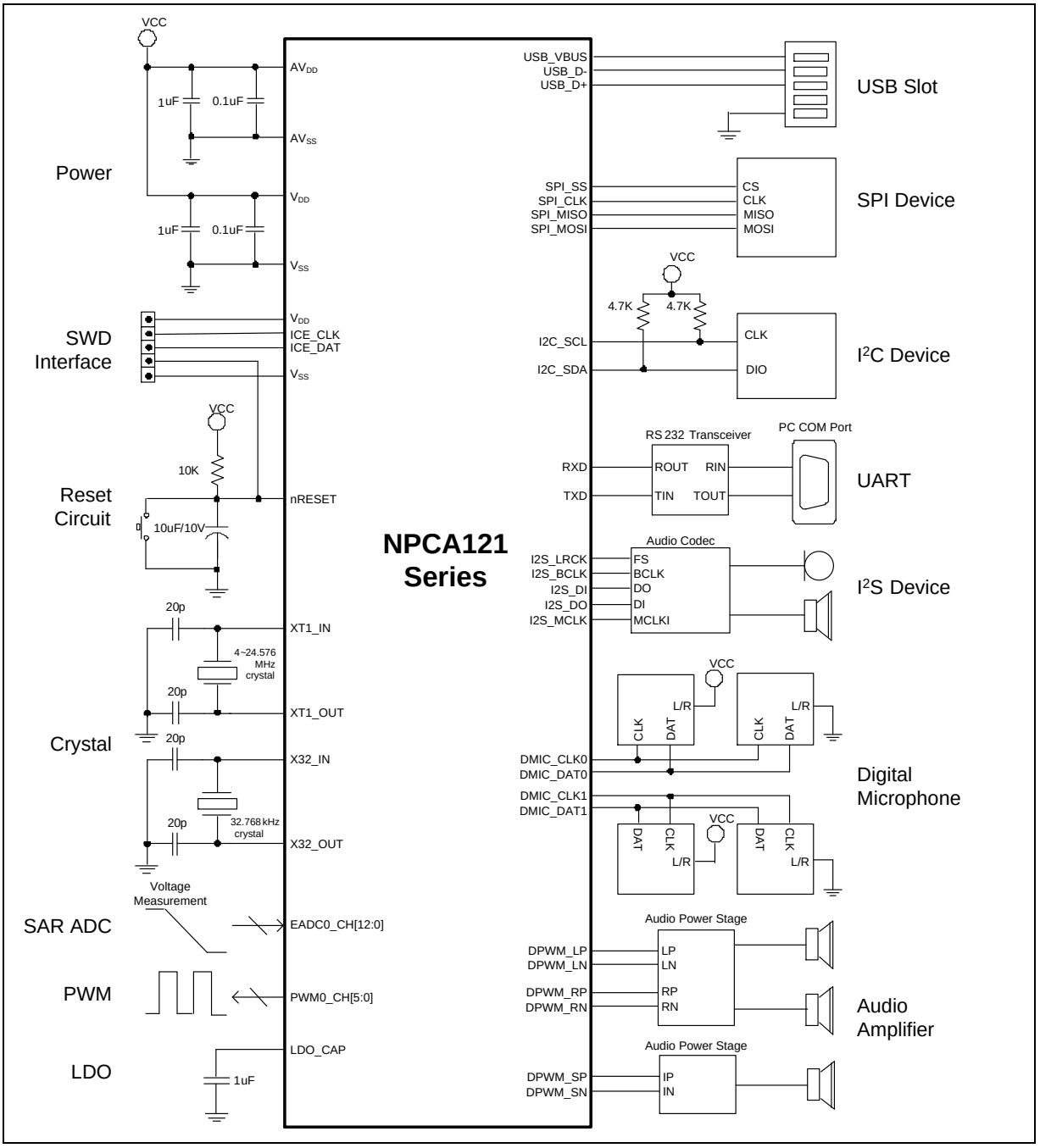
31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
COEFFDAT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
COEFFDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
COEFFDAT							

Bits	Description	
[31:24]	**Reserved**	Reserved. Any values read should be ignored. When writing to this field always write with reset value.
[23:0]	**COEFFDAT**	Coefficient Data.

7 ELECTRICAL CHARACTERISTICS

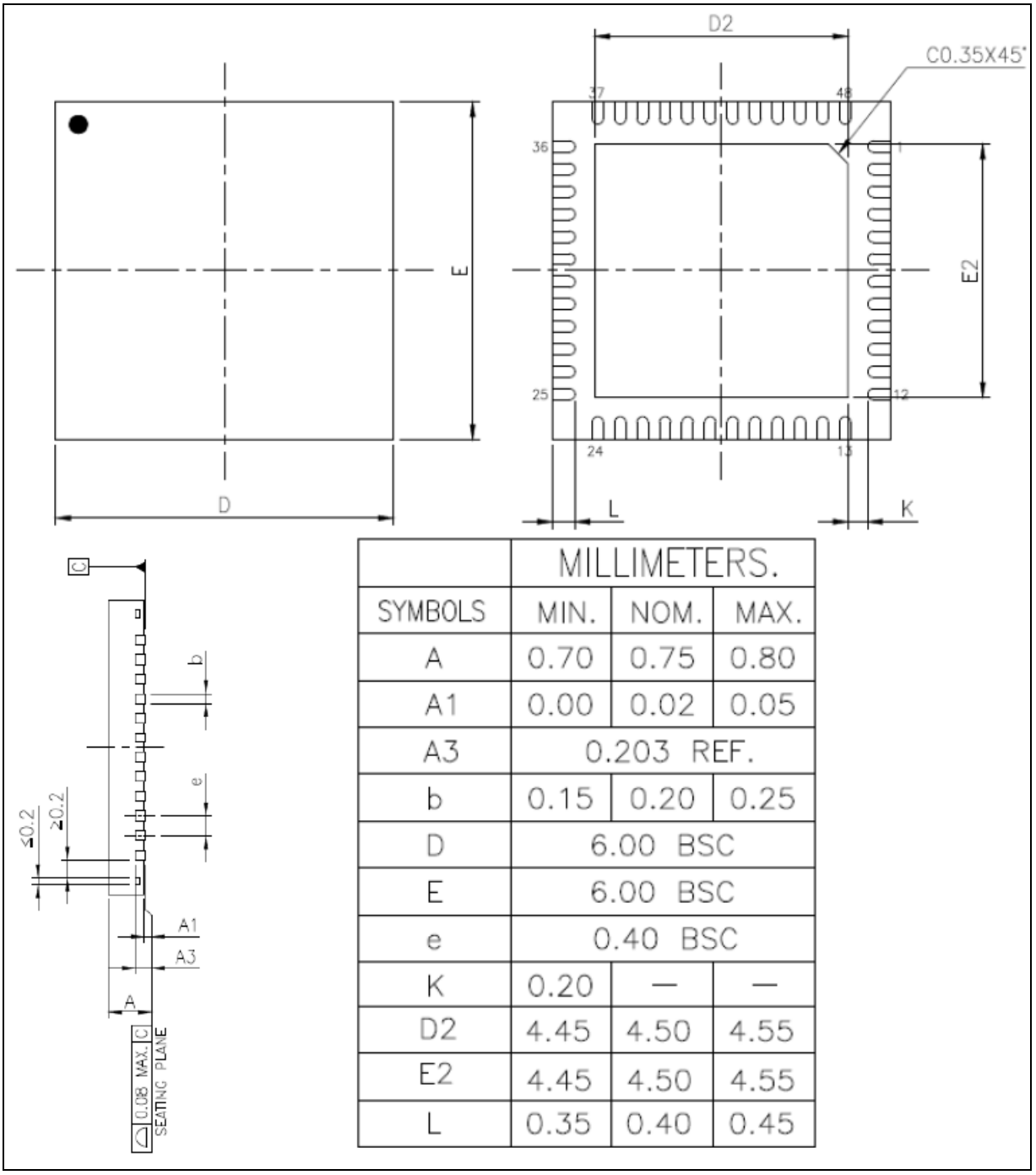
For information on NPCA121 series electrical characteristics, please refer to NPCA121 series Audio Enhancing Engine Datasheet.

8 APPLICATION CIRCUIT



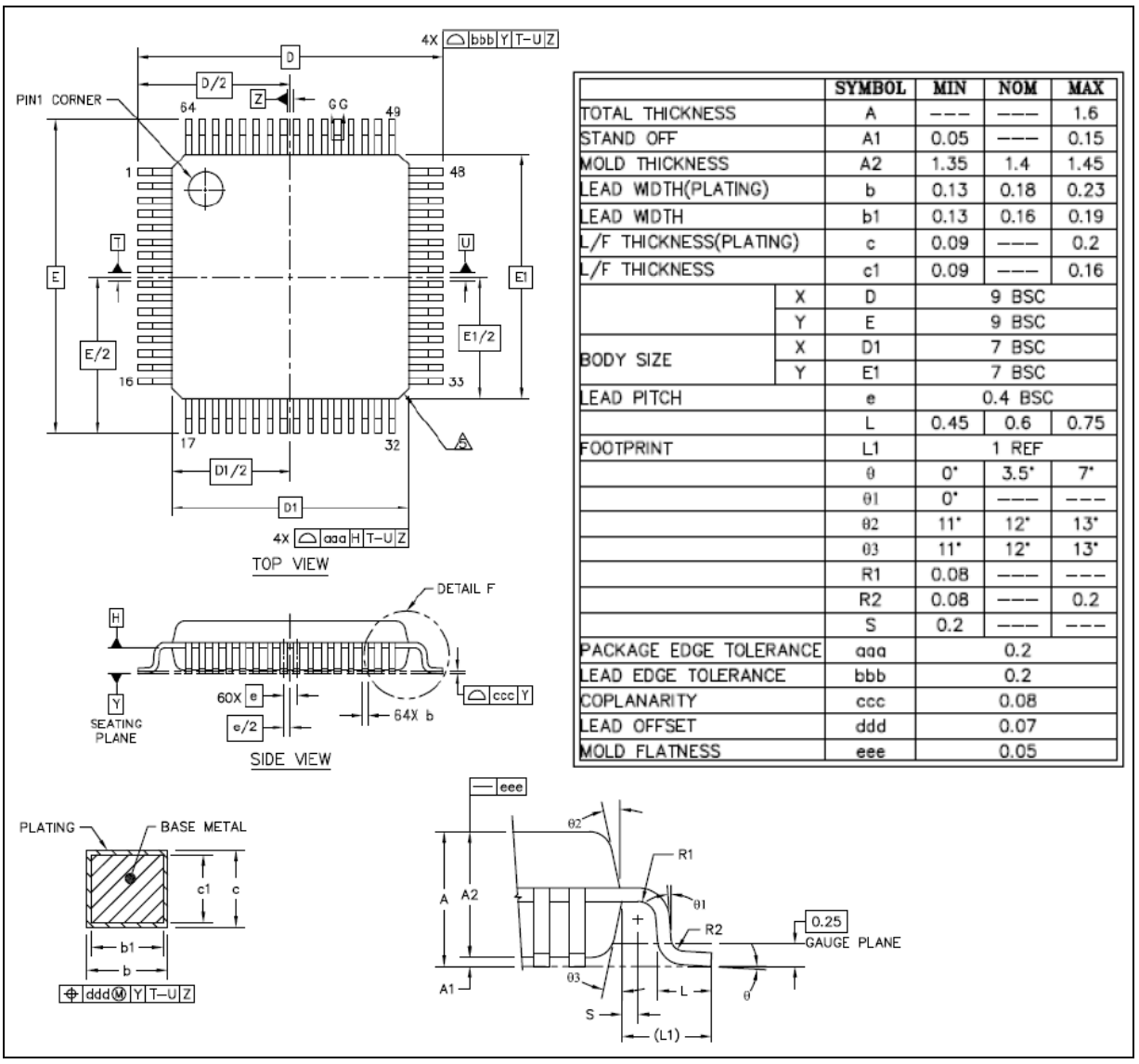
9 PACKAGE DIMENSIONS

9.1 QFN 48L (6x6x0.8 mm^3 Pitch 0.4 mm)



	MILLIMETERS.		
SYMBOLS	MIN.	NOM.	MAX.
A	0.70	0.75	0.80
A1	0.00	0.02	0.05
A3		0.203 REF.	
b	0.15	0.20	0.25
D		6.00 BSC	
E		6.00 BSC	
e		0.40 BSC	
K	0.20	—	—
D2	4.45	4.50	4.55
E2	4.45	4.50	4.55
L	0.35	0.40	0.45

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



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

10 REVISION HISTORY

Date	Revision	Description
2019.11.05	0.1	1. Preliminary version release

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.