NuMicro® Family

Arm® Cortex®-M4-based Microcontroller

M471V/M471K/M471C Series
Technical Reference Manual

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

www.nuvoton.com

TABLE OF CONTENTS

LIST OF FIGURES

List of Tables

1 GENERAL DESCRIPTION

The NuMicro® M471V/M471K/M471C series is a 32-bit microcontroller based on Arm® Cortex®-M4F core, with DSP instruction set and single-precision floating-point unit (FPU), targeted for smart home appliance applications. For the growing requirement of the safety functions on the home appliance, the M471V/M471K/M471C series provides certified Software Test Library (STL) and an application note for IEC60730-1 Class B Annex H. This certified STL can significantly reduce the development time and efforts to pass IEC60730-1 Class B certification for home appliances. The M471V/M471K/M471C series runs up to 120 MHz, and features 2.5V to 5.5V wide operating voltage, -40°C to 105°C wide operating temperature, a variety of packages with wide pin pitch, and excellent high immunity characteristics by ESD HBM 8 KV and EFT 4.4 KV, which greatly meet the rigid requirements for stability, reliability and safety of home appliance systems.

As the new smart function added on home appliances, the M471V/M471K/M471C series provides up to 512 KB dual-bank of Flash memory for code storage, 64 KB SRAM for run time operation and 32 KB independent Data Flash for parameters. The dual bank design of 512 KB Flash memory supports the Firmware update through the Over-The-Air (FOTA) process. Additionally, in response to the code security requirements, the M471V/M471K/M471C series supports Execute-Only Memory (XOM) function to protect confidential program code information from stealing in the run-time. Finally, the 32 KB independent data Flash provides a 256 Bytes page erasing unit to make the parameter data update and access more efficiently. In order to reduce the data access overhead of CPU core to peripherals, a peripheral direct memory access (PDMA) is provided.

The M471V/M471K/M471C series supports plenty of peripherals, including a Customize IR receiver (CIR) for remote controller, up to 24 channels of 16-bit PWM, 6 sets of UART, 2 sets of SPI/I²S, 2 sets of I²C, and a real-time clock (RTC).

The M471V/M471K/M471C series also provides rich analog peripherals including 2 sets of analog comparators, up to 24 channels of 12-bit SAR ADC, and 1 channel of 8-bit DAC. The M471V/M471K/M471C series also integrates a pseudo random number generator (PRNG) to support the requirement for encryption and decryption of smart home appliances.

For the development, Nuvoton provides the NuMaker-M471KI evaluation board, and Nuvoton Nu-Link debugger. The 3rd Party IDE such as Keil MDK, IAR EWARM, Eclippse IDE with GNU GCC compilers are also supported.

Product Line	Core (MHz)	PDMA	CRC	Timer (32-bit)	PWM	UART	I²C	QSPI/ SPI	ADC	ACMP	DAC	CIR	PRNG
M471V/M471K/M471C Series	120	√	√	4	24	6	2	SPI x2	24	2	1	√	√

Table 1-1 NuMicro® M471V/M471K/M471C Series Key Features Support Table

This series supports two package choices which are designed for home appliance PCB demands.

- LQFP100: 14 mm x 14 mm, pin pitch 0.5 mm
- LQFP128: 14 mm x 14 mm, pin pitch 0.4 mm
- WLCSP100: 4.4648x 4.4480x 0.58 mm, ball pitch 0.4 mm

The NuMicro® M471V/M471K/M471C series is suitable for a wide range of applications such as:

- Washing Machine
- Refrigerator
- Air conditioner
- Air Purifier
- Other home appliances

2 FEATURES

2.1 NuMicro® M471V/M471K/M471C Features

Core and System	
Arm® Cortex®-M4	• Arm® Cortex®-M4 processor, running up to 120 MHz • Built-in Memory Protection Unit (MPU) • Built-in Nested Vectored Interrupt Controller (NVIC) • Hardware IEEE 754 compliant Floating-point Unit (FPU) • DSP extension with hardware divider and single-cycle 32-bit hardware multiplier • 24-bit system tick timer • Programmble and maskable interrupt • Low Power Sleep mode by WFI and WFE instructions
Brown-out Detector (BOD)	• Four-level BOD with brown-out interrupt and reset option (4.4V/3.7V/2.7V/2.4V)
Low Voltage Reset (LVR)	• LVR with 2.35V threshold voltage level
Security	• 96-bit Unique ID (UID). • 128-bit Unique Customer ID (UCID).
Memories	
Flash	• Two bank 512 KB on-chip Application ROM (APROM) • Supports FOTA function • 4 KB on-chip Flash for user-defined loader (LDROM) • 32 Kbytes data Flash which is rewritable in parallel with instruction execution • All on-chip Flash expect data Flash supporting 2048-byte page erase • Data Flash supporting 256-byte page erase • Fast Flash programming verification with CRC • On-chip Flash programming with In-Chip Programming (ICP), In-System Programming (ISP) and In-Application Programming (IAP) capabilities • Configurable boot up sources user-defined loader (LDROM) or Application ROM (APROM) • 2-wired ICP Flash updating through SWD interface • 32-bit/64-bit and multi-word Flash programming function
SRAM	• Up to 64 KB on-chip SRAM includes:

	• Byte-, half-word- and word-access • 32 KB with hardware parity check • Supports PDMA operation
Cyclic Redundancy Calculation (CRC)	• Supports CRC-CCITT, CRC-8, CRC-16 and CRC-32 polynomials • Programmable initial value and seed value • Programmable order reverse setting and one's complement setting for input data and CRC checksum • 8-bit, 16-bit, and 32-bit data width • 8-bit write mode with 1-AHB clock cycle operation • 16-bit write mode with 2-AHB clock cycle operation • 32-bit write mode with 4-AHB clock cycle operation • Uses PDMA to write data with performing CRC operation
Peripheral DMA (PDMA)	• Six independent and configurable channels for automatic data transfer between memories and peripherals • Basic and Scatter-Gather transfer modes • Each channel supports circular buffer management using Scatter-Gather Transfer mode • Fixed-priority and Round-robin priorities modes • Single and burst transfer types • Byte-, half-word- and word tranfer unit with count up to 65536 • Incremental or fixed source and destination address
Pseudo Random Number Generator (PRNG)	• Supports 64 bits, 128 bits, 192 bits, and 256 bits random number generation.
Clocks	
External Clock Source	• 4~24 MHz High-speed eXternal crystal oscillator (HXT) for precise timing operation • 32.768 kHz Low-speed eXternal crystal oscillator (LXT) for RTC function and low-power system operation • Supports clock failure detection for external crystal oscillators and exception generatation (NMI)
Internal Clock Source	• 38 kHz Low-speed Internal RC oscillator (LIRC) for watchdog timer and wakeup operation • 48 MHz High-speed Internal RC oscillator (HIRC) trimmed to 1% accuracy at -20 to 85°C • Up to 120 MHz on-chip PLL, sourced from HIRC or HXT, allowing CPU operation up to the maximum CPU frequency without the need for a high-frequency crystal
Real-Time Clock (RTC)	• The RTC clock source is from Low-speed external crystal • Able to wake up CPU from any reduced power mode

	• Supports ±5 ppm within 5 seconds software clock accuracy compensation • Supports Alarm registers (second, minute, hour, day, month, year) • Supports RTC Time Tick and Alarm Match interrupt • Automatic leap year recognition • Supports 1 Hz clock output for calibration
Timers	
32-bit Timer	**TIMER** • 4 sets of 32-bit timers with 24-bit up counters and 8-bit prescale counters • Independent clock source for each timer • One-shot, Periodic, Toggle and Continuous Counting operation modes • Event counting function to count the event from external pin • Input capture function to capture or reset counter value • External capture pin event for interval measurement • External capture pin event to reset 24-bit up counter • Chip wake-up from Idle/Power-down mode if a timer interrupt signal is generated • Timer interrupt flag or external capture interrupt flag to trigger BPWM, EPWM, EADC, DAC and PDMA • Internal capture triggered source from ACMP output • Inter-Timer trigger capture mode **PWM** • 16-bit compare register and period register • Double buffer for period register and compare register • Supports inverse in PWM output • PWM interrupt wake-up from system Power-down mode
Enhanced PWM (EPWM)	• 16-bit counters with 12-bit clock presacle supporting 12 PWM output channels • Up to 12 independent input capture channels with 16-bit resolution counter • Supports dead time with maximum divided 12-bit prescale • Up, down or up-down PWM counter type • Supports complementary mode for 3 complementary paired PWM output channels • Synchronous function for phase control • Counter synchronous start function • Brake function with auto recovery mechanism • Mask function and tri-state output for each PWM channel • Trigger EADC or DAC to start conversion immediately

	• Trigger EADC to start conversion after a short delay • Hardware short-circuit output check
Basic PWM (BPWM)	• 16-bit counters with 12-bit clock presacle supporting 12 PWM output channels • Up to 12 independent input capture channels with 16-bit resolution counter • Up, down or up-down PWM counter type • Counter synchronous start function • Complementary mode for 6 complementary paired PWM output channels • Mask function and tri-state output for each PWM channel • Able to trigger EADC to start conversion
Watchdog	• 18-bit free running up counter for WDT time-out interval • Supports multiple clock sources from LIRC (default selection), HCLK/2048 and LXT with 8 selectable time-out period • Able to wake up system from Power-down or Idle mode • Time-out event to trigger interrupt or reset system • Supports four WDT reset delay periods, including 1026, 130, 18 or 3 WDT_CLK reset delay period • Configured to force WDT enabled on chip power-on or reset
Window Watchdog	• Clock sourced from HCLK/2048 or LIRC; the window set by 6-bit counter with 11-bit prescale • Suspended in Idle/Power-down mode
Analog Interfaces	
Enhanced Analog-to-Digital Converter (EADC)	• One 12-bit, 24-ch 1.895 MSPS SAR EADC with up to 24 single-ended input channels or 1 differential input pair • Three internal channels for band-gap VBG input, Temperature sensor input and DAC input • Supports external V_{REF} pin or internal reference voltage • Analog-to-Digital conversion can be triggered by software enable, external pin, Timer 0~3 overflow pulse trigger or PWM trigger • Configurable EADC sampling time • Double data buffers for sample module 0~3 • Supports PDMA operation
Digital-to-Analog Converter (DAC)	• One 8-bit, 200 KSPS voltage type DAC without buffer • Reference voltage can be switched from AV_{DD}, V_{REF} pin and Internal reference voltage • The trigger of the DAC conversion can be done by the software, external triggers or timer0 to timer3, EPWM0 • Available for monitoring by the ACMP

	• PDMA operation
Analog Comparator (ACMP)	• Two Analog Comparators • Supports four multiplexed I/O pins at positive input • Supports one I/O pin, band-gap voltage, DAC voltage output, internal voltage reference, and 16-level voltage divider from AV_{DD} or V_{REF} at negative input • Supports wake up from Power-down by interrput • Supports triggers for brake events and cycle-by-cycle control for PWM • Supports window compare mode and window latch mode • Supports programmable hysteresis window: 20 mV and 40 mV

Internal reference voltage	• One Internal reference voltage with 2.048V, 2.560V, 3.072V, 4.096V
	• Can be configured to supply a reference voltage to the following:
	  • ADC reference Voltage
	  • DAC reference Voltage
	  • Comparator negative input

Temperature sensor	• One built-in temperature sensor with 1°C resolution
Communication Interfaces	
Low-power UART	• Six Low-power UARTs with up to 2 MHz baud rate • Auto-Baud Rate measurement and baud rate compensation function • Supports low power UART (LPUART): baud rate clock from LXT(32.768 kHz) with 9600bps in Power-down mode even system clock is stopped • UART0/UART1 supporting 16-byte FIFOs with programmable level trigger • UART2/UART3/UART4/UART5 supporting 1-byte FIFOs with programmable level trigger • Auto flow control (nCTS and nRTS) • Supports IrDA (SIR) function on UART0/UART1 • Supports LIN function on UART0/UART1 • Supports RS-485 9-bit mode and direction control • Supports nCTS, incoming data, Received Data FIFO reached threshold and RS-485 Address Match (AAD mode) wake-up function in idle mode • Supports hardware or software enabled to program nRTS pin to control RS-485 transmission direction • Supports wake-up function • 8-bit receiver FIFO time-out detection function • Supports break error, frame error, parity error and receive/transmit FIFO overflow detection function

	• PDMA operation
I²C	• Two sets of I²C devices with Master/Slave mode • Supports Standard mode (100 kbps), Fast mode (400 kbps), Fast mode plus (1 Mbps) • Programmable clocks allowing for 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 • PDMA operation
SPI/I²S	• Two SPI/I²S controllers with Master/Slave mode • SPI/I²S provides separate 4-level of 32-bit (or 8-level of 16-bit) transmit and receive FIFO buffers **SPI** • Up to 24 MHz in Master mode • Configurable bit length of a transfer word from 8 to 32-bit • MSB first or LSB first transfer sequence • Byte reorder function • Supports Byte or Word Suspend mode • Supports one data channel half-duplex transfer • Supports receive-only mode **I²S** • Supports mono and stereo audio data with 8-, 16-, 24- and 32-bit audio data sizes • Supports PCM mode A, PCM mode B, I²S and MSB justified data format • PDMA operation
Customize IR Receiver (CIR)	• 1 channel • Four pattern matching (header, data 0, data 1, and special data detection) • 8-byte receive buffer per unit
GPIO	• Supports four I/O modes: Quasi bi-direction, Push-Pull output, Open-Drain output and Input only with high impendence mode • Selectable TTL/Schmitt trigger input • Configured as interrupt source with edge/level trigger setting • Supports independent pull-up control • Supports high driver and high sink current I/O • Supports software selectable slew rate control

3 PARTS INFORMATION

Package is Halogen-free, RoHS-compliant and TSCA-compliant.

3.1 Package Type

WLCSP100	LQFP100	LQFP128
M471CI8AE	M471VI8AE	M471KI8AE

3.2 M471V/M471K/M471C Series Naming Rule

M4	71	K	I	8	A	E
Core	**Series**	**Package**	**Flash Size**	**SRAM Size**	**Revision**	**Temperature**
Cortex®-M4F	71: Base	V: LQFP100 (14x14 mm) K: LQFP128 (14x14 mm) C: WLCSP100 (4.4648x4.448mm)	G: 256 KB I: 512 KB	7: 48 KB 8: 64 KB		E:-40˚C ~ 105˚C

3.3 M471V/M471K/M471C Series Selection Guide

PART NUMBER		M471		
		VI8AE	KI8AE	CI8AE
Flash (KB)		512 (dual bank)	512 (dual bank)	512 (dual bank)
SRAM (KB)		64 (32 KB hardware parity check)	64 (32 KB hardware parity check)	64 (32 KB hardware parity check)
Data Flash (KB)		32		
LDROM (KB)		4		
XOM (regions)		4		
System Frequency (MHz)		120		
I/O		91	119	83
32-bit Timer		4		
Customize IR Receiver		√		
Connectivity	UART	6		
	SPI/I²S	2		
	I²C	2		
16-bit EPWM		12		
16-bit BPWM		12		
PMDA		6-ch		
RTC		√		
12-bit ADC		23	24	24
8-bit DAC		1		
Analog Comparator		2		
PRNG		√		
Package		LQFP 100 (14x14mm)	LQFP 128 (14x14mm)	C: WLCSP100 (4.4648x4.448mm)

4 PIN CONFIGURATION

Users can find pin configuaration information in the Multi-function Pin Diagram section or by using NuTool - PinConfig. The NuTool - PinConfig contains all Nuvoton NuMicro® Family chip series with all part number, and helps users configure GPIO multi-function correctly and handily.

4.1 Pin Configuration

4.1.1 Pin Diagram

4.1.1.1 LQFP100 Pin Diagram

Corresponding Part Number: M471VI8AE

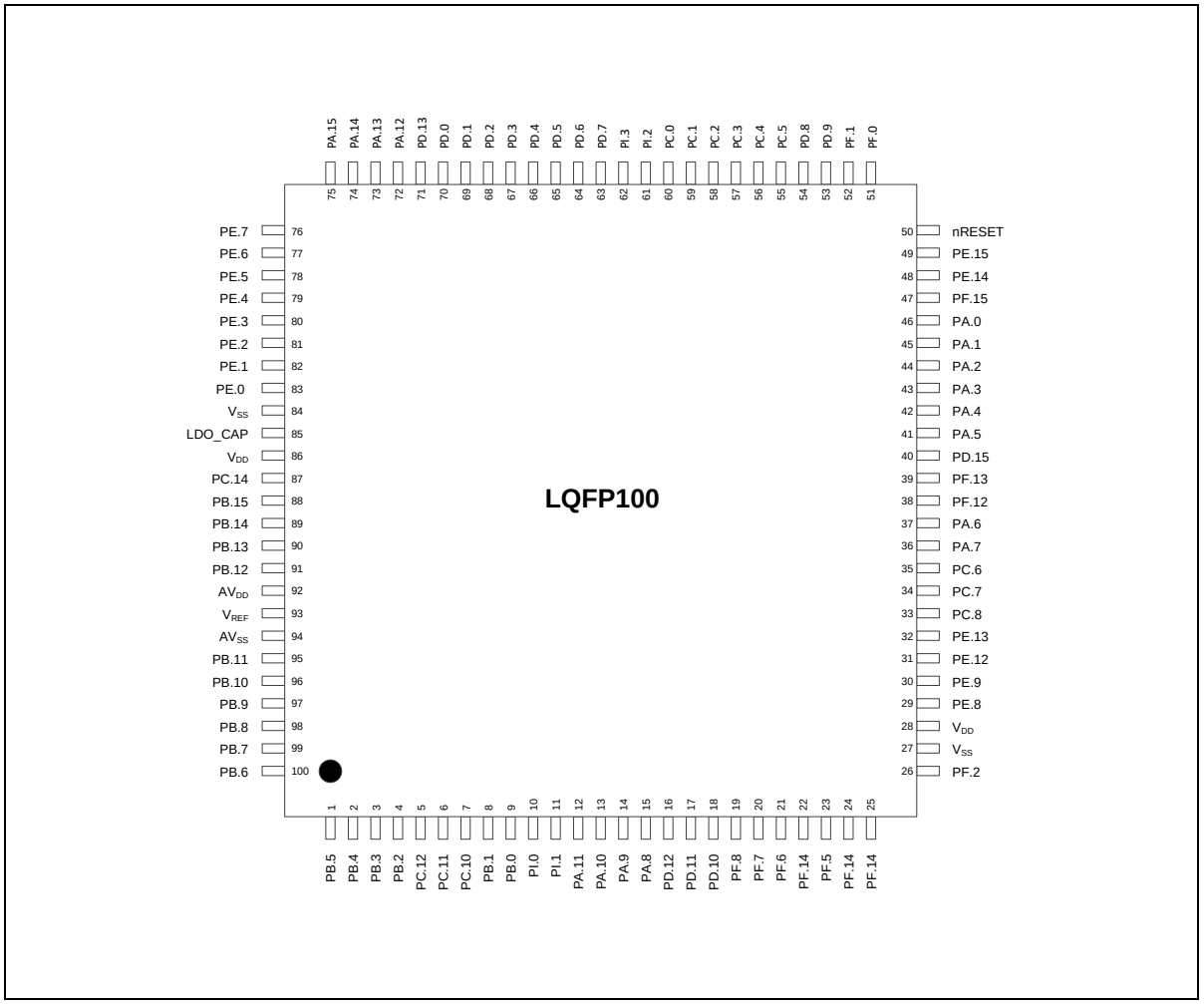


Figure 4.1-1 LQFP 100-pin Diagram

4.1.1.2 LQFP128 Pin Diagram

Corresponding Part Number: M471KI8AE

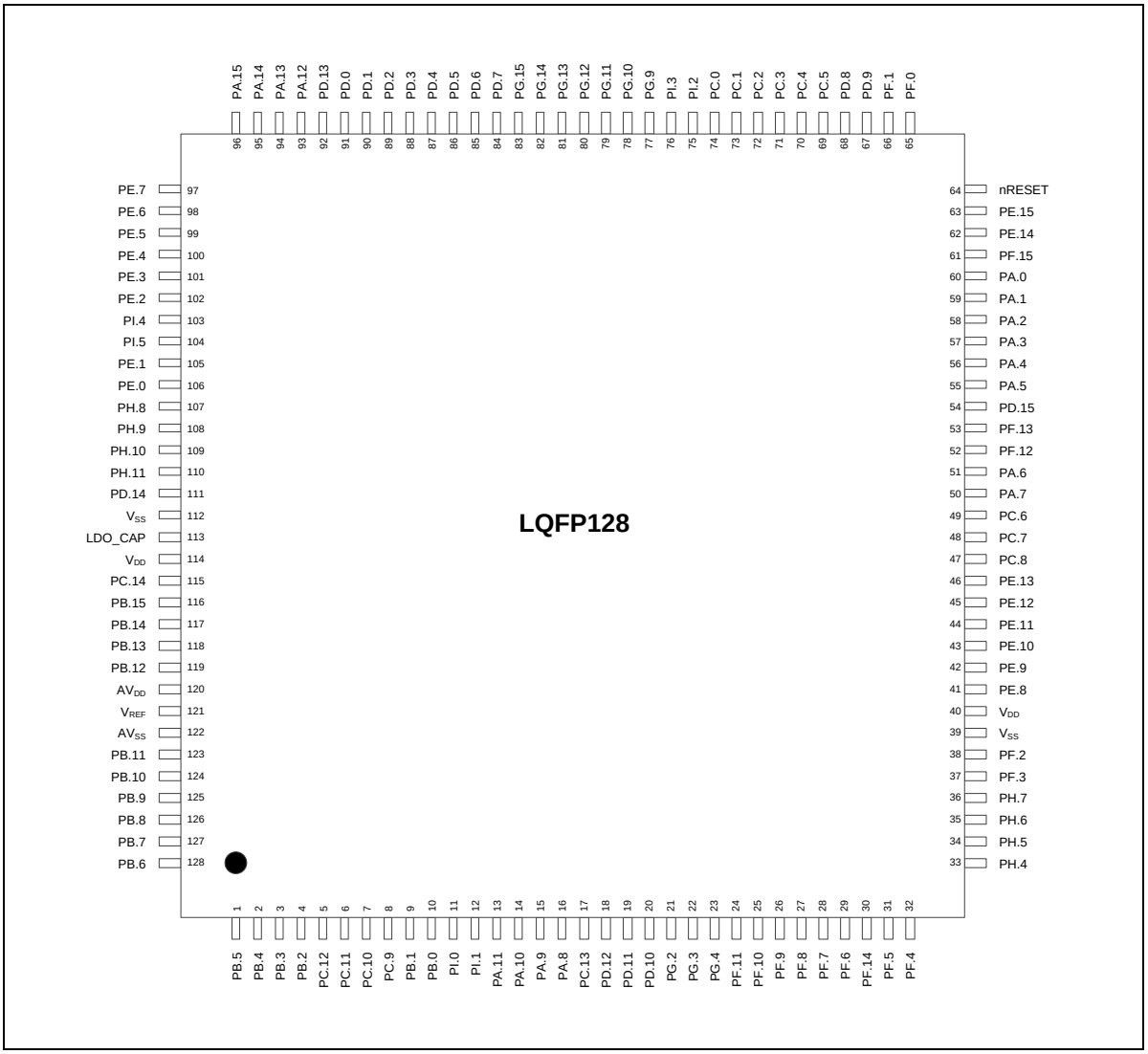


Figure 4.1-2 LQFP 128-pin Diagram

4.1.1.3 WLCSP100 Pin Diagram

Corresponding Part Number: M471CI8AE

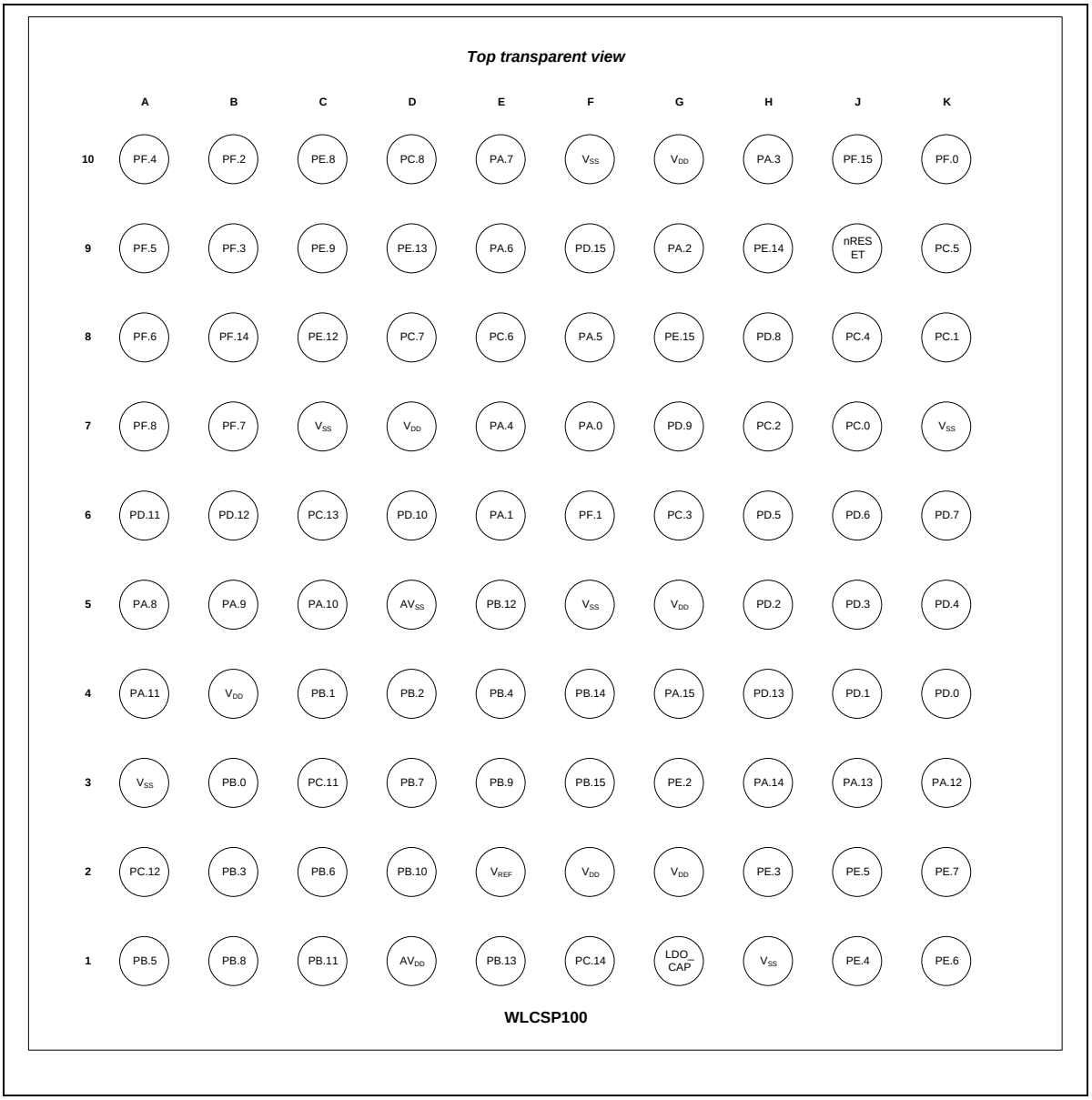


Figure 4.1-3 WLCSP 100-pin Diagram

4.1.2 Multi-function Pin Diagram

4.1.2.1 LQFP-100 Pin Multi-function Pin Diagram

Corresponding Part Number: M471VI8AE

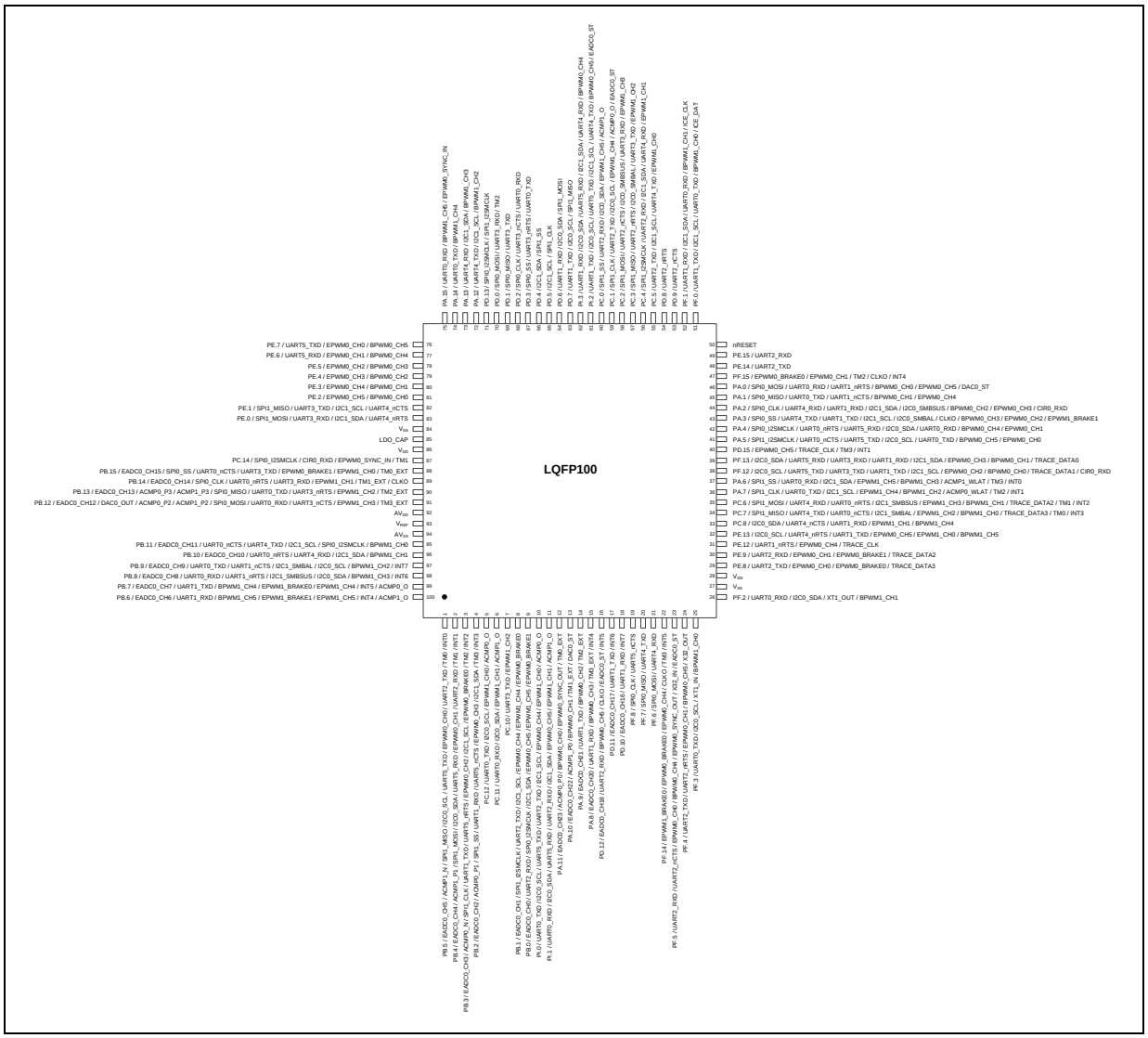


Figure 4.1-4 LQFP 100-pin Multi-function Pin Diagram

4.1.2.2 LQFP-128 Pin Multi-function Pin Diagram

Corresponding Part Number: M471KI8AE

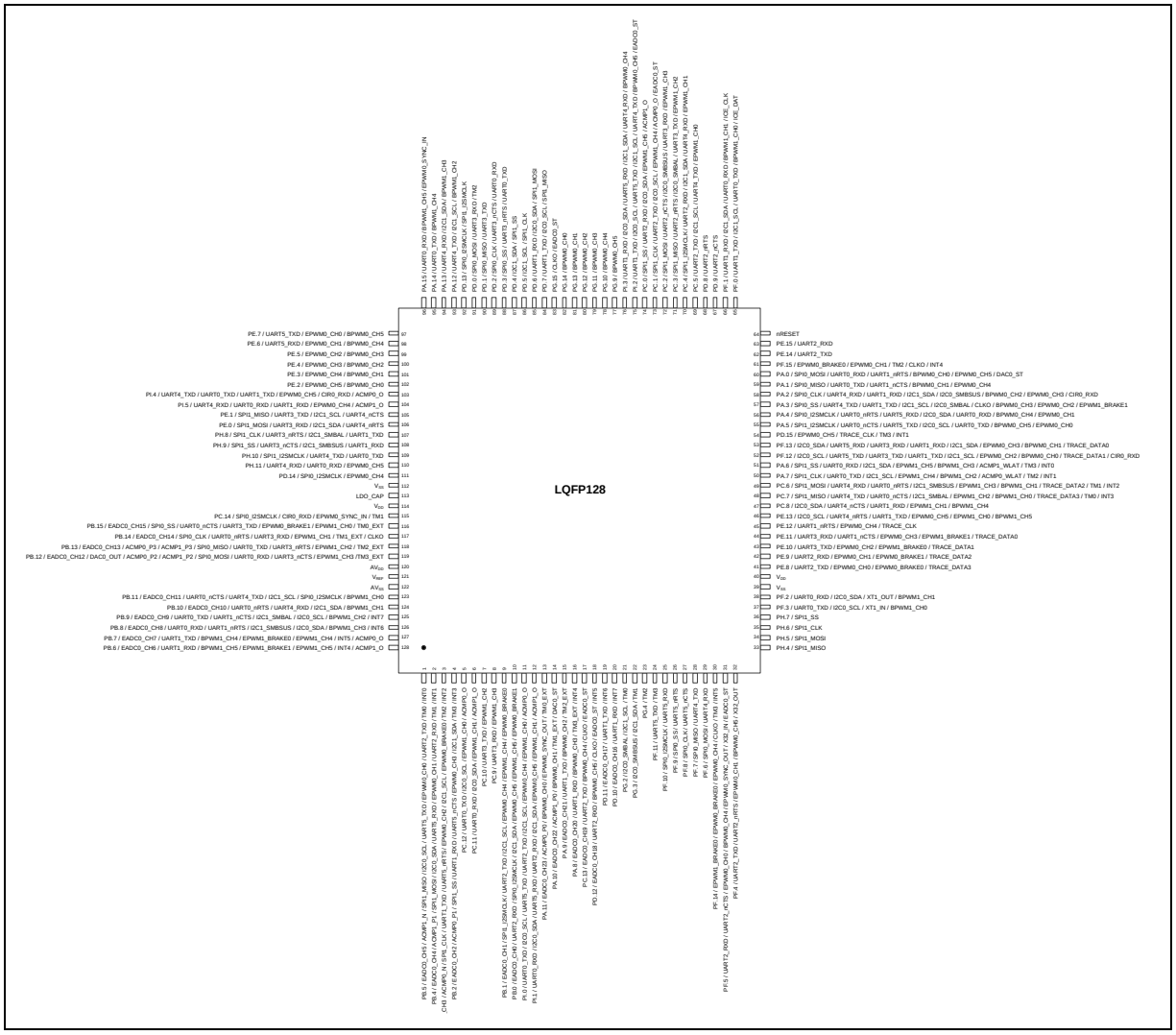


Figure 4.1-5 LQFP 128-pin Multi-function Pin Diagram

4.2 Pin Mapping

Different part number with same package might has different function. Please refer to the selection guide in section 3.3, Pin Configuration in section 4.1 or NuTool - PinConfig.

Corresponding Part Number: M471VI8AE, M471KI8AE, M471CI8AE

	M471		
Pin Name	100 Pin	128 Pin	WLCSP 100 Pin
PB.5	1	1	A1
PB.4	2	2	E4
PB.3	3	3	B2
PB.2	4	4	D4
PC.12	5	5	A2
PC.11	6	6	C3
PC.10	7	7	
PC.9		8	
PB.1	8	9	C4
PB.0	9	10	B3
PI.0	10	11	
V_{SS}			A3
V_{DD}			B4
PI.1	11	12	
PA.11	12	13	A4
PA.10	13	14	C5
PA.9	14	15	B5
PA.8	15	16	A5
PC.13		17	C6
PD.12	16	18	B6
PD.11	17	19	A6
PD.10	18	20	D6
PG.2		21	
PG.3		22	
PG.4		23	
PF.11		24	
PF.10		25	
PF.9		26	
PF.8	19	27	A7

PF.7	20	28	B7
PF.6	21	29	A8
PF.14	22	30	B8
PF.5	23	31	A9
PF.4	24	32	A10
PH.4		33	
PH.5		34	
PH.6		35	
PH.7		36	
PF.3	25	37	B9
PF.2	26	38	B10
V_{SS}	27	39	C7
V_{DD}	28	40	D7
PE.8	29	41	C10
PE.9	30	42	C9
PE.10		43	
PE.11		44	
PE.12	31	45	C8
PE.13	32	46	D9
PC.8	33	47	D10
PC.7	34	48	D8
PC.6	35	49	E8
PA.7	36	50	E10
PA.6	37	51	E9
PF.12	38	52	
V_{SS}			F10
V_{DD}			G10
PF.13	39	53	
PD.15	40	54	F9
PA.5	41	55	F8
PA.4	42	56	E7
PA.3	43	57	H10
PA.2	44	58	G9
PA.1	45	59	E6
PA.0	46	60	F7

PF.15	47	61	J10
PE.14	48	62	H9
PE.15	49	63	G8
nRESET	50	64	J9
PF.0	51	65	K10
PF.1	52	66	F6
PD.9	53	67	G7
PD.8	54	68	H8
PC.5	55	69	K9
PC.4	56	70	J8
PC.3	57	71	G6
PC.2	58	72	H7
PC.1	59	73	K8
PC.0	60	74	J7
PI.2	61	75	
V_{SS}			K7
V_{DD}			G5
PI.3	62	76	
PG.9		77	
PG.10		78	
PG.11		79	
PG.12		80	
PG.13		81	
PG.14		82	
PG.15		83	
PD.7	63	84	K6
PD.6	64	85	J6
PD.5	65	86	H6
PD.4	66	87	K5
PD.3	67	88	J5
PD.2	68	89	H5
PD.1	69	90	J4
PD.0	70	91	K4
PD.13	71	92	H4
PA.12	72	93	K3

PA.13	73	94	J3
PA.14	74	95	H3
PA.15	75	96	G4
PE.7	76	97	K2
PE.6	77	98	K1
PE.5	78	99	J2
PE.4	79	100	J1
PE.3	80	101	H2
PE.2	81	102	G3
V_{SS}			H1
V_{DD}			G2
PI.4		103	
PI.5		104	
PE.1	82	105	
PE.0	83	106	
PH.8		107	
PH.9		108	
PH.10		109	
PH.11		110	
PD.14		111	
V_{SS}	84	112	F5
LDO_CAP	85	113	G1
V_{DD}	86	114	F2
PC.14	87	115	F1
PB.15	88	116	F3
PB.14	89	117	F4
PB.13	90	118	E1
PB.12	91	119	E5
AV_{DD}	92	120	D1
V_{REF}	93	121	E2
AV_{SS}	94	122	D5
PB.11	95	123	C1
PB.10	96	124	D2
PB.9	97	125	E3
PB.8	98	126	B1

PB.7	99	127	D3
PB.6	100	128	C2

4.3 Pin Functional Description

Corresponding Part Number: M471VI8AE, M471KI8AE, M471CI8AE

4.3.1 Multi-function Summary Table

Group	Pin Name	GPIO	MFP	Type	Description
ACMP0	ACMP0_N	PB.3	MFP1	A	Analog comparator 0 negative input pin.
	ACMP0_O	PC.12	MFP14	O	Analog comparator 0 output pin.
		PI.0	MFP14	O	
		PC.1	MFP14	O	
		PI.4	MFP15	O	
		PB.7	MFP15	O	
	ACMP0_P0	PA.11	MFP1	A	Analog comparator 0 positive input 0 pin.
	ACMP0_P1	PB.2	MFP1	A	Analog comparator 0 positive input 1 pin.
	ACMP0_P2	PB.12	MFP1	A	Analog comparator 0 positive input 2 pin.
	ACMP0_P3	PB.13	MFP1	A	Analog comparator 0 positive input 3 pin.
	ACMP0_WLAT	PA.7	MFP13	I	Analog comparator 0 window latch input pin
ACMP1	ACMP1_N	PB.5	MFP1	A	Analog comparator 1 negative input pin.
	ACMP1_O	PC.11	MFP14	O	Analog comparator 1 output pin.
		PI.1	MFP14	O	
		PC.0	MFP14	O	
		PI.5	MFP15	O	
		PB.6	MFP15	O	
	ACMP1_P0	PA.10	MFP1	A	Analog comparator 1 positive input 0 pin.
	ACMP1_P1	PB.4	MFP1	A	Analog comparator 1 positive input 1 pin.
	ACMP1_P2	PB.12	MFP1	A	Analog comparator 1 positive input 2 pin.
	ACMP1_P3	PB.13	MFP1	A	Analog comparator 1 positive input 3 pin.
	ACMP1_WLAT	PA.6	MFP13	I	Analog comparator 1 window latch input pin
BPWM0	BPWM0_CH0	PA.11	MFP9	I/O	BPWM0 channel 0 output/capture input.
		PF.12	MFP11	I/O	
		PA.0	MFP12	I/O	
		PG.14	MFP12	I/O	
		PE.2	MFP13	I/O	

Group	Pin Name	GPIO	MFP	Type	Description
	BPWM0_CH1	PA.10	MFP9	I/O	BPWM0 channel 1 output/capture input.
		PF.13	MFP11	I/O	
		PA.1	MFP12	I/O	
		PG.13	MFP12	I/O	
		PE.3	MFP13	I/O	
	BPWM0_CH2	PA.9	MFP9	I/O	BPWM0 channel 2 output/capture input.
		PA.2	MFP12	I/O	
		PG.12	MFP12	I/O	
		PE.4	MFP13	I/O	
	BPWM0_CH3	PA.8	MFP9	I/O	BPWM0 channel 3 output/capture input.
		PA.3	MFP12	I/O	
		PG.11	MFP12	I/O	
		PE.5	MFP13	I/O	
	BPWM0_CH4	PC.13	MFP9	I/O	BPWM0 channel 4 output/capture input.
		PF.5	MFP8	I/O	
		PA.4	MFP12	I/O	
		PI.3	MFP12	I/O	
		PG.10	MFP12	I/O	
		PE.6	MFP13	I/O	
	BPWM0_CH5	PD.12	MFP9	I/O	BPWM0 channel 5 output/capture input.
		PF.4	MFP8	I/O	
		PA.5	MFP12	I/O	
		PI.2	MFP12	I/O	
		PG.9	MFP12	I/O	
		PE.7	MFP13	I/O	
BPWM1	BPWM1_CH0	PF.3	MFP11	I/O	BPWM1 channel 0 output/capture input.
		PC.7	MFP12	I/O	
		PF.0	MFP12	I/O	
		PB.11	MFP10	I/O	
	BPWM1_CH1	PF.2	MFP11	I/O	BPWM1 channel 1 output/capture input.
		PC.6	MFP12	I/O	
		PF.1	MFP12	I/O	

Group	Pin Name	GPIO	MFP	Type	Description
		PB.10	MFP10	I/O	
	BPWM1_CH2	PA.7	MFP12	I/O	BPWM1 channel 2 output/capture input.
		PA.12	MFP11	I/O	
		PB.9	MFP10	I/O	
	BPWM1_CH3	PA.6	MFP12	I/O	BPWM1 channel 3 output/capture input.
		PA.13	MFP11	I/O	
		PB.8	MFP10	I/O	
	BPWM1_CH4	PC.8	MFP12	I/O	BPWM1 channel 4 output/capture input.
		PA.14	MFP11	I/O	
		PB.7	MFP10	I/O	
	BPWM1_CH5	PE.13	MFP12	I/O	BPWM1 channel 5 output/capture input.
		PA.15	MFP11	I/O	
		PB.6	MFP10	I/O	
CIR0	CIR0_RXD	PF.12	MFP15	I	CIR0 data receiver input pin.
		PA.2	MFP15	I	
		PI.4	MFP14	I	
		PC.14	MFP10	I	
CLKO	CLKO	PC.13	MFP13	O	Clock Out
		PD.12	MFP13	O	
		PF.14	MFP13	O	
		PA.3	MFP11	O	
		PF.15	MFP14	O	
		PG.15	MFP14	O	
		PB.14	MFP14	O	
DAC0	DAC0_OUT	PB.12	MFP1	A	DAC0 channel analog output.
	DAC0_ST	PA.10	MFP14	I	DAC0 external trigger input.
		PA.0	MFP15	I	
EADC0	EADC0_CH0	PB.0	MFP1	A	EADC0 channel 0 analog input.
	EADC0_CH1	PB.1	MFP1	A	EADC0 channel 1 analog input.
	EADC0_CH2	PB.2	MFP1	A	EADC0 channel 2 analog input.
	EADC0_CH3	PB.3	MFP1	A	EADC0 channel 3 analog input.
	EADC0_CH4	PB.4	MFP1	A	EADC0 channel 4 analog input.

Group	Pin Name	GPIO	MFP	Type	Description
	EADC0_CH5	PB.5	MFP1	A	EADC0 channel 5 analog input.
	EADC0_CH6	PB.6	MFP1	A	EADC0 channel 6 analog input.
	EADC0_CH7	PB.7	MFP1	A	EADC0 channel 7 analog input.
	EADC0_CH8	PB.8	MFP1	A	EADC0 channel 8 analog input.
	EADC0_CH9	PB.9	MFP1	A	EADC0 channel 9 analog input.
	EADC0_CH10	PB.10	MFP1	A	EADC0 channel 10 analog input.
	EADC0_CH11	PB.11	MFP1	A	EADC0 channel 11 analog input.
	EADC0_CH12	PB.12	MFP1	A	EADC0 channel 12 analog input.
	EADC0_CH13	PB.13	MFP1	A	EADC0 channel 13 analog input.
	EADC0_CH14	PB.14	MFP1	A	EADC0 channel 14 analog input.
	EADC0_CH15	PB.15	MFP1	A	EADC0 channel 15 analog input.
	EADC0_CH16	PD.10	MFP1	A	EADC0 channel 16 analog input.
	EADC0_CH17	PD.11	MFP1	A	EADC0 channel 17 analog input.
	EADC0_CH18	PD.12	MFP1	A	EADC0 channel 18 analog input.
	EADC0_CH19	PC.13	MFP1	A	EADC0 channel 19 analog input.
	EADC0_CH20	PA.8	MFP1	A	EADC0 channel 20 analog input.
	EADC0_CH21	PA.9	MFP1	A	EADC0 channel 21 analog input.
	EADC0_CH22	PA.10	MFP1	A	EADC0 channel 22 analog input.
	EADC0_CH23	PA.11	MFP1	A	EADC0 channel 23 analog input.
	EADC0_ST	PC.13	MFP14	I	EADC0 external trigger input.
		PD.12	MFP14	I	
		PF.5	MFP11	I	
		PC.1	MFP15	I	
		PI.2	MFP15	I	
		PG.15	MFP15	I	
EPWM0	EPWM0_BRAKE0	PB.3	MFP13	I	EPWM0 Brake 0 input pin.
		PB.1	MFP13	I	
		PF.14	MFP10	I	
		PE.8	MFP11	I	
		PF.15	MFP11	I	
	EPWM0_BRAKE1	PB.0	MFP13	I	EPWM0 Brake 1 input pin.
		PE.9	MFP11	I	

Group	Pin Name	GPIO	MFP	Type	Description
		PB.15	MFP10	I	
	EPWM0_CH0	PB.5	MFP11	I/O	EPWM0 channel 0 output/capture input.
		PF.5	MFP7	I/O	
		PE.8	MFP10	I/O	
		PA.5	MFP13	I/O	
		PE.7	MFP12	I/O	
	EPWM0_CH1	PB.4	MFP11	I/O	EPWM0 channel 1 output/capture input.
		PF.4	MFP7	I/O	
		PE.9	MFP10	I/O	
		PA.4	MFP13	I/O	
		PF.15	MFP12	I/O	
		PE.6	MFP12	I/O	
	EPWM0_CH2	PB.3	MFP11	I/O	EPWM0 channel 2 output/capture input.
		PE.10	MFP10	I/O	
		PF.12	MFP10	I/O	
		PA.3	MFP13	I/O	
		PE.5	MFP12	I/O	
	EPWM0_CH3	PB.2	MFP11	I/O	EPWM0 channel 3 output/capture input.
		PE.11	MFP10	I/O	
		PF.13	MFP10	I/O	
		PA.2	MFP13	I/O	
		PE.4	MFP12	I/O	
	EPWM0_CH4	PB.1	MFP11	I/O	EPWM0 channel 4 output/capture input.
		PI.0	MFP11	I/O	
		PF.14	MFP12	I/O	
		PE.12	MFP10	I/O	
		PA.1	MFP13	I/O	
		PE.3	MFP12	I/O	
		PI.5	MFP11	I/O	
		PD.14	MFP11	I/O	
	EPWM0_CH5	PB.0	MFP11	I/O	EPWM0 channel 5 output/capture input.
		PI.1	MFP11	I/O	

Group	Pin Name	GPIO	MFP	Type	Description
		PE.13	MFP10	I/O	
		PD.15	MFP12	I/O	
		PA.0	MFP13	I/O	
		PE.2	MFP12	I/O	
		PI.4	MFP11	I/O	
		PH.11	MFP11	I/O	
	EPWM0_SYNC_IN	PA.15	MFP12	I	EPWM0 counter synchronous trigger input pin.
		PC.14	MFP11	I	
	EPWM0_SYNC_OUT	PA.11	MFP10	O	EPWM0 counter synchronous trigger output pin.
		PF.5	MFP9	O	
EPWM1	EPWM1_BRAKE0	PF.14	MFP9	I	EPWM1 Brake 0 input pin.
		PE.10	MFP11	I	
		PB.7	MFP11	I	
	EPWM1_BRAKE1	PE.11	MFP11	I	EPWM1 Brake 1 input pin.
		PA.3	MFP15	I	
		PB.6	MFP11	I	
	EPWM1_CH0	PC.12	MFP12	I/O	EPWM1 channel 0 output/capture input.
		PI.0	MFP12	I/O	
		PE.13	MFP11	I/O	
		PC.5	MFP12	I/O	
		PB.15	MFP11	I/O	
	EPWM1_CH1	PC.11	MFP12	I/O	EPWM1 channel 1 output/capture input.
		PI.1	MFP12	I/O	
		PC.8	MFP11	I/O	
		PC.4	MFP12	I/O	
		PB.14	MFP11	I/O	
	EPWM1_CH2	PC.10	MFP12	I/O	EPWM1 channel 2 output/capture input.
		PC.7	MFP11	I/O	
		PC.3	MFP12	I/O	
		PB.13	MFP11	I/O	
	EPWM1_CH3	PC.9	MFP12	I/O	EPWM1 channel 3 output/capture input.
		PC.6	MFP11	I/O	

Group	Pin Name	GPIO	MFP	Type	Description
		PC.2	MFP12	I/O	
		PB.12	MFP11	I/O	
	EPWM1_CH4	PB.1	MFP12	I/O	EPWM1 channel 4 output/capture input.
		PA.7	MFP11	I/O	
		PC.1	MFP12	I/O	
		PB.7	MFP12	I/O	
	EPWM1_CH5	PB.0	MFP12	I/O	EPWM1 channel 5 output/capture input.
		PA.6	MFP11	I/O	
		PC.0	MFP12	I/O	
		PB.6	MFP12	I/O	
I²C0	I2C0_SCL	PB.5	MFP6	I/O	I²C0 clock pin.
		PC.12	MFP4	I/O	
		PI.0	MFP4	I/O	
		PF.3	MFP4	I/O	
		PE.13	MFP4	I/O	
		PF.12	MFP4	I/O	
		PA.5	MFP9	I/O	
		PC.1	MFP9	I/O	
		PI.2	MFP4	I/O	
		PD.7	MFP4	I/O	
		PB.9	MFP9	I/O	
	I2C0_SDA	PB.4	MFP6	I/O	I²C0 data input/output pin.
		PC.11	MFP4	I/O	
		PI.1	MFP4	I/O	
		PF.2	MFP4	I/O	
		PC.8	MFP4	I/O	
		PF.13	MFP4	I/O	
		PA.4	MFP9	I/O	
		PC.0	MFP9	I/O	
		PI.3	MFP4	I/O	
		PD.6	MFP4	I/O	
		PB.8	MFP9	I/O	

Group	Pin Name	GPIO	MFP	Type	Description
	I2C0_SMBAL	PG.2	MFP4	O	I²C0 SMBus SMBALTER pin
		PA.3	MFP10	O	
		PC.3	MFP9	O	
	I2C0_SMBSUS	PG.3	MFP4	O	I²C0 SMBus SMBSUS pin (PMBus CONTROL pin)
		PA.2	MFP10	O	
		PC.2	MFP9	O	
I²C1	I2C1_SCL	PB.3	MFP12	I/O	I²C1 clock pin.
		PB.1	MFP9	I/O	
		PI.0	MFP9	I/O	
		PG.2	MFP5	I/O	
		PA.7	MFP8	I/O	
		PF.12	MFP9	I/O	
		PA.3	MFP9	I/O	
		PF.0	MFP3	I/O	
		PC.5	MFP9	I/O	
		PI.2	MFP9	I/O	
		PD.5	MFP4	I/O	
		PA.12	MFP4	I/O	
		PE.1	MFP8	I/O	
		PB.11	MFP7	I/O	
	I2C1_SDA	PB.2	MFP12	I/O	I²C1 data input/output pin.
		PB.0	MFP9	I/O	
		PI.1	MFP9	I/O	
		PG.3	MFP5	I/O	
		PA.6	MFP8	I/O	
		PF.13	MFP9	I/O	
		PA.2	MFP9	I/O	
		PF.1	MFP3	I/O	
		PC.4	MFP9	I/O	
		PI.3	MFP9	I/O	
		PD.4	MFP4	I/O	
		PA.13	MFP4	I/O	

Group	Pin Name	GPIO	MFP	Type	Description
		PE.0	MFP8	I/O	
		PB.10	MFP7	I/O	
	I2C1_SMBAL	PC.7	MFP8	O	I^2C1 SMBus SMBALTER pin
		PH.8	MFP8	O	
		PB.9	MFP7	O	
	I2C1_SMBSUS	PC.6	MFP8	O	I^2C1 SMBus SMBSUS pin (PMBus CONTROL pin)
		PH.9	MFP8	O	
		PB.8	MFP7	O	
ICE	ICE_CLK	PF.1	MFP14	I	Serial wired debugger clock pin. Note: It is recommended to use 100 kΩ pull-up resistor on ICE_CLK pin.
	ICE_DAT	PF.0	MFP14	I/O	Serial wired debugger data pin. Note: It is recommended to use 100 kΩ pull-up resistor on ICE_DAT pin.
INT0	INT0	PB.5	MFP15	I	External interrupt 0 input pin.
		PA.6	MFP15	I	
INT1	INT1	PB.4	MFP15	I	External interrupt 1 input pin.
		PA.7	MFP15	I	
		PD.15	MFP15	I	
INT2	INT2	PB.3	MFP15	I	External interrupt 2 input pin.
		PC.6	MFP15	I	
INT3	INT3	PB.2	MFP15	I	External interrupt 3 input pin.
		PC.7	MFP15	I	
INT4	INT4	PA.8	MFP15	I	External interrupt 4 input pin.
		PF.15	MFP15	I	
		PB.6	MFP13	I	
INT5	INT5	PD.12	MFP15	I	External interrupt 5 input pin.
		PF.14	MFP15	I	
		PB.7	MFP13	I	
INT6	INT6	PD.11	MFP15	I	External interrupt 6 input pin.
		PB.8	MFP13	I	
INT7	INT7	PD.10	MFP15	I	External interrupt 7 input pin.
		PB.9	MFP13	I	

Group	Pin Name	GPIO	MFP	Type	Description
SPI0	SPI0_CLK	PF.8	MFP5	I/O	SPI0 serial clock pin.
		PA.2	MFP4	I/O	
		PD.2	MFP4	I/O	
		PB.14	MFP4	I/O	
	SPI0_I2SMCLK	PB.0	MFP8	I/O	SPI0 I²S master clock output pin
		PF.10	MFP5	I/O	
		PA.4	MFP4	I/O	
		PD.13	MFP4	I/O	
		PD.14	MFP5	I/O	
		PC.14	MFP4	I/O	
		PB.11	MFP9	I/O	
	SPI0_MISO	PF.7	MFP5	I/O	SPI0 MISO (Master In, Slave Out) pin.
		PA.1	MFP4	I/O	
		PD.1	MFP4	I/O	
		PB.13	MFP4	I/O	
	SPI0_MOSI	PF.6	MFP5	I/O	SPI0 MOSI (Master Out, Slave In) pin.
		PA.0	MFP4	I/O	
		PD.0	MFP4	I/O	
		PB.12	MFP4	I/O	
	SPI0_SS	PF.9	MFP5	I/O	SPI0 slave select pin.
		PA.3	MFP4	I/O	
		PD.3	MFP4	I/O	
		PB.15	MFP4	I/O	
SPI1	SPI1_CLK	PB.3	MFP5	I/O	SPI1 serial clock pin.
		PH.6	MFP3	I/O	
		PA.7	MFP4	I/O	
		PC.1	MFP7	I/O	
		PD.5	MFP5	I/O	
		PH.8	MFP6	I/O	
	SPI1_I2SMCLK	PB.1	MFP5	I/O	SPI1 I²S master clock output pin
		PA.5	MFP4	I/O	
		PC.4	MFP7	I/O	

Group	Pin Name	GPIO	MFP	Type	Description
		PD.13	MFP5	I/O	
		PH.10	MFP6	I/O	
	SPI1_MISO	PB.5	MFP5	I/O	SPI1 MISO (Master In, Slave Out) pin.
		PH.4	MFP3	I/O	
		PC.7	MFP4	I/O	
		PC.3	MFP7	I/O	
		PD.7	MFP5	I/O	
		PE.1	MFP6	I/O	
	SPI1_MOSI	PB.4	MFP5	I/O	SPI1 MOSI (Master Out, Slave In) pin.
		PH.5	MFP3	I/O	
		PC.6	MFP4	I/O	
		PC.2	MFP7	I/O	
		PD.6	MFP5	I/O	
		PE.0	MFP6	I/O	
	SPI1_SS	PB.2	MFP5	I/O	SPI1 slave select pin.
		PH.7	MFP3	I/O	
		PA.6	MFP4	I/O	
		PC.0	MFP7	I/O	
		PD.4	MFP5	I/O	
		PH.9	MFP6	I/O	
TM0	TM0	PB.5	MFP14	I/O	Timer0 event counter input/toggle output pin.
		PG.2	MFP13	I/O	
		PC.7	MFP14	I/O	
	TM0_EXT	PA.11	MFP13	I/O	Timer0 external capture input/toggle output pin.
		PB.15	MFP13	I/O	
TM1	TM1	PB.4	MFP14	I/O	Timer1 event counter input/toggle output pin.
		PG.3	MFP13	I/O	
		PC.6	MFP14	I/O	
		PC.14	MFP13	I/O	
	TM1_EXT	PA.10	MFP13	I/O	Timer1 external capture input/toggle output pin.
		PB.14	MFP13	I/O	
TM2	TM2	PB.3	MFP14	I/O	Timer2 event counter input/toggle output

Group	Pin Name	GPIO	MFP	Type	Description
		PG.4	MFP13	I/O	pin.
		PA.7	MFP14	I/O	
		PF.15	MFP13	I/O	
		PD.0	MFP14	I/O	
	TM2_EXT	PA.9	MFP13	I/O	Timer2 external capture input/toggle output pin.
		PB.13	MFP13	I/O	
TM3	TM3	PB.2	MFP14	I/O	Timer3 event counter input/toggle output pin.
		PF.11	MFP13	I/O	
		PF.14	MFP14	I/O	
		PA.6	MFP14	I/O	
		PD.15	MFP14	I/O	
	TM3_EXT	PA.8	MFP13	I/O	Timer3 external capture input/toggle output pin.
		PB.12	MFP13	I/O	
TRACE	TRACE_CLK	PE.12	MFP14	O	ETM Trace Clock output pin
		PD.15	MFP13	O	
	TRACE_DATA0	PE.11	MFP14	O	ETM Trace Data 0 output pin
		PF.13	MFP13	O	
	TRACE_DATA1	PE.10	MFP14	O	ETM Trace Data 1 output pin
		PF.12	MFP13	O	
	TRACE_DATA2	PE.9	MFP14	O	ETM Trace Data 2 output pin
		PC.6	MFP13	O	
	TRACE_DATA3	PE.8	MFP14	O	ETM Trace Data 3 output pin
		PC.7	MFP13	O	
UART0	UART0_RXD	PC.11	MFP3	I	UART0 data receiver input pin.
		PI.1	MFP3	I	
		PF.2	MFP3	I	
		PA.6	MFP7	I	
		PA.4	MFP11	I	
		PA.0	MFP7	I	
		PF.1	MFP4	I	
		PD.2	MFP9	I	
		PA.15	MFP3	I	

Group	Pin Name	GPIO	MFP	Type	Description
		PI.5	MFP8	I	
		PH.11	MFP8	I	
		PB.12	MFP6	I	
		PB.8	MFP5	I	
	UART0_TXD	PC.12	MFP3	O	UART0 data transmitter output pin.
		PI.0	MFP3	O	
		PF.3	MFP3	O	
		PA.7	MFP7	O	
		PA.5	MFP11	O	
		PA.1	MFP7	O	
		PF.0	MFP4	O	
		PD.3	MFP9	O	
		PA.14	MFP3	O	
		PI.4	MFP8	O	
		PH.10	MFP8	O	
		PB.13	MFP6	O	
		PB.9	MFP5	O	
	UART0_nCTS	PC.7	MFP7	I	UART0 clear to Send input pin.
		PA.5	MFP7	I	
		PB.15	MFP6	I	
		PB.11	MFP5	I	
	UART0_nRTS	PC.6	MFP7	O	UART0 request to Send output pin.
		PA.4	MFP7	O	
		PB.14	MFP6	O	
		PB.10	MFP5	O	
UART1	UART1_RXD	PB.2	MFP6	I	UART1 data receiver input pin.
		PA.8	MFP7	I	
		PD.10	MFP3	I	
		PC.8	MFP8	I	
		PF.13	MFP8	I	
		PA.2	MFP8	I	
		PF.1	MFP2	I	

Group	Pin Name	GPIO	MFP	Type	Description
		PI.3	MFP3	I	
		PD.6	MFP3	I	
		PI.5	MFP10	I	
		PH.9	MFP10	I	
		PB.6	MFP6	I	
	UART1_TXD	PB.3	MFP6	O	UART1 data transmitter output pin.
		PA.9	MFP7	O	
		PD.11	MFP3	O	
		PE.13	MFP8	O	
		PF.12	MFP8	O	
		PA.3	MFP8	O	
		PF.0	MFP2	O	
		PI.2	MFP3	O	
		PD.7	MFP3	O	
		PI.4	MFP10	O	
		PH.8	MFP10	O	
		PB.7	MFP6	O	
	UART1_nCTS	PE.11	MFP8	I	UART1 clear to Send input pin.
		PA.1	MFP8	I	
		PB.9	MFP6	I	
	UART1_nRTS	PE.12	MFP8	O	UART1 request to Send output pin.
		PA.0	MFP8	O	
		PB.8	MFP6	O	
UART2	UART2_RXD	PB.4	MFP12	I	UART2 data receiver input pin.
		PB.0	MFP7	I	
		PI.1	MFP7	I	
		PD.12	MFP7	I	
		PF.5	MFP2	I	
		PE.9	MFP7	I	
		PE.15	MFP3	I	
		PC.4	MFP8	I	
		PC.0	MFP8	I	

Group	Pin Name	GPIO	MFP	Type	Description
	UART2_TXD	PB.5	MFP12	O	UART2 data transmitter output pin.
		PB.1	MFP7	O	
		PI.0	MFP7	O	
		PC.13	MFP7	O	
		PF.4	MFP2	O	
		PE.8	MFP7	O	
		PE.14	MFP3	O	
		PC.5	MFP8	O	
		PC.1	MFP8	O	
	UART2_nCTS	PF.5	MFP4	I	UART2 clear to Send input pin.
		PD.9	MFP4	I	
		PC.2	MFP8	I	
	UART2_nRTS	PF.4	MFP4	O	UART2 request to Send output pin.
		PD.8	MFP4	O	
		PC.3	MFP8	O	
UART3	UART3_RXD	PC.9	MFP7	I	UART3 data receiver input pin.
		PE.11	MFP7	I	
		PF.13	MFP7	I	
		PC.2	MFP11	I	
		PD.0	MFP5	I	
		PE.0	MFP7	I	
		PB.14	MFP7	I	
	UART3_TXD	PC.10	MFP7	O	UART3 data transmitter output pin.
		PE.10	MFP7	O	
		PF.12	MFP7	O	
		PC.3	MFP11	O	
		PD.1	MFP5	O	
		PE.1	MFP7	O	
		PB.15	MFP7	O	
	UART3_nCTS	PD.2	MFP5	I	UART3 clear to Send input pin.
		PH.9	MFP7	I	
		PB.12	MFP7	I	

Group	Pin Name	GPIO	MFP	Type	Description
	UART3_nRTS	PD.3	MFP5	O	UART3 request to Send output pin.
		PH.8	MFP7	O	
		PB.13	MFP7	O	
UART4	UART4_RXD	PF.6	MFP6	I	UART4 data receiver input pin.
		PC.6	MFP5	I	
		PA.2	MFP7	I	
		PC.4	MFP11	I	
		PI.3	MFP11	I	
		PA.13	MFP3	I	
		PI.5	MFP7	I	
		PH.11	MFP7	I	
		PB.10	MFP6	I	
	UART4_TXD	PF.7	MFP6	O	UART4 data transmitter output pin.
		PC.7	MFP5	O	
		PA.3	MFP7	O	
		PC.5	MFP11	O	
		PI.2	MFP11	O	
		PA.12	MFP3	O	
		PI.4	MFP7	O	
		PH.10	MFP7	O	
		PB.11	MFP6	O	
	UART4_nCTS	PC.8	MFP5	I	UART4 clear to Send input pin.
		PE.1	MFP9	I	
	UART4_nRTS	PE.13	MFP5	O	UART4 request to Send output pin.
		PE.0	MFP9	O	
UART5	UART5_RXD	PB.4	MFP7	I	UART5 data receiver input pin.
		PI.1	MFP6	I	
		PF.10	MFP6	I	
		PF.13	MFP6	I	
		PA.4	MFP8	I	
		PI.3	MFP6	I	
		PE.6	MFP8	I	

Group	Pin Name	GPIO	MFP	Type	Description
	UART5_TXD	PB.5	MFP7	O	UART5 data transmitter output pin.
		PI.0	MFP6	O	
		PF.11	MFP6	O	
		PF.12	MFP6	O	
		PA.5	MFP8	O	
		PI.2	MFP6	O	
		PE.7	MFP8	O	
	UART5_nCTS	PB.2	MFP7	I	UART5 clear to Send input pin.
		PF.8	MFP6	I	
	UART5_nRTS	PB.3	MFP7	O	UART5 request to Send output pin.
		PF.9	MFP6	O	
X32	X32_IN	PF.5	MFP10	I	External 32.768 kHz crystal input pin.
	X32_OUT	PF.4	MFP10	O	External 32.768 kHz crystal output pin.
XT1	XT1_IN	PF.3	MFP10	I	External 4~24 MHz (high speed) crystal input pin.
	XT1_OUT	PF.2	MFP10	O	External 4~24 MHz (high speed) crystal output pin.

4.3.2 Multi-function Summary Table Sorted by GPIO

	Pin Name	Type	MFP	Description
PA.0	PA.0	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MOSI	I/O	MFP4	SPI0 MOSI (Master Out, Slave In) pin.
	UART0_RXD	I	MFP7	UART0 data receiver input pin.
	UART1_nRTS	O	MFP8	UART1 request to Send output pin.
	BPWM0_CH0	I/O	MFP12	BPWM0 channel 0 output/capture input.
	EPWM0_CH5	I/O	MFP13	EPWM0 channel 5 output/capture input.
	DAC0_ST	I	MFP15	DAC0 external trigger input.
PA.1	PA.1	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MISO	I/O	MFP4	SPI0 MISO (Master In, Slave Out) pin.
	UART0_TXD	O	MFP7	UART0 data transmitter output pin.
	UART1_nCTS	I	MFP8	UART1 clear to Send input pin.
	BPWM0_CH1	I/O	MFP12	BPWM0 channel 1 output/capture input.
	EPWM0_CH4	I/O	MFP13	EPWM0 channel 4 output/capture input.
PA.2	PA.2	I/O	MFP0	General purpose digital I/O pin.
	SPI0_CLK	I/O	MFP4	SPI0 serial clock pin.
	UART4_RXD	I	MFP7	UART4 data receiver input pin.
	UART1_RXD	I	MFP8	UART1 data receiver input pin.
	I2C1_SDA	I/O	MFP9	I^2C1 data input/output pin.
	I2C0_SMBSUS	O	MFP10	I^2C0 SMBus SMBSUS pin (PMBus CONTROL pin)
	BPWM0_CH2	I/O	MFP12	BPWM0 channel 2 output/capture input.
	EPWM0_CH3	I/O	MFP13	EPWM0 channel 3 output/capture input.
	CIR0_RXD	I	MFP15	CIR0 data receiver input pin.
PA.3	PA.3	I/O	MFP0	General purpose digital I/O pin.
	SPI0_SS	I/O	MFP4	SPI0 slave select pin.
	UART4_TXD	O	MFP7	UART4 data transmitter output pin.
	UART1_TXD	O	MFP8	UART1 data transmitter output pin.
	I2C1_SCL	I/O	MFP9	I^2C1 clock pin.
	I2C0_SMBAL	O	MFP10	I^2C0 SMBus SMBALTER pin
	CLKO	O	MFP11	Clock Out
	BPWM0_CH3	I/O	MFP12	BPWM0 channel 3 output/capture input.
	EPWM0_CH2	I/O	MFP13	EPWM0 channel 2 output/capture input.

	Pin Name	Type	MFP	Description
	EPWM1_BRAKE1	I	MFP15	EPWM1 Brake 1 input pin.
PA.4	PA.4	I/O	MFP0	General purpose digital I/O pin.
	SPI0_I2SMCLK	I/O	MFP4	SPI0 I²S master clock output pin
	UART0_nRTS	O	MFP7	UART0 request to Send output pin.
	UART5_RXD	I	MFP8	UART5 data receiver input pin.
	I2C0_SDA	I/O	MFP9	I²C0 data input/output pin.
	UART0_RXD	I	MFP11	UART0 data receiver input pin.
	BPWM0_CH4	I/O	MFP12	BPWM0 channel 4 output/capture input.
	EPWM0_CH1	I/O	MFP13	EPWM0 channel 1 output/capture input.
PA.5	PA.5	I/O	MFP0	General purpose digital I/O pin.
	SPI1_I2SMCLK	I/O	MFP4	SPI1 I²S master clock output pin
	UART0_nCTS	I	MFP7	UART0 clear to Send input pin.
	UART5_TXD	O	MFP8	UART5 data transmitter output pin.
	I2C0_SCL	I/O	MFP9	I²C0 clock pin.
	UART0_TXD	O	MFP11	UART0 data transmitter output pin.
	BPWM0_CH5	I/O	MFP12	BPWM0 channel 5 output/capture input.
	EPWM0_CH0	I/O	MFP13	EPWM0 channel 0 output/capture input.
PA.6	PA.6	I/O	MFP0	General purpose digital I/O pin.
	SPI1_SS	I/O	MFP4	SPI1 slave select pin.
	UART0_RXD	I	MFP7	UART0 data receiver input pin.
	I2C1_SDA	I/O	MFP8	I²C1 data input/output pin.
	EPWM1_CH5	I/O	MFP11	EPWM1 channel 5 output/capture input.
	BPWM1_CH3	I/O	MFP12	BPWM1 channel 3 output/capture input.
	ACMP1_WLAT	I	MFP13	Analog comparator 1 window latch input pin
	TM3	I/O	MFP14	Timer3 event counter input/toggle output pin.
	INT0	I	MFP15	External interrupt 0 input pin.
PA.7	PA.7	I/O	MFP0	General purpose digital I/O pin.
	SPI1_CLK	I/O	MFP4	SPI1 serial clock pin.
	UART0_TXD	O	MFP7	UART0 data transmitter output pin.
	I2C1_SCL	I/O	MFP8	I²C1 clock pin.
	EPWM1_CH4	I/O	MFP11	EPWM1 channel 4 output/capture input.
	BPWM1_CH2	I/O	MFP12	BPWM1 channel 2 output/capture input.

	Pin Name	Type	MFP	Description
	ACMP0_WLAT	I	MFP13	Analog comparator 0 window latch input pin
	TM2	I/O	MFP14	Timer2 event counter input/toggle output pin.
	INT1	I	MFP15	External interrupt 1 input pin.
PA.8	PA.8	I/O	MFP0	General purpose digital I/O pin.
	EADC0_CH20	A	MFP1	EADC0 channel 20 analog input.
	UART1_RXD	I	MFP7	UART1 data receiver input pin.
	BPWM0_CH3	I/O	MFP9	BPWM0 channel 3 output/capture input.
	TM3_EXT	I/O	MFP13	Timer3 external capture input/toggle output pin.
	INT4	I	MFP15	External interrupt 4 input pin.
PA.9	PA.9	I/O	MFP0	General purpose digital I/O pin.
	EADC0_CH21	A	MFP1	EADC0 channel 21 analog input.
	UART1_TXD	O	MFP7	UART1 data transmitter output pin.
	BPWM0_CH2	I/O	MFP9	BPWM0 channel 2 output/capture input.
	TM2_EXT	I/O	MFP13	Timer2 external capture input/toggle output pin.
PA.10	PA.10	I/O	MFP0	General purpose digital I/O pin.
	EADC0_CH22	A	MFP1	EADC0 channel 22 analog input.
	ACMP1_P0	A	MFP1	Analog comparator 1 positive input 0 pin.
	BPWM0_CH1	I/O	MFP9	BPWM0 channel 1 output/capture input.
	TM1_EXT	I/O	MFP13	Timer1 external capture input/toggle output pin.
	DAC0_ST	I	MFP14	DAC0 external trigger input.
PA.11	PA.11	I/O	MFP0	General purpose digital I/O pin.
	EADC0_CH23	A	MFP1	EADC0 channel 23 analog input.
	ACMP0_P0	A	MFP1	Analog comparator 0 positive input 0 pin.
	BPWM0_CH0	I/O	MFP9	BPWM0 channel 0 output/capture input.
	EPWM0_SYNC_OUT	O	MFP10	EPWM0 counter synchronous trigger output pin.
	TM0_EXT	I/O	MFP13	Timer0 external capture input/toggle output pin.
PA.12	PA.12	I/O	MFP0	General purpose digital I/O pin.
	UART4_TXD	O	MFP3	UART4 data transmitter output pin.
	I2C1_SCL	I/O	MFP4	I^2C1 clock pin.
	BPWM1_CH2	I/O	MFP11	BPWM1 channel 2 output/capture input.
PA.13	PA.13	I/O	MFP0	General purpose digital I/O pin.
	UART4_RXD	I	MFP3	UART4 data receiver input pin.

	Pin Name	Type	MFP	Description
	I2C1_SDA	I/O	MFP4	I²C1 data input/output pin.
	BPWM1_CH3	I/O	MFP11	BPWM1 channel 3 output/capture input.
PA.14	PA.14	I/O	MFP0	General purpose digital I/O pin.
	UART0_TXD	O	MFP3	UART0 data transmitter output pin.
	BPWM1_CH4	I/O	MFP11	BPWM1 channel 4 output/capture input.
PA.15	PA.15	I/O	MFP0	General purpose digital I/O pin.
	UART0_RXD	I	MFP3	UART0 data receiver input pin.
	BPWM1_CH5	I/O	MFP11	BPWM1 channel 5 output/capture input.
	EPWM0_SYNC_IN	I	MFP12	EPWM0 counter synchronous trigger input pin.
PB.0	PB.0	I/O	MFP0	General purpose digital I/O pin.
	EADC0_CH0	A	MFP1	EADC0 channel 0 analog input.
	UART2_RXD	I	MFP7	UART2 data receiver input pin.
	SPI0_I2SMCLK	I/O	MFP8	SPI0 I²S master clock output pin
	I2C1_SDA	I/O	MFP9	I²C1 data input/output pin.
	EPWM0_CH5	I/O	MFP11	EPWM0 channel 5 output/capture input.
	EPWM1_CH5	I/O	MFP12	EPWM1 channel 5 output/capture input.
	EPWM0_BRAKE1	I	MFP13	EPWM0 Brake 1 input pin.
PB.1	PB.1	I/O	MFP0	General purpose digital I/O pin.
	EADC0_CH1	A	MFP1	EADC0 channel 1 analog input.
	SPI1_I2SMCLK	I/O	MFP5	SPI1 I²S master clock output pin
	UART2_TXD	O	MFP7	UART2 data transmitter output pin.
	I2C1_SCL	I/O	MFP9	I²C1 clock pin.
	EPWM0_CH4	I/O	MFP11	EPWM0 channel 4 output/capture input.
	EPWM1_CH4	I/O	MFP12	EPWM1 channel 4 output/capture input.
	EPWM0_BRAKE0	I	MFP13	EPWM0 Brake 0 input pin.
PB.2	PB.2	I/O	MFP0	General purpose digital I/O pin.
	EADC0_CH2	A	MFP1	EADC0 channel 2 analog input.
	ACMP0_P1	A	MFP1	Analog comparator 0 positive input 1 pin.
	SPI1_SS	I/O	MFP5	SPI1 slave select pin.
	UART1_RXD	I	MFP6	UART1 data receiver input pin.
	UART5_nCTS	I	MFP7	UART5 clear to Send input pin.
	EPWM0_CH3	I/O	MFP11	EPWM0 channel 3 output/capture input.

	Pin Name	Type	MFP	Description
	I2C1_SDA	I/O	MFP12	I^2C1 data input/output pin.
	TM3	I/O	MFP14	Timer3 event counter input/toggle output pin.
	INT3	I	MFP15	External interrupt 3 input pin.
PB.3	PB.3	I/O	MFP0	General purpose digital I/O pin.
	EADC0_CH3	A	MFP1	EADC0 channel 3 analog input.
	ACMP0_N	A	MFP1	Analog comparator 0 negative input pin.
	SPI1_CLK	I/O	MFP5	SPI1 serial clock pin.
	UART1_TXD	O	MFP6	UART1 data transmitter output pin.
	UART5_nRTS	O	MFP7	UART5 request to Send output pin.
	EPWM0_CH2	I/O	MFP11	EPWM0 channel 2 output/capture input.
	I2C1_SCL	I/O	MFP12	I^2C1 clock pin.
	EPWM0_BRAKE0	I	MFP13	EPWM0 Brake 0 input pin.
	TM2	I/O	MFP14	Timer2 event counter input/toggle output pin.
	INT2	I	MFP15	External interrupt 2 input pin.
PB.4	PB.4	I/O	MFP0	General purpose digital I/O pin.
	EADC0_CH4	A	MFP1	EADC0 channel 4 analog input.
	ACMP1_P1	A	MFP1	Analog comparator 1 positive input 1 pin.
	SPI1_MOSI	I/O	MFP5	SPI1 MOSI (Master Out, Slave In) pin.
	I2C0_SDA	I/O	MFP6	I^2C0 data input/output pin.
	UART5_RXD	I	MFP7	UART5 data receiver input pin.
	EPWM0_CH1	I/O	MFP11	EPWM0 channel 1 output/capture input.
	UART2_RXD	I	MFP12	UART2 data receiver input pin.
	TM1	I/O	MFP14	Timer1 event counter input/toggle output pin.
	INT1	I	MFP15	External interrupt 1 input pin.
PB.5	PB.5	I/O	MFP0	General purpose digital I/O pin.
	EADC0_CH5	A	MFP1	EADC0 channel 5 analog input.
	ACMP1_N	A	MFP1	Analog comparator 1 negative input pin.
	SPI1_MISO	I/O	MFP5	SPI1 MISO (Master In, Slave Out) pin.
	I2C0_SCL	I/O	MFP6	I^2C0 clock pin.
	UART5_TXD	O	MFP7	UART5 data transmitter output pin.
	EPWM0_CH0	I/O	MFP11	EPWM0 channel 0 output/capture input.
	UART2_TXD	O	MFP12	UART2 data transmitter output pin.

	Pin Name	Type	MFP	Description
	TM0	I/O	MFP14	Timer0 event counter input/toggle output pin.
	INT0	I	MFP15	External interrupt 0 input pin.
PB.6	PB.6	I/O	MFP0	General purpose digital I/O pin.
	EADC0_CH6	A	MFP1	EADC0 channel 6 analog input.
	UART1_RXD	I	MFP6	UART1 data receiver input pin.
	BPWM1_CH5	I/O	MFP10	BPWM1 channel 5 output/capture input.
	EPWM1_BRAKE1	I	MFP11	EPWM1 Brake 1 input pin.
	EPWM1_CH5	I/O	MFP12	EPWM1 channel 5 output/capture input.
	INT4	I	MFP13	External interrupt 4 input pin.
	ACMP1_O	O	MFP15	Analog comparator 1 output pin.
PB.7	PB.7	I/O	MFP0	General purpose digital I/O pin.
	EADC0_CH7	A	MFP1	EADC0 channel 7 analog input.
	UART1_TXD	O	MFP6	UART1 data transmitter output pin.
	BPWM1_CH4	I/O	MFP10	BPWM1 channel 4 output/capture input.
	EPWM1_BRAKE0	I	MFP11	EPWM1 Brake 0 input pin.
	EPWM1_CH4	I/O	MFP12	EPWM1 channel 4 output/capture input.
	INT5	I	MFP13	External interrupt 5 input pin.
	ACMP0_O	O	MFP15	Analog comparator 0 output pin.
PB.8	PB.8	I/O	MFP0	General purpose digital I/O pin.
	EADC0_CH8	A	MFP1	EADC0 channel 8 analog input.
	UART0_RXD	I	MFP5	UART0 data receiver input pin.
	UART1_nRTS	O	MFP6	UART1 request to Send output pin.
	I2C1_SMBSUS	O	MFP7	I^2C1 SMBus SMBSUS pin (PMBus CONTROL pin)
	I2C0_SDA	I/O	MFP9	I^2C0 data input/output pin.
	BPWM1_CH3	I/O	MFP10	BPWM1 channel 3 output/capture input.
	INT6	I	MFP13	External interrupt 6 input pin.
PB.9	PB.9	I/O	MFP0	General purpose digital I/O pin.
	EADC0_CH9	A	MFP1	EADC0 channel 9 analog input.
	UART0_TXD	O	MFP5	UART0 data transmitter output pin.
	UART1_nCTS	I	MFP6	UART1 clear to Send input pin.
	I2C1_SMBAL	O	MFP7	I^2C1 SMBus SMBALTER pin
	I2C0_SCL	I/O	MFP9	I^2C0 clock pin.

	Pin Name	Type	MFP	Description
	BPWM1_CH2	I/O	MFP10	BPWM1 channel 2 output/capture input.
	INT7	I	MFP13	External interrupt 7 input pin.
PB.10	PB.10	I/O	MFP0	General purpose digital I/O pin.
	EADC0_CH10	A	MFP1	EADC0 channel 10 analog input.
	UART0_nRTS	O	MFP5	UART0 request to Send output pin.
	UART4_RXD	I	MFP6	UART4 data receiver input pin.
	I2C1_SDA	I/O	MFP7	I^2C1 data input/output pin.
	BPWM1_CH1	I/O	MFP10	BPWM1 channel 1 output/capture input.
PB.11	PB.11	I/O	MFP0	General purpose digital I/O pin.
	EADC0_CH11	A	MFP1	EADC0 channel 11 analog input.
	UART0_nCTS	I	MFP5	UART0 clear to Send input pin.
	UART4_TXD	O	MFP6	UART4 data transmitter output pin.
	I2C1_SCL	I/O	MFP7	I^2C1 clock pin.
	SPI0_I2SMCLK	I/O	MFP9	SPI0 I^2S master clock output pin
	BPWM1_CH0	I/O	MFP10	BPWM1 channel 0 output/capture input.
PB.12	PB.12	I/O	MFP0	General purpose digital I/O pin.
	EADC0_CH12	A	MFP1	EADC0 channel 12 analog input.
	DAC0_OUT	A	MFP1	DAC0 channel analog output.
	ACMP0_P2	A	MFP1	Analog comparator 0 positive input 2 pin.
	ACMP1_P2	A	MFP1	Analog comparator 1 positive input 2 pin.
	SPI0_MOSI	I/O	MFP4	SPI0 MOSI (Master Out, Slave In) pin.
	UART0_RXD	I	MFP6	UART0 data receiver input pin.
	UART3_nCTS	I	MFP7	UART3 clear to Send input pin.
	EPWM1_CH3	I/O	MFP11	EPWM1 channel 3 output/capture input.
	TM3_EXT	I/O	MFP13	Timer3 external capture input/toggle output pin.
PB.13	PB.13	I/O	MFP0	General purpose digital I/O pin.
	EADC0_CH13	A	MFP1	EADC0 channel 13 analog input.
	ACMP0_P3	A	MFP1	Analog comparator 0 positive input 3 pin.
	ACMP1_P3	A	MFP1	Analog comparator 1 positive input 3 pin.
	SPI0_MISO	I/O	MFP4	SPI0 MISO (Master In, Slave Out) pin.
	UART0_TXD	O	MFP6	UART0 data transmitter output pin.
	UART3_nRTS	O	MFP7	UART3 request to Send output pin.

	Pin Name	Type	MFP	Description
	EPWM1_CH2	I/O	MFP11	EPWM1 channel 2 output/capture input.
	TM2_EXT	I/O	MFP13	Timer2 external capture input/toggle output pin.
PB.14	PB.14	I/O	MFP0	General purpose digital I/O pin.
	EADC0_CH14	A	MFP1	EADC0 channel 14 analog input.
	SPI0_CLK	I/O	MFP4	SPI0 serial clock pin.
	UART0_nRTS	O	MFP6	UART0 request to Send output pin.
	UART3_RXD	I	MFP7	UART3 data receiver input pin.
	EPWM1_CH1	I/O	MFP11	EPWM1 channel 1 output/capture input.
	TM1_EXT	I/O	MFP13	Timer1 external capture input/toggle output pin.
	CLKO	O	MFP14	Clock Out
PB.15	PB.15	I/O	MFP0	General purpose digital I/O pin.
	EADC0_CH15	A	MFP1	EADC0 channel 15 analog input.
	SPI0_SS	I/O	MFP4	SPI0 slave select pin.
	UART0_nCTS	I	MFP6	UART0 clear to Send input pin.
	UART3_TXD	O	MFP7	UART3 data transmitter output pin.
	EPWM0_BRAKE1	I	MFP10	EPWM0 Brake 1 input pin.
	EPWM1_CH0	I/O	MFP11	EPWM1 channel 0 output/capture input.
	TM0_EXT	I/O	MFP13	Timer0 external capture input/toggle output pin.
PC.0	PC.0	I/O	MFP0	General purpose digital I/O pin.
	SPI1_SS	I/O	MFP7	SPI1 slave select pin.
	UART2_RXD	I	MFP8	UART2 data receiver input pin.
	I2C0_SDA	I/O	MFP9	I^2C0 data input/output pin.
	EPWM1_CH5	I/O	MFP12	EPWM1 channel 5 output/capture input.
	ACMP1_O	O	MFP14	Analog comparator 1 output pin.
PC.1	PC.1	I/O	MFP0	General purpose digital I/O pin.
	SPI1_CLK	I/O	MFP7	SPI1 serial clock pin.
	UART2_TXD	O	MFP8	UART2 data transmitter output pin.
	I2C0_SCL	I/O	MFP9	I^2C0 clock pin.
	EPWM1_CH4	I/O	MFP12	EPWM1 channel 4 output/capture input.
	ACMP0_O	O	MFP14	Analog comparator 0 output pin.
	EADC0_ST	I	MFP15	EADC0 external trigger input.
PC.2	PC.2	I/O	MFP0	General purpose digital I/O pin.

	Pin Name	Type	MFP	Description
	SPI1_MOSI	I/O	MFP7	SPI1 MOSI (Master Out, Slave In) pin.
	UART2_nCTS	I	MFP8	UART2 clear to Send input pin.
	I2C0_SMBSUS	O	MFP9	I^2C0 SMBus SMBSUS pin (PMBus CONTROL pin)
	UART3_RXD	I	MFP11	UART3 data receiver input pin.
	EPWM1_CH3	I/O	MFP12	EPWM1 channel 3 output/capture input.
PC.3	PC.3	I/O	MFP0	General purpose digital I/O pin.
	SPI1_MISO	I/O	MFP7	SPI1 MISO (Master In, Slave Out) pin.
	UART2_nRTS	O	MFP8	UART2 request to Send output pin.
	I2C0_SMBAL	O	MFP9	I^2C0 SMBus SMBALTER pin
	UART3_TXD	O	MFP11	UART3 data transmitter output pin.
	EPWM1_CH2	I/O	MFP12	EPWM1 channel 2 output/capture input.
PC.4	PC.4	I/O	MFP0	General purpose digital I/O pin.
	SPI1_I2SMCLK	I/O	MFP7	SPI1 I^2S master clock output pin
	UART2_RXD	I	MFP8	UART2 data receiver input pin.
	I2C1_SDA	I/O	MFP9	I^2C1 data input/output pin.
	UART4_RXD	I	MFP11	UART4 data receiver input pin.
	EPWM1_CH1	I/O	MFP12	EPWM1 channel 1 output/capture input.
PC.5	PC.5	I/O	MFP0	General purpose digital I/O pin.
	UART2_TXD	O	MFP8	UART2 data transmitter output pin.
	I2C1_SCL	I/O	MFP9	I^2C1 clock pin.
	UART4_TXD	O	MFP11	UART4 data transmitter output pin.
	EPWM1_CH0	I/O	MFP12	EPWM1 channel 0 output/capture input.
PC.6	PC.6	I/O	MFP0	General purpose digital I/O pin.
	SPI1_MOSI	I/O	MFP4	SPI1 MOSI (Master Out, Slave In) pin.
	UART4_RXD	I	MFP5	UART4 data receiver input pin.
	UART0_nRTS	O	MFP7	UART0 request to Send output pin.
	I2C1_SMBSUS	O	MFP8	I^2C1 SMBus SMBSUS pin (PMBus CONTROL pin)
	EPWM1_CH3	I/O	MFP11	EPWM1 channel 3 output/capture input.
	BPWM1_CH1	I/O	MFP12	BPWM1 channel 1 output/capture input.
	TRACE_DATA2	O	MFP13	ETM Trace Data 2 output pin
	TM1	I/O	MFP14	Timer1 event counter input/toggle output pin.
	INT2	I	MFP15	External interrupt 2 input pin.

	Pin Name	Type	MFP	Description
PC.7	PC.7	I/O	MFP0	General purpose digital I/O pin.
	SPI1_MISO	I/O	MFP4	SPI1 MISO (Master In, Slave Out) pin.
	UART4_TXD	O	MFP5	UART4 data transmitter output pin.
	UART0_nCTS	I	MFP7	UART0 clear to Send input pin.
	I2C1_SMBAL	O	MFP8	I^2C1 SMBus SMBALTER pin
	EPWM1_CH2	I/O	MFP11	EPWM1 channel 2 output/capture input.
	BPWM1_CH0	I/O	MFP12	BPWM1 channel 0 output/capture input.
	TRACE_DATA3	O	MFP13	ETM Trace Data 3 output pin
	TM0	I/O	MFP14	Timer0 event counter input/toggle output pin.
	INT3	I	MFP15	External interrupt 3 input pin.
PC.8	PC.8	I/O	MFP0	General purpose digital I/O pin.
	I2C0_SDA	I/O	MFP4	I^2C0 data input/output pin.
	UART4_nCTS	I	MFP5	UART4 clear to Send input pin.
	UART1_RXD	I	MFP8	UART1 data receiver input pin.
	EPWM1_CH1	I/O	MFP11	EPWM1 channel 1 output/capture input.
	BPWM1_CH4	I/O	MFP12	BPWM1 channel 4 output/capture input.
PC.9	PC.9	I/O	MFP0	General purpose digital I/O pin.
	UART3_RXD	I	MFP7	UART3 data receiver input pin.
	EPWM1_CH3	I/O	MFP12	EPWM1 channel 3 output/capture input.
PC.10	PC.10	I/O	MFP0	General purpose digital I/O pin.
	UART3_TXD	O	MFP7	UART3 data transmitter output pin.
	EPWM1_CH2	I/O	MFP12	EPWM1 channel 2 output/capture input.
PC.11	PC.11	I/O	MFP0	General purpose digital I/O pin.
	UART0_RXD	I	MFP3	UART0 data receiver input pin.
	I2C0_SDA	I/O	MFP4	I^2C0 data input/output pin.
	EPWM1_CH1	I/O	MFP12	EPWM1 channel 1 output/capture input.
	ACMP1_O	O	MFP14	Analog comparator 1 output pin.
PC.12	PC.12	I/O	MFP0	General purpose digital I/O pin.
	UART0_TXD	O	MFP3	UART0 data transmitter output pin.
	I2C0_SCL	I/O	MFP4	I^2C0 clock pin.
	EPWM1_CH0	I/O	MFP12	EPWM1 channel 0 output/capture input.
	ACMP0_O	O	MFP14	Analog comparator 0 output pin.

	Pin Name	Type	MFP	Description
PC.13	PC.13	I/O	MFP0	General purpose digital I/O pin.
	EADC0_CH19	A	MFP1	EADC0 channel 19 analog input.
	UART2_TXD	O	MFP7	UART2 data transmitter output pin.
	BPWM0_CH4	I/O	MFP9	BPWM0 channel 4 output/capture input.
	CLKO	O	MFP13	Clock Out
	EADC0_ST	I	MFP14	EADC0 external trigger input.
PC.14	PC.14	I/O	MFP0	General purpose digital I/O pin.
	SPI0_I2SMCLK	I/O	MFP4	SPI0 I^2S master clock output pin
	CIR0_RXD	I	MFP10	CIR0 data receiver input pin.
	EPWM0_SYNC_IN	I	MFP11	EPWM0 counter synchronous trigger input pin.
	TM1	I/O	MFP13	Timer1 event counter input/toggle output pin.
PD.0	PD.0	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MOSI	I/O	MFP4	SPI0 MOSI (Master Out, Slave In) pin.
	UART3_RXD	I	MFP5	UART3 data receiver input pin.
	TM2	I/O	MFP14	Timer2 event counter input/toggle output pin.
PD.1	PD.1	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MISO	I/O	MFP4	SPI0 MISO (Master In, Slave Out) pin.
	UART3_TXD	O	MFP5	UART3 data transmitter output pin.
PD.2	PD.2	I/O	MFP0	General purpose digital I/O pin.
	SPI0_CLK	I/O	MFP4	SPI0 serial clock pin.
	UART3_nCTS	I	MFP5	UART3 clear to Send input pin.
	UART0_RXD	I	MFP9	UART0 data receiver input pin.
PD.3	PD.3	I/O	MFP0	General purpose digital I/O pin.
	SPI0_SS	I/O	MFP4	SPI0 slave select pin.
	UART3_nRTS	O	MFP5	UART3 request to Send output pin.
	UART0_TXD	O	MFP9	UART0 data transmitter output pin.
PD.4	PD.4	I/O	MFP0	General purpose digital I/O pin.
	I2C1_SDA	I/O	MFP4	I^2C1 data input/output pin.
	SPI1_SS	I/O	MFP5	SPI1 slave select pin.
PD.5	PD.5	I/O	MFP0	General purpose digital I/O pin.
	I2C1_SCL	I/O	MFP4	I^2C1 clock pin.
	SPI1_CLK	I/O	MFP5	SPI1 serial clock pin.

	Pin Name	Type	MFP	Description
PD.6	PD.6	I/O	MFP0	General purpose digital I/O pin.
	UART1_RXD	I	MFP3	UART1 data receiver input pin.
	I2C0_SDA	I/O	MFP4	I^2C0 data input/output pin.
	SPI1_MOSI	I/O	MFP5	SPI1 MOSI (Master Out, Slave In) pin.
PD.7	PD.7	I/O	MFP0	General purpose digital I/O pin.
	UART1_TXD	O	MFP3	UART1 data transmitter output pin.
	I2C0_SCL	I/O	MFP4	I^2C0 clock pin.
	SPI1_MISO	I/O	MFP5	SPI1 MISO (Master In, Slave Out) pin.
PD.8	PD.8	I/O	MFP0	General purpose digital I/O pin.
	UART2_nRTS	O	MFP4	UART2 request to Send output pin.
PD.9	PD.9	I/O	MFP0	General purpose digital I/O pin.
	UART2_nCTS	I	MFP4	UART2 clear to Send input pin.
PD.10	PD.10	I/O	MFP0	General purpose digital I/O pin.
	EADC0_CH16	A	MFP1	EADC0 channel 16 analog input.
	UART1_RXD	I	MFP3	UART1 data receiver input pin.
	INT7	I	MFP15	External interrupt 7 input pin.
PD.11	PD.11	I/O	MFP0	General purpose digital I/O pin.
	EADC0_CH17	A	MFP1	EADC0 channel 17 analog input.
	UART1_TXD	O	MFP3	UART1 data transmitter output pin.
	INT6	I	MFP15	External interrupt 6 input pin.
PD.12	PD.12	I/O	MFP0	General purpose digital I/O pin.
	EADC0_CH18	A	MFP1	EADC0 channel 18 analog input.
	UART2_RXD	I	MFP7	UART2 data receiver input pin.
	BPWM0_CH5	I/O	MFP9	BPWM0 channel 5 output/capture input.
	CLKO	O	MFP13	Clock Out
	EADC0_ST	I	MFP14	EADC0 external trigger input.
	INT5	I	MFP15	External interrupt 5 input pin.
PD.13	PD.13	I/O	MFP0	General purpose digital I/O pin.
	SPI0_I2SMCLK	I/O	MFP4	SPI0 I^2S master clock output pin
	SPI1_I2SMCLK	I/O	MFP5	SPI1 I^2S master clock output pin
PD.14	PD.14	I/O	MFP0	General purpose digital I/O pin.
	SPI0_I2SMCLK	I/O	MFP5	SPI0 I^2S master clock output pin

	Pin Name	Type	MFP	Description
	EPWM0_CH4	I/O	MFP11	EPWM0 channel 4 output/capture input.
PD.15	PD.15	I/O	MFP0	General purpose digital I/O pin.
	EPWM0_CH5	I/O	MFP12	EPWM0 channel 5 output/capture input.
	TRACE_CLK	O	MFP13	ETM Trace Clock output pin
	TM3	I/O	MFP14	Timer3 event counter input/toggle output pin.
	INT1	I	MFP15	External interrupt 1 input pin.
PE.0	PE.0	I/O	MFP0	General purpose digital I/O pin.
	SPI1_MOSI	I/O	MFP6	SPI1 MOSI (Master Out, Slave In) pin.
	UART3_RXD	I	MFP7	UART3 data receiver input pin.
	I2C1_SDA	I/O	MFP8	I^2C1 data input/output pin.
	UART4_nRTS	O	MFP9	UART4 request to Send output pin.
PE.1	PE.1	I/O	MFP0	General purpose digital I/O pin.
	SPI1_MISO	I/O	MFP6	SPI1 MISO (Master In, Slave Out) pin.
	UART3_TXD	O	MFP7	UART3 data transmitter output pin.
	I2C1_SCL	I/O	MFP8	I^2C1 clock pin.
	UART4_nCTS	I	MFP9	UART4 clear to Send input pin.
PE.2	PE.2	I/O	MFP0	General purpose digital I/O pin.
	EPWM0_CH5	I/O	MFP12	EPWM0 channel 5 output/capture input.
	BPWM0_CH0	I/O	MFP13	BPWM0 channel 0 output/capture input.
PE.3	PE.3	I/O	MFP0	General purpose digital I/O pin.
	EPWM0_CH4	I/O	MFP12	EPWM0 channel 4 output/capture input.
	BPWM0_CH1	I/O	MFP13	BPWM0 channel 1 output/capture input.
PE.4	PE.4	I/O	MFP0	General purpose digital I/O pin.
	EPWM0_CH3	I/O	MFP12	EPWM0 channel 3 output/capture input.
	BPWM0_CH2	I/O	MFP13	BPWM0 channel 2 output/capture input.
PE.5	PE.5	I/O	MFP0	General purpose digital I/O pin.
	EPWM0_CH2	I/O	MFP12	EPWM0 channel 2 output/capture input.
	BPWM0_CH3	I/O	MFP13	BPWM0 channel 3 output/capture input.
PE.6	PE.6	I/O	MFP0	General purpose digital I/O pin.
	UART5_RXD	I	MFP8	UART5 data receiver input pin.
	EPWM0_CH1	I/O	MFP12	EPWM0 channel 1 output/capture input.
	BPWM0_CH4	I/O	MFP13	BPWM0 channel 4 output/capture input.

	Pin Name	Type	MFP	Description
PE.7	PE.7	I/O	MFP0	General purpose digital I/O pin.
	UART5_TXD	O	MFP8	UART5 data transmitter output pin.
	EPWM0_CH0	I/O	MFP12	EPWM0 channel 0 output/capture input.
	BPWM0_CH5	I/O	MFP13	BPWM0 channel 5 output/capture input.
PE.8	PE.8	I/O	MFP0	General purpose digital I/O pin.
	UART2_TXD	O	MFP7	UART2 data transmitter output pin.
	EPWM0_CH0	I/O	MFP10	EPWM0 channel 0 output/capture input.
	EPWM0_BRAKE0	I	MFP11	EPWM0 Brake 0 input pin.
	TRACE_DATA3	O	MFP14	ETM Trace Data 3 output pin
PE.9	PE.9	I/O	MFP0	General purpose digital I/O pin.
	UART2_RXD	I	MFP7	UART2 data receiver input pin.
	EPWM0_CH1	I/O	MFP10	EPWM0 channel 1 output/capture input.
	EPWM0_BRAKE1	I	MFP11	EPWM0 Brake 1 input pin.
	TRACE_DATA2	O	MFP14	ETM Trace Data 2 output pin
PE.10	PE.10	I/O	MFP0	General purpose digital I/O pin.
	UART3_TXD	O	MFP7	UART3 data transmitter output pin.
	EPWM0_CH2	I/O	MFP10	EPWM0 channel 2 output/capture input.
	EPWM1_BRAKE0	I	MFP11	EPWM1 Brake 0 input pin.
	TRACE_DATA1	O	MFP14	ETM Trace Data 1 output pin
PE.11	PE.11	I/O	MFP0	General purpose digital I/O pin.
	UART3_RXD	I	MFP7	UART3 data receiver input pin.
	UART1_nCTS	I	MFP8	UART1 clear to Send input pin.
	EPWM0_CH3	I/O	MFP10	EPWM0 channel 3 output/capture input.
	EPWM1_BRAKE1	I	MFP11	EPWM1 Brake 1 input pin.
	TRACE_DATA0	O	MFP14	ETM Trace Data 0 output pin
PE.12	PE.12	I/O	MFP0	General purpose digital I/O pin.
	UART1_nRTS	O	MFP8	UART1 request to Send output pin.
	EPWM0_CH4	I/O	MFP10	EPWM0 channel 4 output/capture input.
	TRACE_CLK	O	MFP14	ETM Trace Clock output pin
PE.13	PE.13	I/O	MFP0	General purpose digital I/O pin.
	I2C0_SCL	I/O	MFP4	I^2C0 clock pin.
	UART4_nRTS	O	MFP5	UART4 request to Send output pin.

	Pin Name	Type	MFP	Description
	UART1_TXD	O	MFP8	UART1 data transmitter output pin.
	EPWM0_CH5	I/O	MFP10	EPWM0 channel 5 output/capture input.
	EPWM1_CH0	I/O	MFP11	EPWM1 channel 0 output/capture input.
	BPWM1_CH5	I/O	MFP12	BPWM1 channel 5 output/capture input.
PE.14	PE.14	I/O	MFP0	General purpose digital I/O pin.
	UART2_TXD	O	MFP3	UART2 data transmitter output pin.
PE.15	PE.15	I/O	MFP0	General purpose digital I/O pin.
	UART2_RXD	I	MFP3	UART2 data receiver input pin.
PF.0	PF.0	I/O	MFP0	General purpose digital I/O pin.
	UART1_TXD	O	MFP2	UART1 data transmitter output pin.
	I2C1_SCL	I/O	MFP3	I^2C1 clock pin.
	UART0_TXD	O	MFP4	UART0 data transmitter output pin.
	BPWM1_CH0	I/O	MFP12	BPWM1 channel 0 output/capture input.
	ICE_DAT	I/O	MFP14	Serial wired debugger data pin. Note: It is recommended to use 100 kΩ pull-up resistor on ICE_DAT pin.
PF.1	PF.1	I/O	MFP0	General purpose digital I/O pin.
	UART1_RXD	I	MFP2	UART1 data receiver input pin.
	I2C1_SDA	I/O	MFP3	I^2C1 data input/output pin.
	UART0_RXD	I	MFP4	UART0 data receiver input pin.
	BPWM1_CH1	I/O	MFP12	BPWM1 channel 1 output/capture input.
	ICE_CLK	I	MFP14	Serial wired debugger clock pin. Note: It is recommended to use 100 kΩ pull-up resistor on ICE_CLK pin.
PF.2	PF.2	I/O	MFP0	General purpose digital I/O pin.
	UART0_RXD	I	MFP3	UART0 data receiver input pin.
	I2C0_SDA	I/O	MFP4	I^2C0 data input/output pin.
	XT1_OUT	O	MFP10	External 4~24 MHz (high speed) crystal output pin.
	BPWM1_CH1	I/O	MFP11	BPWM1 channel 1 output/capture input.
PF.3	PF.3	I/O	MFP0	General purpose digital I/O pin.
	UART0_TXD	O	MFP3	UART0 data transmitter output pin.
	I2C0_SCL	I/O	MFP4	I^2C0 clock pin.
	XT1_IN	I	MFP10	External 4~24 MHz (high speed) crystal input pin.

	Pin Name	Type	MFP	Description
	BPWM1_CH0	I/O	MFP11	BPWM1 channel 0 output/capture input.
PF.4	PF.4	I/O	MFP0	General purpose digital I/O pin.
	UART2_TXD	O	MFP2	UART2 data transmitter output pin.
	UART2_nRTS	O	MFP4	UART2 request to Send output pin.
	EPWM0_CH1	I/O	MFP7	EPWM0 channel 1 output/capture input.
	BPWM0_CH5	I/O	MFP8	BPWM0 channel 5 output/capture input.
	X32_OUT	O	MFP10	External 32.768 kHz crystal output pin.
PF.5	PF.5	I/O	MFP0	General purpose digital I/O pin.
	UART2_RXD	I	MFP2	UART2 data receiver input pin.
	UART2_nCTS	I	MFP4	UART2 clear to Send input pin.
	EPWM0_CH0	I/O	MFP7	EPWM0 channel 0 output/capture input.
	BPWM0_CH4	I/O	MFP8	BPWM0 channel 4 output/capture input.
	EPWM0_SYNC_OUT	O	MFP9	EPWM0 counter synchronous trigger output pin.
	X32_IN	I	MFP10	External 32.768 kHz crystal input pin.
	EADC0_ST	I	MFP11	EADC0 external trigger input.
PF.6	PF.6	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MOSI	I/O	MFP5	SPI0 MOSI (Master Out, Slave In) pin.
	UART4_RXD	I	MFP6	UART4 data receiver input pin.
PF.7	PF.7	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MISO	I/O	MFP5	SPI0 MISO (Master In, Slave Out) pin.
	UART4_TXD	O	MFP6	UART4 data transmitter output pin.
PF.8	PF.8	I/O	MFP0	General purpose digital I/O pin.
	SPI0_CLK	I/O	MFP5	SPI0 serial clock pin.
	UART5_nCTS	I	MFP6	UART5 clear to Send input pin.
PF.9	PF.9	I/O	MFP0	General purpose digital I/O pin.
	SPI0_SS	I/O	MFP5	SPI0 slave select pin.
	UART5_nRTS	O	MFP6	UART5 request to Send output pin.
PF.10	PF.10	I/O	MFP0	General purpose digital I/O pin.
	SPI0_I2SMCLK	I/O	MFP5	SPI0 I^2S master clock output pin
	UART5_RXD	I	MFP6	UART5 data receiver input pin.
PF.11	PF.11	I/O	MFP0	General purpose digital I/O pin.
	UART5_TXD	O	MFP6	UART5 data transmitter output pin.

	Pin Name	Type	MFP	Description
	TM3	I/O	MFP13	Timer3 event counter input/toggle output pin.
PF.12	PF.12	I/O	MFP0	General purpose digital I/O pin.
	I2C0_SCL	I/O	MFP4	I^2C0 clock pin.
	UART5_TXD	O	MFP6	UART5 data transmitter output pin.
	UART3_TXD	O	MFP7	UART3 data transmitter output pin.
	UART1_TXD	O	MFP8	UART1 data transmitter output pin.
	I2C1_SCL	I/O	MFP9	I^2C1 clock pin.
	EPWM0_CH2	I/O	MFP10	EPWM0 channel 2 output/capture input.
	BPWM0_CH0	I/O	MFP11	BPWM0 channel 0 output/capture input.
	TRACE_DATA1	O	MFP13	ETM Trace Data 1 output pin
	CIR0_RXD	I	MFP15	CIR0 data receiver input pin.
PF.13	PF.13	I/O	MFP0	General purpose digital I/O pin.
	I2C0_SDA	I/O	MFP4	I^2C0 data input/output pin.
	UART5_RXD	I	MFP6	UART5 data receiver input pin.
	UART3_RXD	I	MFP7	UART3 data receiver input pin.
	UART1_RXD	I	MFP8	UART1 data receiver input pin.
	I2C1_SDA	I/O	MFP9	I^2C1 data input/output pin.
	EPWM0_CH3	I/O	MFP10	EPWM0 channel 3 output/capture input.
	BPWM0_CH1	I/O	MFP11	BPWM0 channel 1 output/capture input.
	TRACE_DATA0	O	MFP13	ETM Trace Data 0 output pin
PF.14	PF.14	I/O	MFP0	General purpose digital I/O pin.
	EPWM1_BRAKE0	I	MFP9	EPWM1 Brake 0 input pin.
	EPWM0_BRAKE0	I	MFP10	EPWM0 Brake 0 input pin.
	EPWM0_CH4	I/O	MFP12	EPWM0 channel 4 output/capture input.
	CLKO	O	MFP13	Clock Out
	TM3	I/O	MFP14	Timer3 event counter input/toggle output pin.
	INT5	I	MFP15	External interrupt 5 input pin.
PF.15	PF.15	I/O	MFP0	General purpose digital I/O pin.
	EPWM0_BRAKE0	I	MFP11	EPWM0 Brake 0 input pin.
	EPWM0_CH1	I/O	MFP12	EPWM0 channel 1 output/capture input.
	TM2	I/O	MFP13	Timer2 event counter input/toggle output pin.
	CLKO	O	MFP14	Clock Out

	Pin Name	Type	MFP	Description
	INT4	I	MFP15	External interrupt 4 input pin.
PG.2	PG.2	I/O	MFP0	General purpose digital I/O pin.
	I2C0_SMBAL	O	MFP4	I^2C0 SMBus SMBALTER pin
	I2C1_SCL	I/O	MFP5	I^2C1 clock pin.
	TM0	I/O	MFP13	Timer0 event counter input/toggle output pin.
PG.3	PG.3	I/O	MFP0	General purpose digital I/O pin.
	I2C0_SMBSUS	O	MFP4	I^2C0 SMBus SMBSUS pin (PMBus CONTROL pin)
	I2C1_SDA	I/O	MFP5	I^2C1 data input/output pin.
	TM1	I/O	MFP13	Timer1 event counter input/toggle output pin.
PG.4	PG.4	I/O	MFP0	General purpose digital I/O pin.
	TM2	I/O	MFP13	Timer2 event counter input/toggle output pin.
PG.9	PG.9	I/O	MFP0	General purpose digital I/O pin.
	BPWM0_CH5	I/O	MFP12	BPWM0 channel 5 output/capture input.
PG.10	PG.10	I/O	MFP0	General purpose digital I/O pin.
	BPWM0_CH4	I/O	MFP12	BPWM0 channel 4 output/capture input.
PG.11	PG.11	I/O	MFP0	General purpose digital I/O pin.
	BPWM0_CH3	I/O	MFP12	BPWM0 channel 3 output/capture input.
PG.12	PG.12	I/O	MFP0	General purpose digital I/O pin.
	BPWM0_CH2	I/O	MFP12	BPWM0 channel 2 output/capture input.
PG.13	PG.13	I/O	MFP0	General purpose digital I/O pin.
	BPWM0_CH1	I/O	MFP12	BPWM0 channel 1 output/capture input.
PG.14	PG.14	I/O	MFP0	General purpose digital I/O pin.
	BPWM0_CH0	I/O	MFP12	BPWM0 channel 0 output/capture input.
PG.15	PG.15	I/O	MFP0	General purpose digital I/O pin.
	CLKO	O	MFP14	Clock Out
	EADC0_ST	I	MFP15	EADC0 external trigger input.
PH.4	PH.4	I/O	MFP0	General purpose digital I/O pin.
	SPI1_MISO	I/O	MFP3	SPI1 MISO (Master In, Slave Out) pin.
PH.5	PH.5	I/O	MFP0	General purpose digital I/O pin.
	SPI1_MOSI	I/O	MFP3	SPI1 MOSI (Master Out, Slave In) pin.
PH.6	PH.6	I/O	MFP0	General purpose digital I/O pin.
	SPI1_CLK	I/O	MFP3	SPI1 serial clock pin.

	Pin Name	Type	MFP	Description
PH.7	PH.7	I/O	MFP0	General purpose digital I/O pin.
	SPI1_SS	I/O	MFP3	SPI1 slave select pin.
PH.8	PH.8	I/O	MFP0	General purpose digital I/O pin.
	SPI1_CLK	I/O	MFP6	SPI1 serial clock pin.
	UART3_nRTS	O	MFP7	UART3 request to Send output pin.
	I2C1_SMBAL	O	MFP8	I^2C1 SMBus SMBALTER pin
	UART1_TXD	O	MFP10	UART1 data transmitter output pin.
PH.9	PH.9	I/O	MFP0	General purpose digital I/O pin.
	SPI1_SS	I/O	MFP6	SPI1 slave select pin.
	UART3_nCTS	I	MFP7	UART3 clear to Send input pin.
	I2C1_SMBSUS	O	MFP8	I^2C1 SMBus SMBSUS pin (PMBus CONTROL pin)
	UART1_RXD	I	MFP10	UART1 data receiver input pin.
PH.10	PH.10	I/O	MFP0	General purpose digital I/O pin.
	SPI1_I2SMCLK	I/O	MFP6	SPI1 I^2S master clock output pin
	UART4_TXD	O	MFP7	UART4 data transmitter output pin.
	UART0_TXD	O	MFP8	UART0 data transmitter output pin.
PH.11	PH.11	I/O	MFP0	General purpose digital I/O pin.
	UART4_RXD	I	MFP7	UART4 data receiver input pin.
	UART0_RXD	I	MFP8	UART0 data receiver input pin.
	EPWM0_CH5	I/O	MFP11	EPWM0 channel 5 output/capture input.
PI.0	PI.0	I/O	MFP0	General purpose digital I/O pin.
	UART0_TXD	O	MFP3	UART0 data transmitter output pin.
	I2C0_SCL	I/O	MFP4	I^2C0 clock pin.
	UART5_TXD	O	MFP6	UART5 data transmitter output pin.
	UART2_TXD	O	MFP7	UART2 data transmitter output pin.
	I2C1_SCL	I/O	MFP9	I^2C1 clock pin.
	EPWM0_CH4	I/O	MFP11	EPWM0 channel 4 output/capture input.
	EPWM1_CH0	I/O	MFP12	EPWM1 channel 0 output/capture input.
	ACMP0_O	O	MFP14	Analog comparator 0 output pin.
PI.1	PI.1	I/O	MFP0	General purpose digital I/O pin.
	UART0_RXD	I	MFP3	UART0 data receiver input pin.
	I2C0_SDA	I/O	MFP4	I^2C0 data input/output pin.

	Pin Name	Type	MFP	Description
	UART5_RXD	I	MFP6	UART5 data receiver input pin.
	UART2_RXD	I	MFP7	UART2 data receiver input pin.
	I2C1_SDA	I/O	MFP9	I^2C1 data input/output pin.
	EPWM0_CH5	I/O	MFP11	EPWM0 channel 5 output/capture input.
	EPWM1_CH1	I/O	MFP12	EPWM1 channel 1 output/capture input.
	ACMP1_O	O	MFP14	Analog comparator 1 output pin.
PI.2	PI.2	I/O	MFP0	General purpose digital I/O pin.
	UART1_TXD	O	MFP3	UART1 data transmitter output pin.
	I2C0_SCL	I/O	MFP4	I^2C0 clock pin.
	UART5_TXD	O	MFP6	UART5 data transmitter output pin.
	I2C1_SCL	I/O	MFP9	I^2C1 clock pin.
	UART4_TXD	O	MFP11	UART4 data transmitter output pin.
	BPWM0_CH5	I/O	MFP12	BPWM0 channel 5 output/capture input.
	EADC0_ST	I	MFP15	EADC0 external trigger input.
PI.3	PI.3	I/O	MFP0	General purpose digital I/O pin.
	UART1_RXD	I	MFP3	UART1 data receiver input pin.
	I2C0_SDA	I/O	MFP4	I^2C0 data input/output pin.
	UART5_RXD	I	MFP6	UART5 data receiver input pin.
	I2C1_SDA	I/O	MFP9	I^2C1 data input/output pin.
	UART4_RXD	I	MFP11	UART4 data receiver input pin.
	BPWM0_CH4	I/O	MFP12	BPWM0 channel 4 output/capture input.
PI.4	PI.4	I/O	MFP0	General purpose digital I/O pin.
	UART4_TXD	O	MFP7	UART4 data transmitter output pin.
	UART0_TXD	O	MFP8	UART0 data transmitter output pin.
	UART1_TXD	O	MFP10	UART1 data transmitter output pin.
	EPWM0_CH5	I/O	MFP11	EPWM0 channel 5 output/capture input.
	CIR0_RXD	I	MFP14	CIR0 data receiver input pin.
	ACMP0_O	O	MFP15	Analog comparator 0 output pin.
PI.5	PI.5	I/O	MFP0	General purpose digital I/O pin.
	UART4_RXD	I	MFP7	UART4 data receiver input pin.
	UART0_RXD	I	MFP8	UART0 data receiver input pin.
	UART1_RXD	I	MFP10	UART1 data receiver input pin.

	Pin Name	Type	MFP	Description
	EPWM0_CH4	I/O	MFP11	EPWM0 channel 4 output/capture input.
	ACMP1_O	O	MFP15	Analog comparator 1 output pin.

Table 4.3-1 GPIO Multi-function Table

5 BLOCK DIAGRAM

5.1 NuMicro® M471V/M471K/M471C Block Diagram

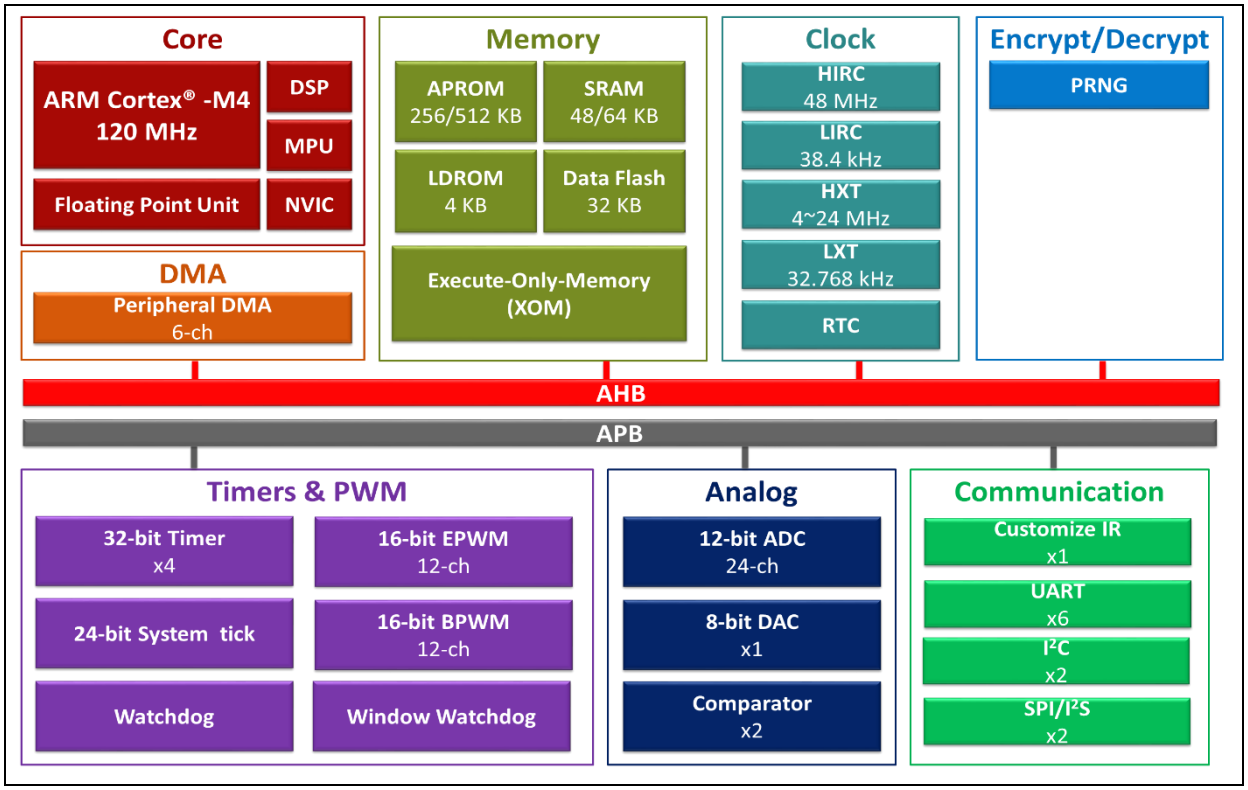


Figure 5.1-1 NuMicro® M471V/M471K/M471C Block Diagram

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 NuMicro® M471V/M471K/M471C series is embedded with Cortex®-M4F processor. Throughout this document, the name Cortex®-M4 refers to both Cortex®-M4 and Cortex®-M4F processors. Figure 6.1-1 shows the functional controller of the processor.

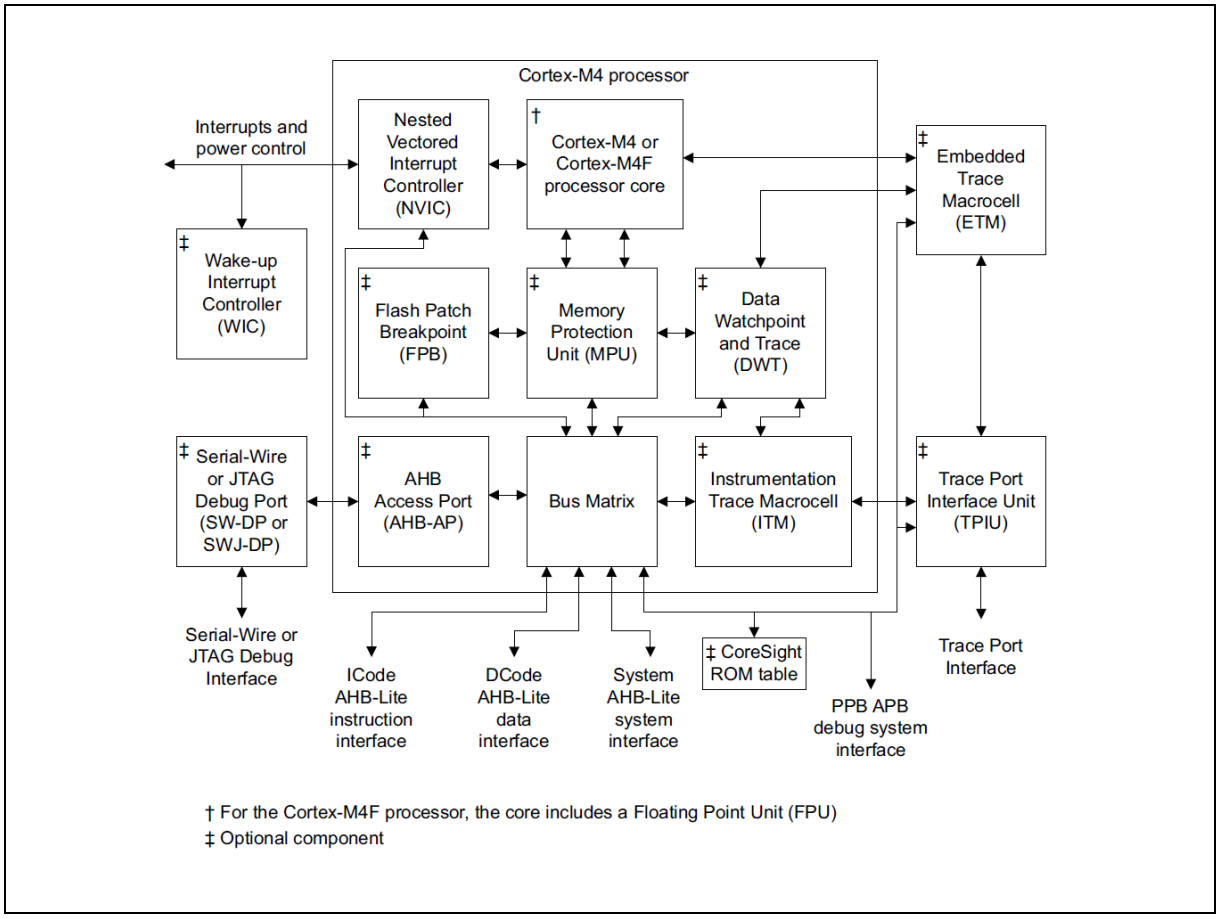


Figure 6.1-1 Cortex®-M4 Block Diagram

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 NuMicro® M471V/M471K/M471C series configured with 64 interrupts)
  - Bits of priority, configurable from 3 to 8
  - Dynamic reprioritization of interrupts
  - Priority grouping which enables selection of preempting interrupt levels and nonpreempting interrupt levels
  - Support for tril-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) or Serial Wire JTAG Debug Port (SWJ-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
- Optional Embedded Trace Macrocell (ETM) for instruction trace.

- 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 Orginization
- System Timer (SysTick)
- Nested Vectored Interrupt Controller (NVIC)
- System Control register

6.2.2 System Reset

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

- 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 will reset whole chip by writing 1 to CHIPRST (SYS_IPRST0[0])
  - MCU Reset to reboot but keeping the booting setting from APROM or LDROM by writing 1 to SYSRESETREQ (AIRCR[2])
  - CPU Reset for Cortex®-M4 core only by writing 1 to CPURST (SYS_IPRST0[1])

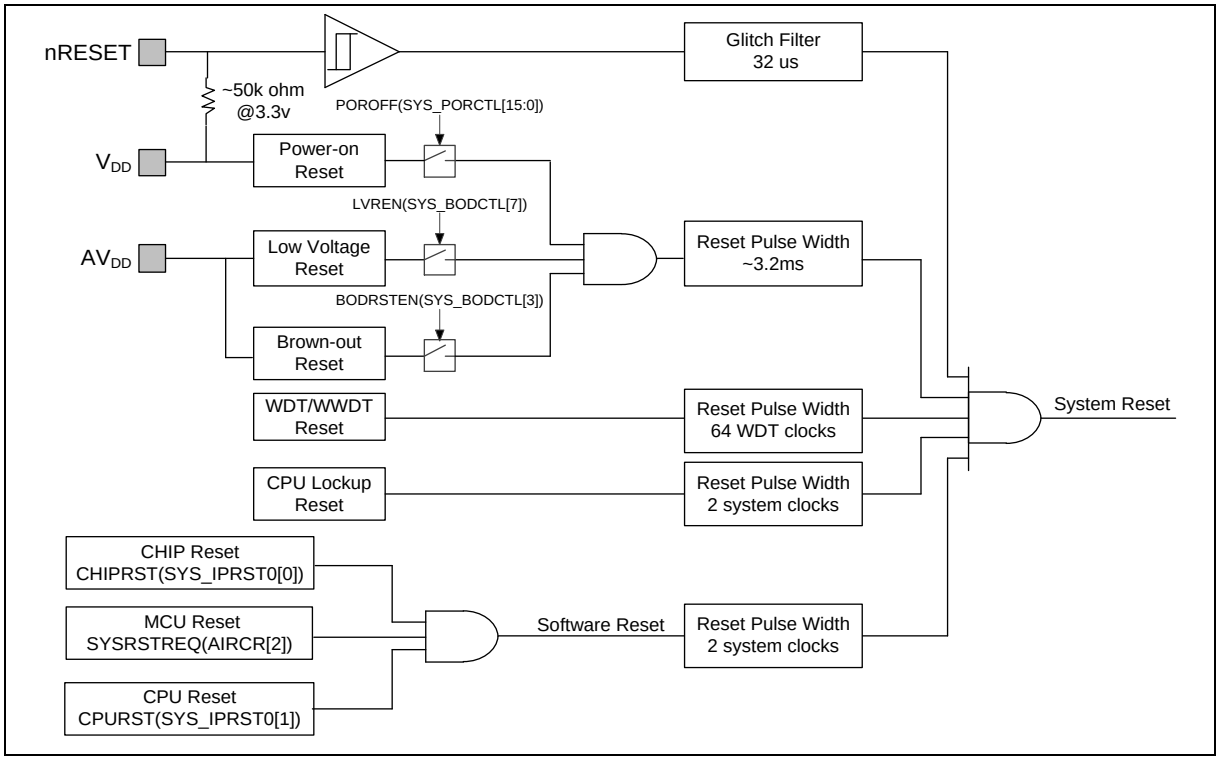


Figure 6.2-1 System Reset Sources

There are a total of 9 reset sources in the NuMicro® family. In general, CPU reset is used to reset Cortex®-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 6.2-1

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[2:1])									
BODRSTEN (SYS_BODCTL[3])									
HXTEN (CLK_PWRCTL[0])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	
LXTEN (CLK_PWRCTL[1])	0x0	-	-	-	-	-	-	-	-
WDTCKEN (CLK_APBCLK0[0])	0x1	-	0x1	-	-	-	0x1	-	-
HCLKSEL	Reload from	Reload from	Reload from	Reload from	Reload from	Reload from	Reload from	Reload from	-

(CLK_CLKSEL0[2:0])	CONFIG0	CONFIG0	CONFIG0	CONFIG0	CONFIG0	CONFIG0	CONFIG0	CONFIG0	
WDTSEL (CLK_CLKSEL1[1:0])	0x3	0x3	-	-	-	-	-	-	-
HXTSTB (CLK_STATUS[0])	0x0	-	-	-	-	-	-	-	-
LXTSTB (CLK_STATUS[1])	0x0	-	-	-	-	-	-	-	-
PLLSTB (CLK_STATUS[2])	0x0	-	-	-	-	-	-	-	-
HIRCSTB (CLK_STATUS[4])	0x0	-	-	-	-	-	-	-	-
CLKSFAIL (CLK_STATUS[7])	0x0	0x0	-	-	-	-	-	-	-
RSTEN (WDT_CTL[1]) WDTEN (WDT_CTL[7])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	Reload from CONFIG0	-	-
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]) BL (FMC_ISPCTL[16])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	Reload from CONFIG0	-	-
CBS (FMC_ISPSTS[2:1))	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	Reload from CONFIG0	-	-
VECMAP (FMC_ISPSTS[23:9])	Reload base on CONFIG0	Reload base on CONFIG0	Reload base on CONFIG0	Reload base on CONFIG0	Reload base on CONFIG0	-	Reload base on CONFIG0	-	-
Other Peripheral Registers	Reset Value								-
FMC Registers	Reset Value								
Note: '-' means that the value of register keeps original setting.									

Table 6.2-1 Reset Value of Registers

6.2.2.1 nRESET Reset

The nRESET reset means to generate a reset signal by pulling low nRESET pin, which is an

asynchronous reset input pin and can be used to reset system at any time. When the nRESET voltage is lower than 0.2 V_{DD} and the state keeps longer than 32 us (glitch filter), chip will be reset. The nRESET reset will control the chip in reset state until the nRESET voltage rises above 0.7 V_{DD} and the state keeps longer than 32 us (glitch filter). The PINRF(SYS_RSTSTS[1]) will be set to 1 if the previous reset source is nRESET reset. Figure 6.2-2 shows the nRESET reset waveform.

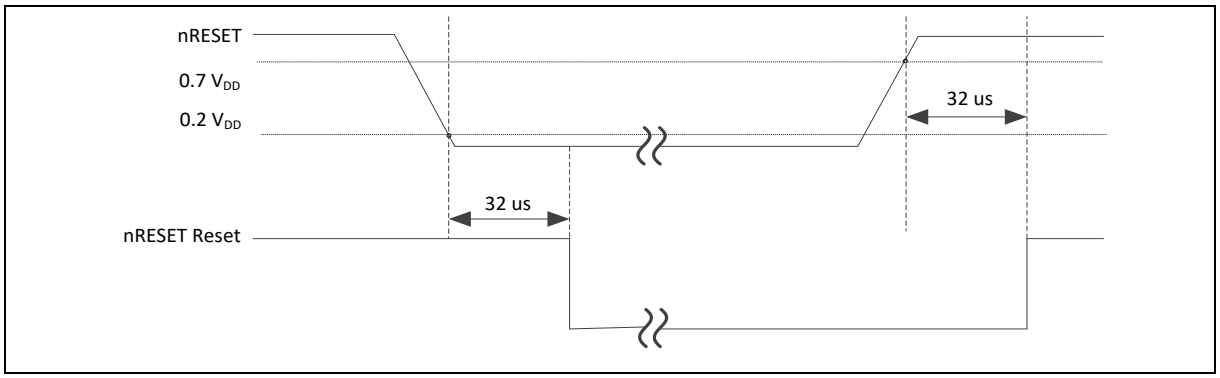


Figure 6.2-2 nRESET Reset Waveform

6.2.2.2 Power-on Reset (POR)

The Power-on reset (POR) is used to generate a stable system reset signal and forces the system to be reset when power-on to avoid unexpected behavior of MCU. When applying the power to MCU, the POR module will detect the rising voltage and generate reset signal to system until the voltage is ready for MCU operation. At POR reset, the PORF(SYS_RSTSTS[0]) will be set to 1 to indicate there is a POR reset event. The PORF(SYS_RSTSTS[0]) bit can be cleared by writing 1 to it. Figure 6.2-3 shows the power-on reset waveform.

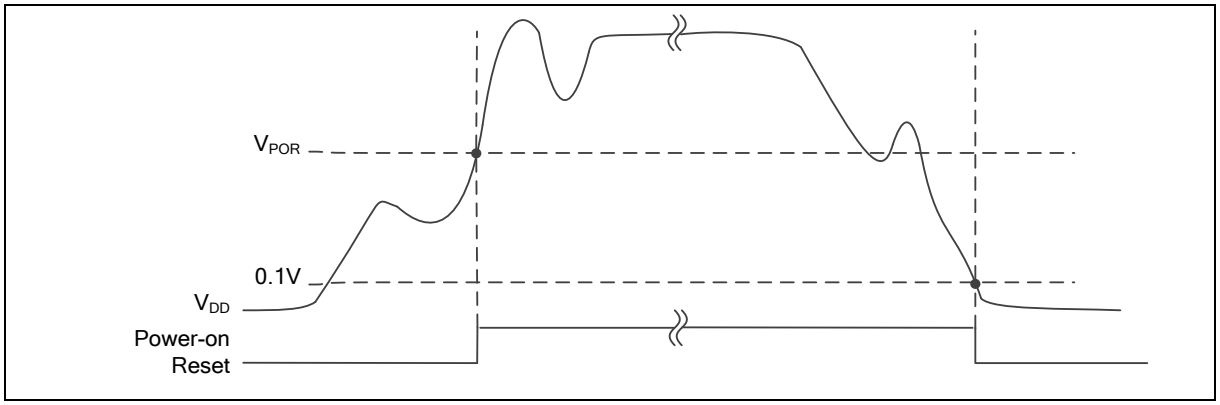


Figure 6.2-3 Power-on Reset (POR) Waveform

6.2.2.3 Low Voltage Reset (LVR)

If the Low Voltage Reset function is enabled by setting the Low Voltage Reset Enable Bit LVREN (SYS_BODCTL[7]) to 1, after 200us delay, LVR detection circuit will be stable and the LVR function will be active. Then LVR function will detect AV_{DD} during system operation. When the AV_{DD} voltage is lower than V_{LVR} and the state keeps longer than De-glitch time set by LVRDGSEL (SYS_BODCTL[14:12]), chip will be reset. The LVR reset will control the chip in reset state until the AV_{DD} voltage rises above V_{LVR} and the state keeps longer than De-glitch time set by LVRDGSEL (SYS_BODCTL[14:12]). The default setting of Low Voltage Reset is enabled without De-glitch function. Figure 6.2-4 shows the Low Voltage Reset waveform.

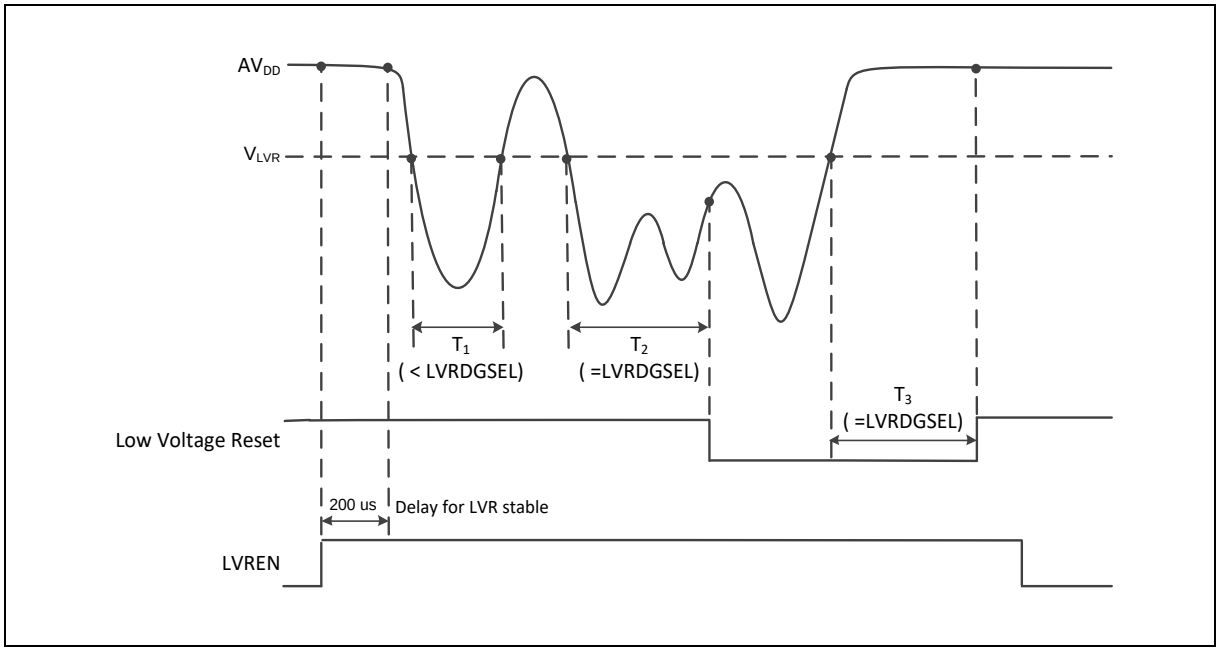


Figure 6.2-4 Low Voltage Reset (LVR) Waveform

6.2.2.4 Brown-out Detector Reset (BOD Reset)

If the Brown-out Detector (BOD) function is enabled by setting the Brown-out Detector Enable Bit BODEN (SYS_BODCTL[0]), Brown-out Detector function will detect AV_{DD} during system operation. When the AV_{DD} voltage is lower than V_{BOD} which is decided by BODEN and BODVL (SYS_BODCTL[17:16]) and the state keeps longer than De-glitch time set by BODDGSEL (SYS_BODCTL[10:8]), chip will be reset. The BOD reset will control the chip in reset state until the AV_{DD} voltage rises above V_{BOD} and the state keeps longer than De-glitch time set by BODDGSEL. The default value of BODEN, BODVL and BODRSTEN (SYS_BODCTL[3]) is set by Flash controller user configuration register CBODEN (CONFIG0 [19]), CBOV (CONFIG0 [22:21]) and CBORST(CONFIG0[20]) respectively. User can determine the initial BOD setting by setting the CONFIG0 register. Figure 6.2-5 shows the Brown-out Detector waveform.

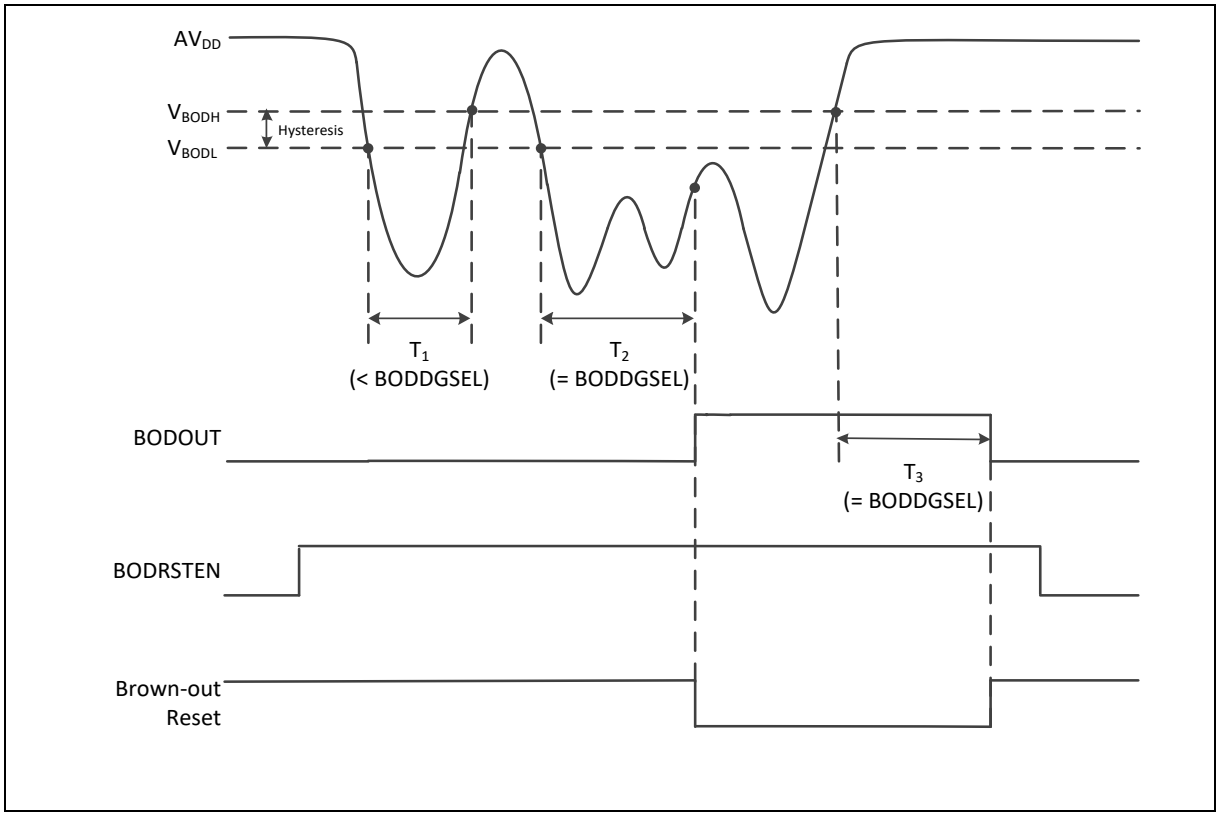


Figure 6.2-5 Brown-out Detector (BOD) Waveform

6.2.2.5 Watchdog Timer Reset (WDT)

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

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

6.2.2.6 CPU Lockup Reset

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

6.2.2.7 CPU Reset, CHIP Reset and MCU Reset

The CPU Reset means only Cortex®-M4 core is reset and all other peripherals remain the same status after CPU reset. User can set the CPURST(SYS_IPRST0[1]) to 1 to assert the CPU Reset signal.

The CHIP Reset is same with Power-on Reset. The CPU and all peripherals are reset and BS(FMC_ISPCTL[1]) bit is automatically reloaded from CONFIG0 setting. User can set the CHIPRST(SYS_IPRST0[0]) to 1 to assert the CHIP Reset signal.

The MCU Reset is similar with CHIP Reset. The difference is that BS(FMC_ISPCTL[1]) will not be reloaded from CONFIG0 setting and keep its original software setting for booting from APROM or

LDROM. User can set the SYSRESETREQ(AIRCR[2]) to 1 to assert the MCU Reset.

6.2.3 System Power Distribution

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

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

The outputs of internal voltage regulators (LDO) require an external capacitor which should be located close to the corresponding pin. Analog power (AV_{DD}) should be the same voltage level of the digital power (V_{DD}). Figure 6.2-6 shows the NuMicro® M471V/M471K/M471C power distribution.

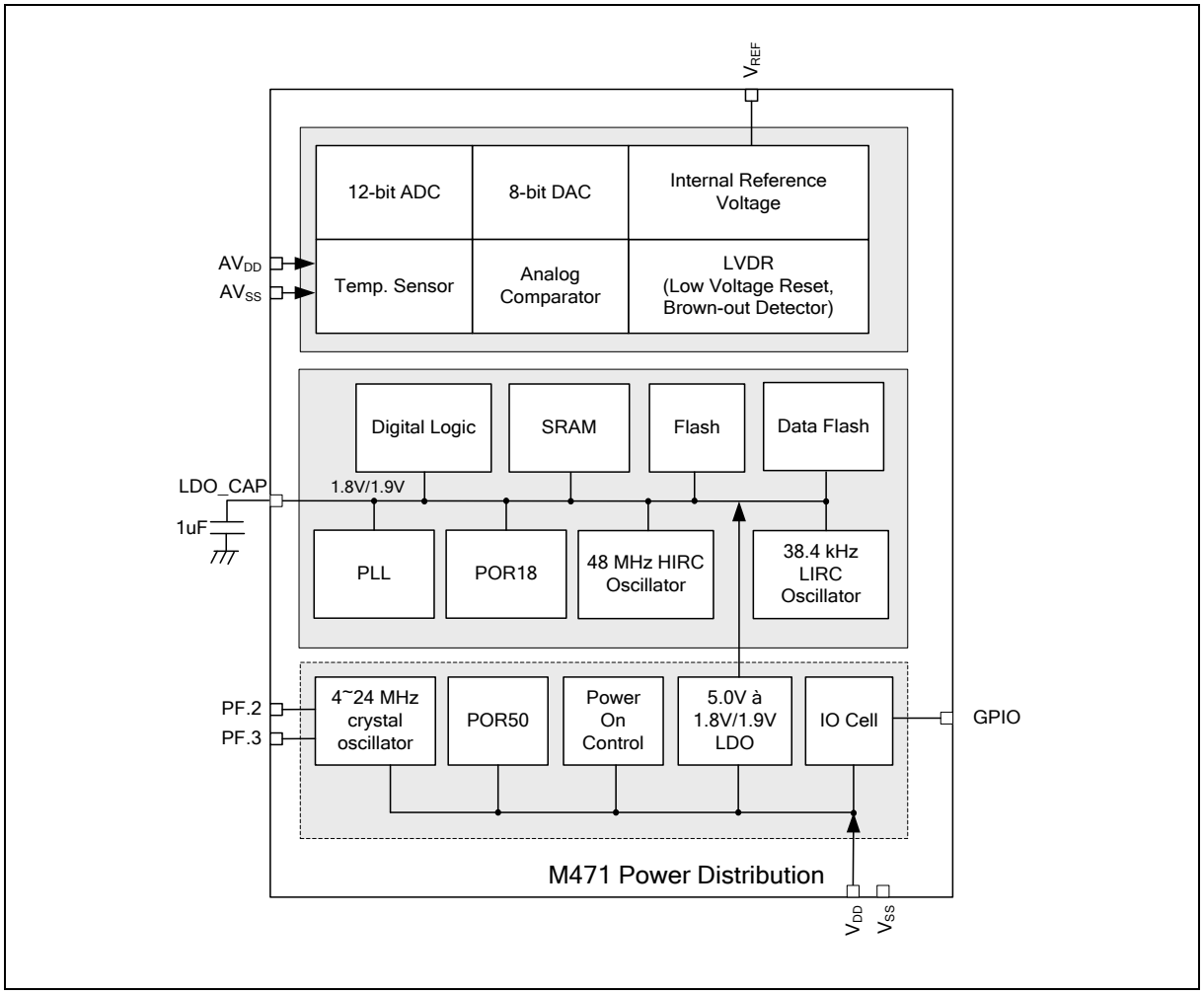


Figure 6.2-6 NuMicro® M471V/M471K/M471C Power Distribution Diagram

6.2.4 Power Modes and Wake-up Sources

The NuMicro® M471V/M471K/M471C series has power manager unit to support several operating modes for saving power. Table 6.2-2 lists all power modes in the M471V/M471K/M471C series.

Mode	CPU Operating Maximum Speed (MHz)	LDO_CAP (V)	Clock Disable
Normal mode	96	1.80	All clocks are disabled by control register.
Turbo mode	120	1.90	All clocks are disabled by control register.
Idle mode	CPU enter Sleep mode	1.80/1.90	Only CPU clock is disabled.
Normal Power-down mode (NPD)	CPU enters Deep Sleep mode	1.80/1.90	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.

Table 6.2-2 Power Mode Table

There are different power mode entry settings for each power mode. They have different entry setting and leaving condition. Table 6.2-3 shows the entry setting for each power mode. When chip power-on, chip is running at normal mode. User can enter each mode by setting SLEEPDEEP (SCR[2]), PDEN (CLK_PWRCTL[7]) and PDMSEL (CLK_PMUCTL[2:0]) and 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
Normal Power-down mode	1	1	0	YES

Table 6.2-3 Power Mode Difference Table

There are several wake-up sources in Idle mode and Power-down mode. Table 6.2-4 lists the available clocks for each power mode.

Power Mode	Normal Mode	Idle Mode	Power-Down Mode
Definition	CPU is in active state	CPU is in sleep state	CPU is in sleep state and all clocks stop except LXT and LIRC. SRAM content retended.
Entry Condition	Chip is in normal mode after system reset released	CPU executes WFI instruction.	CPU sets sleep mode enable and power down enable and executes WFI instruction.
Wake-Up Sources	N/A	All interrupts	RTC, WDT, I²C, Timer, UART, GPIO, EINT, ACMP, CIR0 and BOD.
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 6.2-4 Power Mode Definition Table

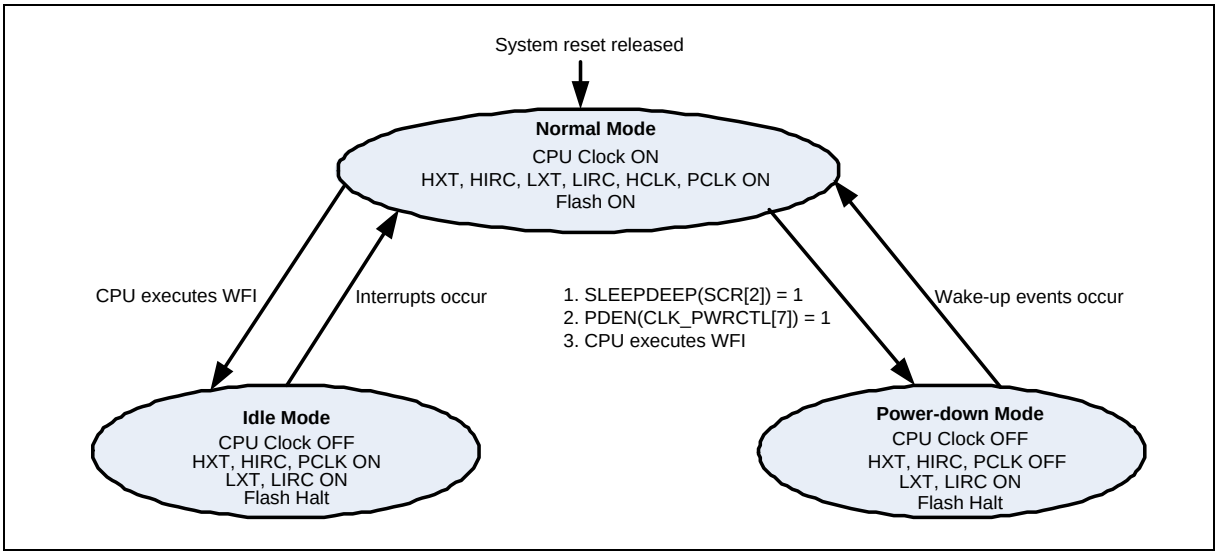


Figure 6.2-7 Power Mode State Machine

	Idle Mode	NPD
HXT	ON	Halt
HIRC	ON	Halt
LXT	ON	ON/OFF[1]
LIRC	ON	ON/OFF[2]
PLL	ON	Halt
HCLK/PCLK	ON	Halt
CPU	Halt	Halt
SRAM retention	ON	ON
FLASH	ON	Halt
TIMER	ON	ON/OFF[3]
WDT	ON	ON/OFF[4]
RTC	ON	ON/OFF[5]
UART	ON	ON/OFF[6]
CIR0	ON	ON/OFF[7]
Others	ON	Halt

Table 6.2-5 Clocks in Power Modes

Note:

1. LXT ON or OFF depends on SOFTWARE setting in normal mode.
2. LIRC ON or OFF depends on SOFTWARE 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.
7. If CIR0 clock source is selected as LIRC/LXT and LIRC/LXT is on.

Wake-up sources in Normal Power-down mode (NPD):

After chip enters power down, the following wake-up sources can wake chip up to normal mode. Table 6.2-6 lists the condition about how to enter Power-down mode again for each peripheral.

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

Wake-Up Source	Wake-Up Condition	Power-Down Mode	Re-Entering Power-Down Mode Condition
		NPD	
BOD	Brown-Out Detector Reset / Interrupt	V	After software writes 1 to clear BODIF (SYS_BODCTL[4]).
LVR	LVR Reset	V	After software writes 1 to clear LVRF (SYS_RSTSTS[3])
POR	POR Reset	V	After software writes 1 to clear PORF (SYS_RSTSTS[0])
EINT	External Interrupt	V	After software write 1 to clear the Px_INTSRC[n] bit.
GPIO	GPIO Interrupt	V	After software write 1 to clear the Px_INTSRC[n] bit.
TIMER	Timer Interrupt	V	After software writes 1 to clear TWKF (TIMERx_INTSTS[1]) and TIF (TIMERx_INTSTS[0]).
WDT	WDT Interrupt	V	After software writes 1 to clear WKF (WDT_CTL[5]) (Write Protect).
RTC	Alarm Interrupt	V	After software writes 1 to clear ALMIF (RTC_INTSTS[0]).
	Time Tick Interrupt	V	After software writes 1 to clear TICKIF (RTC_INTSTS[1]).
UART	nCTS Wake-Up	V	After software writes 1 to clear CTSWKF (UARTx_WKSTS[0]).
	RX Data Wake-Up	V	After software writes 1 to clear DATWKF (UARTx_WKSTS[1]).
	Received FIFO Threshold Wake-Up	V	After software writes 1 to clear RFRTWKF (UARTx_WKSTS[2]).
	RS-485 AAD Mode Wake-Up	V	After software writes 1 to clear RS485WKF (UARTx_WKSTS[3]).
	Received FIFO Threshold Time-out Wake-Up	V	After software writes 1 to clear TOUTWKF (UARTx_WKSTS[4]).
I^2C	Address match Wake-Up	V	After software writes 1 to clear WKAKDONE (I2C_WKSTS[1]). Then software writes 1 to clear WKIF(I2C_WKSTS[0]).
ACMP	Comparator Power-Down Wake-Up Interrupt	V	After software writes 1 to clear WKIF0 (ACMP_STATUS[8]) and WKIF1 (ACMP_STATUS[9]).
CIR0	CIR0 receive pattern match Wake-Up	V	After software writes 1 to clear PDWKF (CIR_STATUS[10]).

Table 6.2-6 Re-Entering Power-down Mode Condition

6.2.5 Power Modes and Power Level Transition

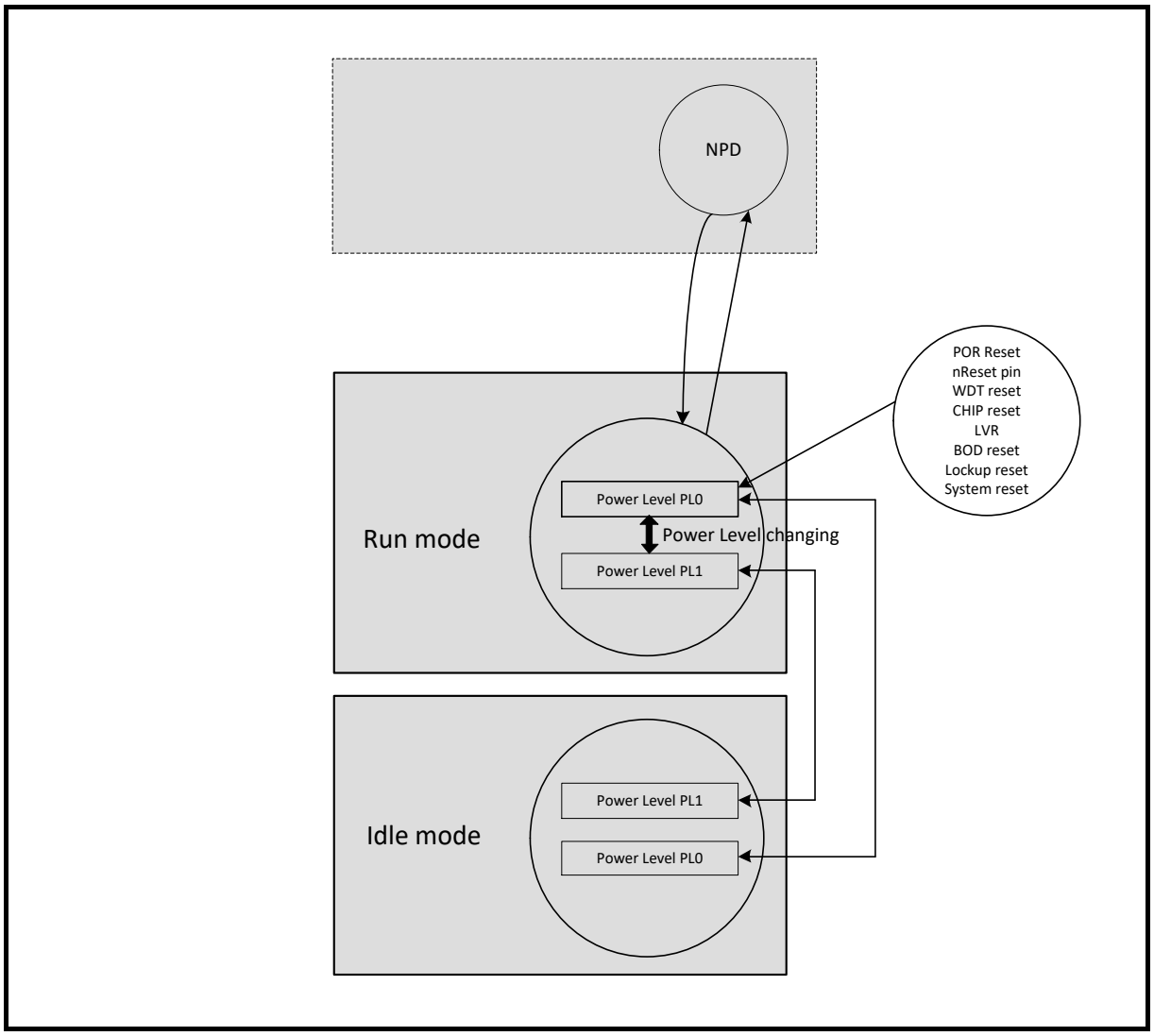


Figure 6.2-8 NuMicro® M471V/M471K/M471C Power Distribution Diagram

6.2.6 System Memory Map

The NuMicro® M471V/M471K/M471C series provides 4G-byte addressing space. The memory locations assigned to each on-chip controllers are shown in Table 6.2-7. The detailed register definition, memory space, and programming will be described in the following sections for each on-chip peripheral. The NuMicro® M471V/M471K/M471C 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)
0x0040_0000 – 0x0040_7FFF	DFLASH_BA	Data FLASH Memory Space (32 Kbytes)
0x2000_0000 – 0x2000_7FFF	SRAM0_BA	SRAM Memory Space (32 Kbytes)
0x2000_8000 – 0x2000_FFFF	SRAM1_BA	SRAM Memory Space (32 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
0x4000_F000 – 0x4000_FFFF	DFMC_BA	Data Flash Memory Control Registers
0x4003_1000 – 0x4003_1FFF	CRC_BA	CRC Generator Registers
APB Controllers Space (0x4000_0000 ~ 0x400F_FFFF)		
0x4004_0000 – 0x4004_00FF	WDT_BA	Watchdog Timer Control Registers
0x4004_0100 – 0x4004_0FFF	WWDT_BA	Window Watchdog Timer Control Registers
0x4004_1000 – 0x4004_1FFF	RTC_BA	Real Time Clock (RTC) Control Register
0x4004_3000 – 0x4004_3FFF	EADC0_BA	Enhanced Analog-Digital-Converter 0 (EADC0) Control Registers
0x4004_5000 – 0x4004_5FFF	ACMP01_BA	Analog Comparator 0/ 1 Control Registers
0x4004_7000 – 0x4004_7FFF	DAC_BA	DAC 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	EPWM0_BA	EPWM0 Control Registers
0x4005_9000 – 0x4005_9FFF	EPWM1_BA	EPWM1 Control Registers
0x4005_A000 – 0x4005_AFFF	BPWM0_BA	BPWM0 Control Registers
0x4005_B000 – 0x4005_BFFF	BPWM1_BA	BPWM1 Control Registers
0x4005_F000 – 0x4005_FFFF	CIR0_BA	Custermize IR Control Registers
0x4006_1000 – 0x4006_1FFF	SPI0_BA	SPI0 Control Registers
0x4006_2000 – 0x4006_2FFF	SPI1_BA	SPI1 Control Registers
0x4007_0000 – 0x4007_0FFF	UART0_BA	UART0 Control Registers
0x4007_1000 – 0x4007_1FFF	UART1_BA	UART1 Control Registers
0x4007_2000 – 0x4007_2FFF	UART2_BA	UART2 Control Registers
0x4007_3000 – 0x4007_3FFF	UART3_BA	UART3 Control Registers
0x4007_4000 – 0x4007_4FFF	UART4_BA	UART4 Control Registers
0x4007_5000 – 0x4007_5FFF	UART5_BA	UART5 Control Registers
0x4008_0000 – 0x4008_0FFF	I2C0_BA	I^2C0 Control Registers
0x4008_1000 – 0x4008_1FFF	I2C1_BA	I^2C1 Control Registers
0x400B_A000 – 0x400B_AFFF	PRNG_BA	PRNG Control Registers
System Controllers Space (0xE000_E000 ~ 0xE000_EFFF)		
0xE000_E010 – 0xE000_E0FF	SCS_BA	System Timer Control Registers

0xE000_E100 – 0xE000_ECFF	SCS_BA	External Interrupt Controller Control Registers
0xE000_ED00 – 0xE000_ED8F	SCS_BA	System Control Registers

Table 6.2-7 Address Space Assignments for On-Chip Controllers

6.2.7 SRAM Memory Orginization

The M471V/M471K/M471C supports embedded SRAM with total 64 Kbytes size and the SRAM organization is separated to two banks: SRAM bank0 and SRAM bank1. The first bank has 32 Kbytes address space, the second bank has 32 Kbyte address space. These two banks address space can be accessed simultaneously. The SRAM bank0 supports parity error check to make sure chip operating more stable.

- Supports total 64 Kbytes SRAM
- Supports byte / half word / word write
- Supports fixed 32 Kbytes SRAM bank0 for independent access
- Supports parity error check function for SRAM bank0
- Supports oversize response error
- Supports remap address to 0x1000_0000

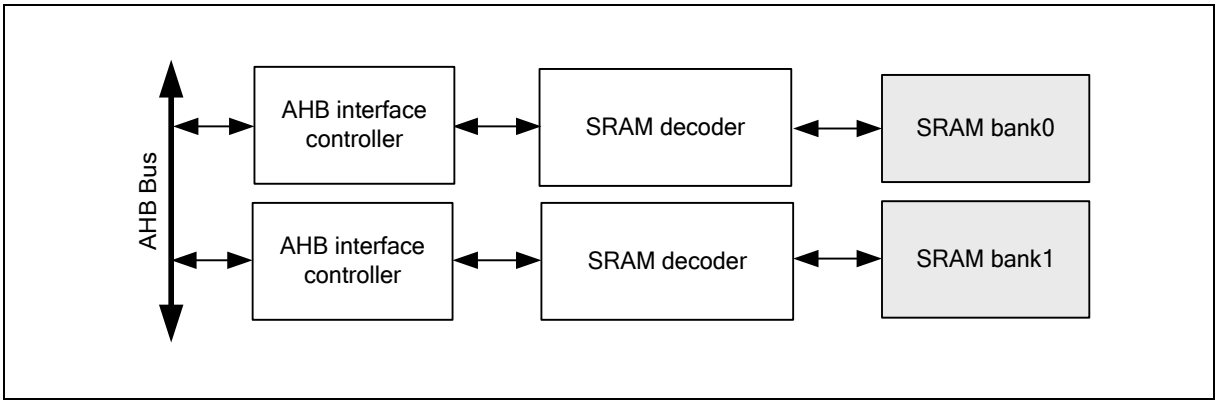


Figure 6.2-9 SRAM Block Diagram

Figure 6.2-9 shows the SRAM organization of M471. There are two SRAM banks in M471. The bank0 is addressed to 32 Kbytes and the bank1 is addressed to 32 Kbytes. The bank0 address space is from 0x2000_0000 to 0x2000_7FFF. The bank1 address space is from 0x2000_8000 to 0x2000_FFFF. The address between 0x2001_0000 to 0x3FFF_FFFF is illegal memory space and chip will enter hardfault if CPU accesses these illegal memory addresses.

The address of each bank is remapping from 0x2000_0000 to 0x1000_0000. CPU can access SRAM bank0 through 0x2000_0000 to 0x2000_7FFF or 0x1000_0000 to 0x1000_7FFF, and access SRAM bank1 through 0x2000_8000 to 0x2000_FFFF or 0x1000_8000 to 0x1000_FFFF.

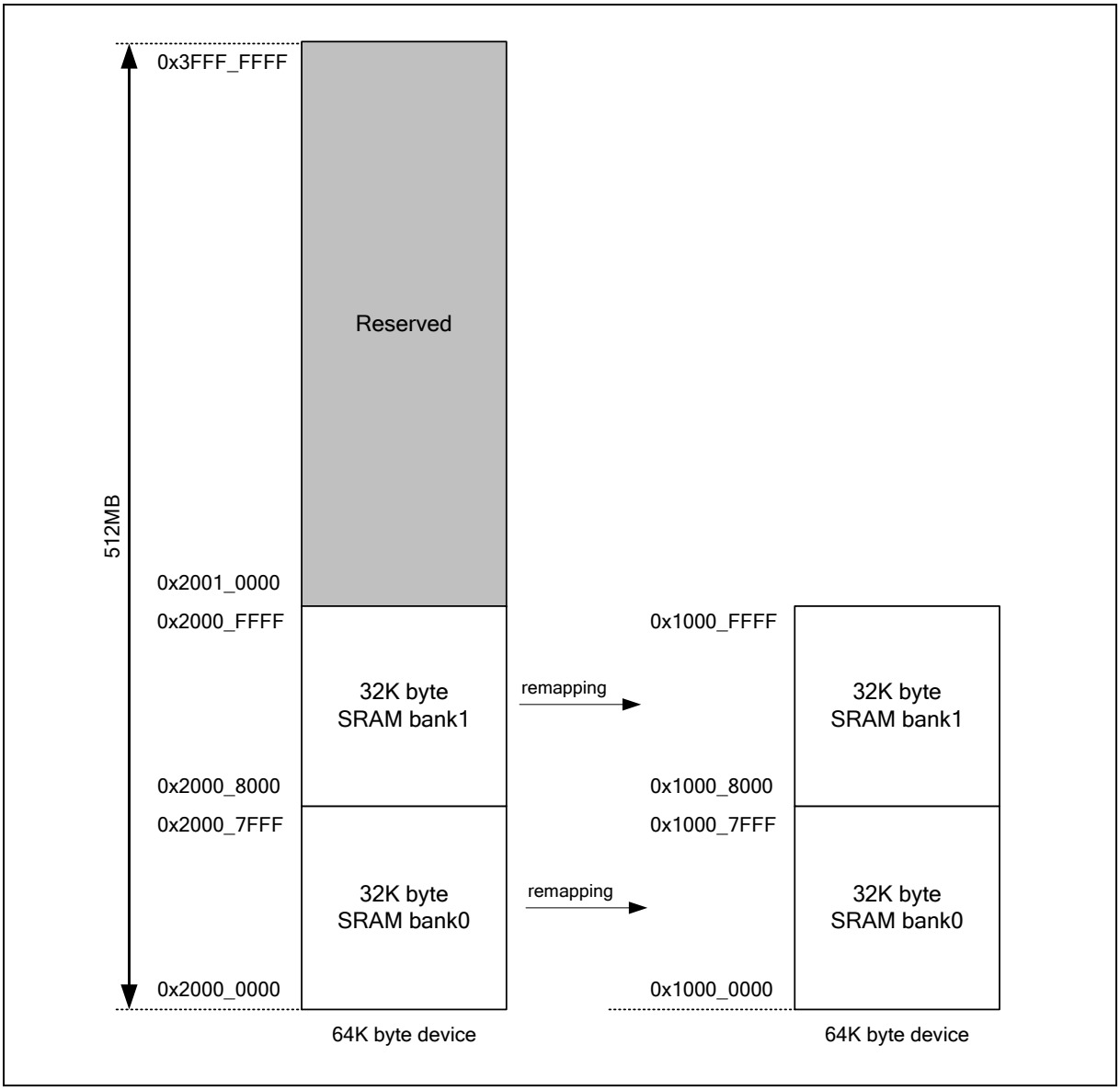


Figure 6.2-10 SRAM Memory Organization

SRAM bank0 has byte parity error check function. When CPU is accessing SRAM bank0, the parity error checking mechanism is dynamic operating. As parity error occured, the PERRIF (SYS_SRAM_STATUS[0]) will be asserted to 1 and the SYS_SRAM_ERRADDR register will recode the address with parity error. Chip will enter interrupt when SRAM parity error occurred if PERRIEN (SYS_SRAM_INTCTL[0]) is set to 1. When SRAM parity error occured, chip will stop detecting SRAM parity error until user writes 1 to clear the PERRIF(SYS_SRAM_STATUS[0]) bit.

6.2.8 Bus Matrix

The M471V/M471K/M471C supports Bus Matrix to manage the access arbitration between masters. The access arbitration can be selected by INTACTEN (SYS_AHBMCTL[0]) to use round-robin algorithm or set Cortex®-M4 CPU as the highest bus priority.

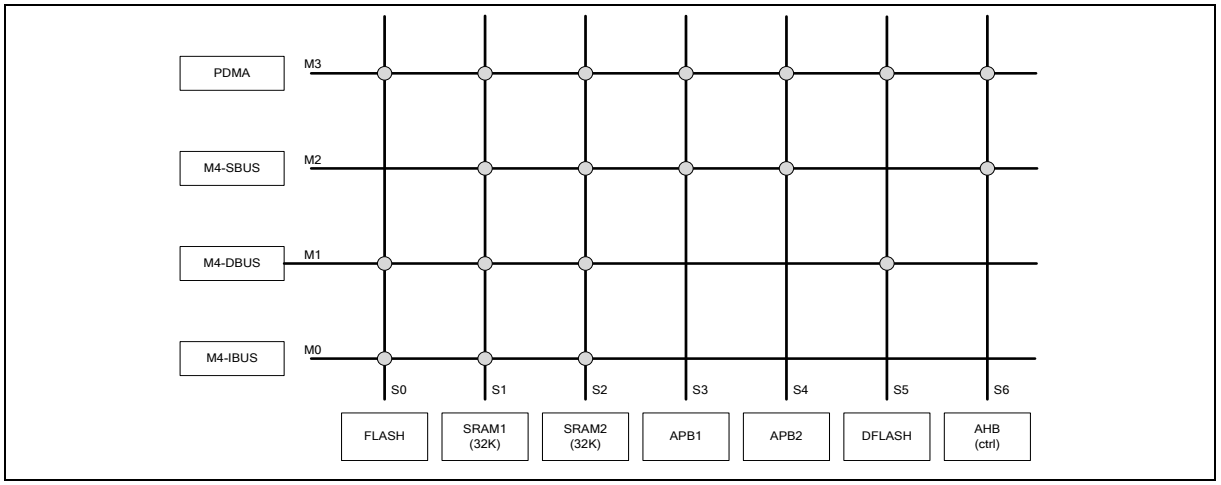


Figure 6.2-11 NuMicro® M471V/M471K/M471C Bus Matrix Diagram

6.2.9 HIRC Auto Trim

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

For instance, the system needs an accurate 48 MHz clock. In such case, if users do not want to use PLL as the system clock source, they need to solder 32.768 kHz crystal in system, user has to set REFCKSEL (SYS_IRCTCTL[10] reference clock selection) to “0”, set FREQSEL (SYS_IRCTCTL[1:0] trim frequency selection) to “01”, and the auto-trim function will be enabled. Interrupt status bit FREQLOCK (SYS_IRCTISTS[0] HIRC frequency lock status) “1” indicates the HIRC output frequency is accurate within 0.25% deviation.

6.2.10 Internal V_{REF}

This chip supports internal V_{REF} function, depend on setting of VREFCTL(SYS_VREFCTL[4:0]) to generate different reference voltage for ADC/DAC/ACMP. User has to enable PRELOADEN (SYS_VREFCTL[6]) to fit the Tstable when VREFCTL(SYS_VREFCTL[4:0]) change setting (except set to 00000) to make sure that V_{REF} works well. Tstable depends on different situations has different requirement, please refer to the relative Datasheet.

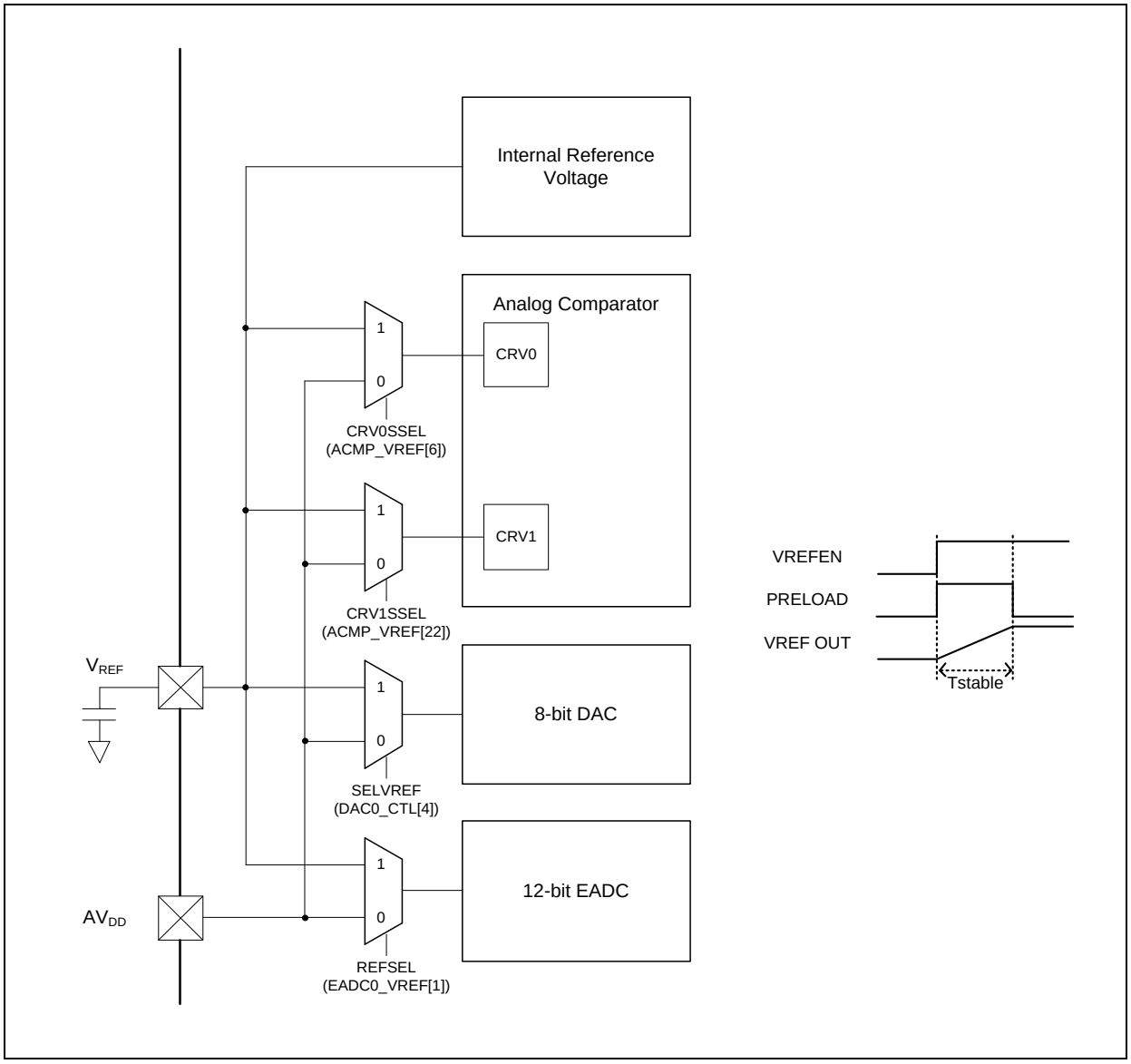


Figure 6.2-12 NuMicro® M471V/M471K/M471C V_{REF} Diagram

6.2.11 UART0_TXD Modulation with EPWM

This chip supports UART0_TXD to modulate with EPWM channel. User can set MODPWMSEL(SYS_MODCTL[7:4]) to choose which EPWM0 channel to modulate with UART0_TXD and set MODEN(SYS_MODCTL[0]) to enable modulation function. User can set TXDINV(UART_LINE[8]) to inverse UART0_TXD before modulating with EPWM.

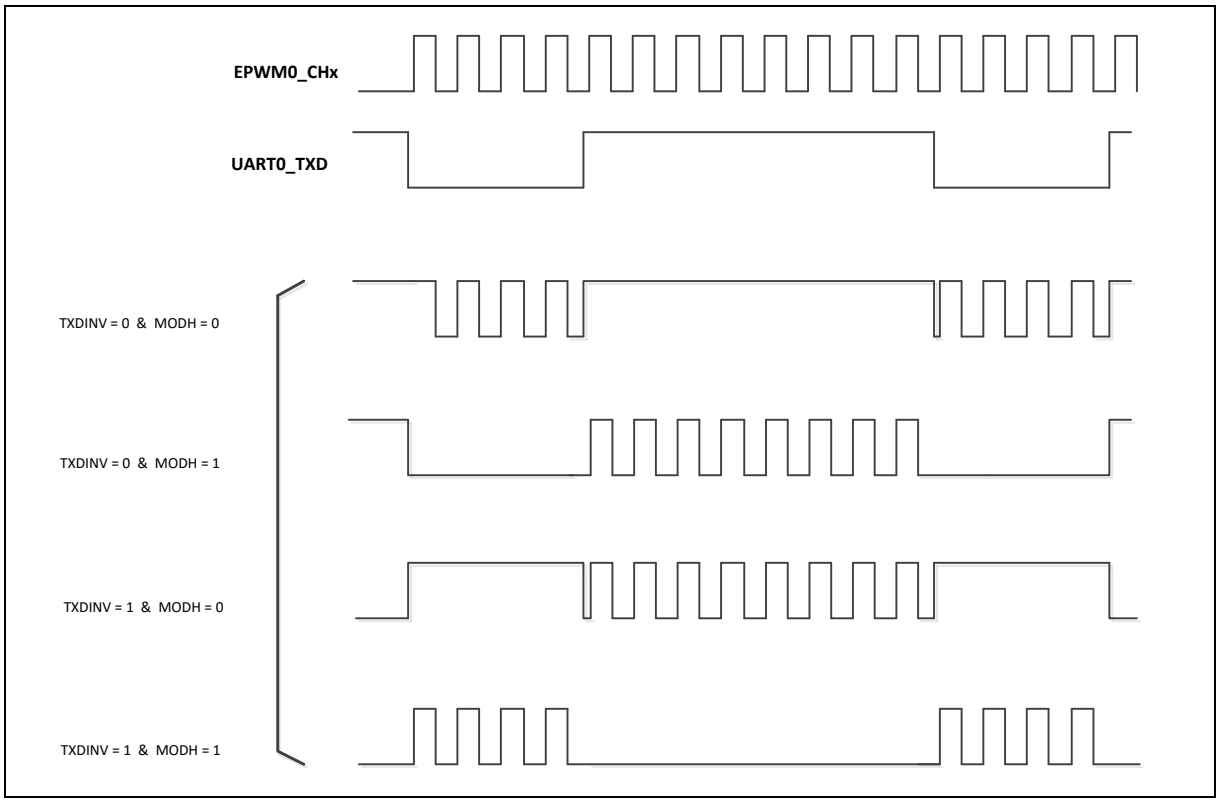


Figure 6.2-11 UART0_TXD Modulated with EPWM Channel

6.2.12 Register Lock Control

Some of the system control registers need to be protected to avoid inadvertent write and disturb the chip operation. These system control registers are protected after the power-on reset till user disables register protection. For user to program these protected registers, a register protection disable sequence needs to be followed by a special programming. The register protection disable sequence is writing the data “59h”, “16h” “88h” to the register SYS_REGLCTL address at 0x4000_0100 continuously. Any different data value, different sequence or any other write to other address during these three data writing will abort the whole sequence.

SYS_IPRST0	**Address 0x4000_0008**
SYS_ALTCTL	address 0x4000_0014
SYS_BODCTL	address 0x4000_0018
SYS_PORCTL	address 0x4000_0024
SYS_VREFCTL	address 0x4000_0028
SYS_SRAM_BISTCTL	address 0x4000_00D0
SYS_SRAM_PARITY	address 0x4000_00D8
SYS_RCADJ	address 0x4000_0110

SYS_HIRC2TCTL	address 0x4000_018C
SYS_HXTTCTL	address 0x4000_0190
SYS_ACMPTCTL	address 0x4000_0194
SYS_PORDISAN	address 0x4000_01EC
SYS_PLCTL	address 0x4000_01F8
CLK_PWRCTL	address 0x4000_0200
CLK_APBCLK0	address 0x4000_0208
CLK_CLKSEL0	address 0x4000_0210
CLK_CLKSEL1	address 0x4000_0214
CLK_PLLCTL	address 0x4000_0240
CLK_PLL2CTL	address 0x4000_0244
CLK_PLLTEST	address 0x4000_0248
CLK_PMUCTL	address 0x4000_0290
NMIEN	address 0x4000_0300
AHBMCTL	address 0x4000_0400
FMC_FTCTL	address 0x4000_5018
FMC_ICPCMD	address 0x4000_501C
FMC_ISPCTL	address 0x4000_C000
FMC_ISPTRG	address 0x4000_C010
FMC_FTCTL	address 0x4000_C018
FMC_ICPCTL	address 0x4000_C01C
FMC_ISPSTS	address 0x4000_C040
FMC_CYCCTL	address 0x4000_C04C
FMC_KPKEYTRG	address 0x4000_C05C
FMC_KPKEYSTS	address 0x4000_C060
WDT_CTL	address 0x4004_0000
WDT_ALTCTL	address 0x4004_0004
EADC_TEST	address 0x4004_3200
ACMP_TEST	address 0x4004_5FF8
DAC0_TEST	address 0x4004_7FF0
DAC1_TEST	address 0x4004_7FF8
TIMER0_CTL	address 0x4005_0000
TIMER1_CTL	address 0x4005_0100
TIMER2_CTL	address 0x4005_1000
TIMER3_CTL	address 0x4005_1100

TIMER0_PWMCTL	address 0x4005_0040
TIMER1_PWMCTL	address 0x4005_0140
TIMER2_PWMCTL	address 0x4005_1040
TIMER3_PWMCTL	address 0x4005_1140
TIMER0_PWMDTCTL	address 0x4005_0058
TIMER1_PWMDTCTL	address 0x4005_0158
TIMER2_PWMDTCTL	address 0x4005_1058
TIMER3_PWMDTCTL	address 0x4005_1158
TIMER0_PWMBRKCTL	address 0x4005_0070
TIMER1_PWMBRKCTL	address 0x4005_0170
TIMER2_PWMBRKCTL	address 0x4005_1070
TIMER3_PWMBRKCTL	address 0x4005_1170
TIMER0_PWMSWBRK	address 0x4005_007C
TIMER1_PWMSWBRK	address 0x4005_017C
TIMER2_PWMSWBRK	address 0x4005_107C
TIMER3_PWMSWBRK	address 0x4005_117C
TIMER0_PWMINTEN1	address 0x4005_0084
TIMER1_PWMINTEN1	address 0x4005_0184
TIMER2_PWMINTEN1	address 0x4005_1084
TIMER3_PWMINTEN1	address 0x4005_1184
TIMER0_PWMINTSTS1	address 0x4005_008C
TIMER1_PWMINTSTS1	address 0x4005_018C
TIMER2_PWMINTSTS1	address 0x4005_108C
TIMER3_PWMINTSTS1	address 0x4005_118C
EPWM_CTL0	address 0x4005_8000/0x4005_9000
EPWM_CTL1	address 0x4005_8000/0x4005_9000
EPWM_DTCTL0_1	address 0x4005_8070/0x4005_9070
EPWM_DTCTL2_3	address 0x4005_8074/0x4005_9074
EPWM_DTCTL4_5	address 0x4005_8078/0x4005_9078
EPWM_BRKCTL0_1	address 0x4005_80C8/0x4005_90C8
EPWM_BRKCTL2_3	address 0x4005_80CC/0x4005_90CC
EPWM_BRKCTL4_5	address 0x4005_80D0/0x4005_90D0
EPWM_SWBRK	address 0x4005_80DC/0x4005_90DC
EPWM_INTEN1	address 0x4005_80E4/0x4005_90E4
EPWM_INTSTS1	address 0x4005_80EC/0x4005_90EC

EPWM_SELFTEST	address 0x4005_8300/0x4005_9300
BPWM_CTL0	address 0x4005_A000/0x4005_B000
BPWM_SELFTEST	address 0x4005_A030/0x4005_B030
SYST_VAL	address 0xE000_E018

6.2.13 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	0x0164_7140 [1]
SYS_RSTSTS	SYS_BA+0x04	R/W	System Reset Status Register	0x0000_004b
SYS_IPRST0	SYS_BA+0x08	R/W	Peripheral Reset Control Register 0	0x0000_0000
SYS_IPRST1	SYS_BA+0x0C	R/W	Peripheral Reset Control Register 1	0x0000_0000
SYS_IPRST2	SYS_BA+0x10	R/W	Peripheral Reset Control Register 2	0x0000_0000
SYS_BODCTL	SYS_BA+0x18	R/W	Brown-out Detector Control Register	0x000X_038X
SYS_IVSCTL	SYS_BA+0x1C	R/W	Internal Voltage Source Control Register	0x0000_0000
SYS_VREFCTL	SYS_BA+0x28	R/W	V_{REF} 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	0x0000_0000
SYS_GPB_MFPH	SYS_BA+0x3C	R/W	GPIOB High Byte Multiple Function Control Register	0x0000_0000
SYS_GPC_MFPL	SYS_BA+0x40	R/W	GPIOC Low Byte Multiple Function Control Register	0x0000_0000
SYS_GPC_MFPH	SYS_BA+0x44	R/W	GPIOC High Byte Multiple Function Control Register	0x0000_0000
SYS_GPD_MFPL	SYS_BA+0x48	R/W	GPIOD Low Byte Multiple Function Control Register	0x0000_0000
SYS_GPD_MFPH	SYS_BA+0x4C	R/W	GPIOD High Byte Multiple Function Control Register	0x0000_0000
SYS_GPE_MFPL	SYS_BA+0x50	R/W	GPIOE Low Byte Multiple Function Control Register	0x0000_0000
SYS_GPE_MFPH	SYS_BA+0x54	R/W	GPIOE High Byte Multiple Function Control Register	0x0000_0000
SYS_GPF_MFPL	SYS_BA+0x58	R/W	GPIOF Low Byte Multiple Function Control Register	0x0000_00EE
SYS_GPF_MFPH	SYS_BA+0x5C	R/W	GPIOF High Byte Multiple Function Control Register	0x0000_0000
SYS_GPG_MFPL	SYS_BA+0x60	R/W	GPIOG Low Byte Multiple Function Control Register	0x0000_0000
SYS_GPG_MFPH	SYS_BA+0x64	R/W	GPIOG High Byte Multiple Function Control Register	0x0000_0000
SYS_GPH_MFPL	SYS_BA+0x68	R/W	GPIOH Low Byte Multiple Function Control Register	0x0000_0000
SYS_GPH_MFPH	SYS_BA+0x6C	R/W	GPIOH High Byte Multiple Function Control Register	0x0000_0000
SYS_GPI_MFPL	SYS_BA+0x70	R/W	GPIOI Low Byte Multiple Function Control Register	0x0000_0000

SYS_GPA_MFOS	SYS_BA+0x80	R/W	GPIOA Multiple Function Output Select Register	0x0000_0000
SYS_GPB_MFOS	SYS_BA+0x84	R/W	GPIOB Multiple Function Output Select Register	0x0000_0000
SYS_GPC_MFOS	SYS_BA+0x88	R/W	GPIOC Multiple Function Output Select Register	0x0000_0000
SYS_GPD_MFOS	SYS_BA+0x8C	R/W	GPIOD Multiple Function Output Select Register	0x0000_0000
SYS_GPE_MFOS	SYS_BA+0x90	R/W	GPIOE Multiple Function Output Select Register	0x0000_0000
SYS_GPF_MFOS	SYS_BA+0x94	R/W	GPIOF Multiple Function Output Select Register	0x0000_0000
SYS_GPG_MFOS	SYS_BA+0x98	R/W	GPIOG Multiple Function Output Select Register	0x0000_0000
SYS_GPH_MFOS	SYS_BA+0x9C	R/W	GPIOH Multiple Function Output Select Register	0x0000_0000
SYS_GPI_MFOS	SYS_BA+0xA0	R/W	GPIOI Multiple Function Output Select Register	0x0000_0000
SYS_MODCTL	SYS_BA+0xB0	R/W	Modulation Control Register	0x0000_0000
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_SRAM_BISTCTL	SYS_BA+0xD0	R/W	System SRAM BIST Test Control Register	0x0000_0000
SYS_SRAM_BISTSTS	SYS_BA+0xD4	R	System SRAM BIST Test Status Register	0x00xx_00xx
SYS_IRCTCTL	SYS_BA+0xF0	R/W	HIRC Trim Control Register	0x0008_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_PORDISAN	SYS_BA+0x1EC	R/W	Analog POR Disable Control Register	0x0000_0000
SYS_PLCTL	SYS_BA+0x1F8	R/W	Power Level Control Register	0x0000_0001
SYS_PLSTS	SYS_BA+0x1FC	R	Power Level Status Register	0x0000_0100
SYS_AHBMCTL	SYS_BA+0x400	R/W	AHB Bus Matrix Priority Control Register	0x0000_0001

6.2.14 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	0x0164_7140[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_004b

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

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**CPULKRF**	**CPU Lockup Reset Flag** 0 = No reset from CPU lockup happened. 1 = The Cortex®-M4 lockup happened and chip is reset. **Note 1:** Write 1 to clear this bit to 0. **Note 2:** When CPU lockup happened under ICE is connected, this flag will set to 1 but chip will not reset.
[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]	**HRESETRF**	**HRESET Reset Flag** The HRESET reset flag is set by the "Reset Signal" from the HRESET. 0 = No reset from HRESET. 1 = Reset from HRESET. **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 had 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 had issued the reset signal to reset the system. **Note 1:** Write 1 to clear this bit to 0. **Note 2:** Watchdog Timer register RSTF(WDT_CTL[2]) bit is set if the system has been reset by WDT time-out reset. Window Watchdog Timer register WWDTRF(WWDT_STATUS[1]) bit is set if the system has been reset by WWDT time-out reset.
[1]	**PINRF**	**NRESET Pin Reset Flag** The nRESET pin reset flag is set by the “Reset Signal” from the nRESET Pin to indicate the previous reset source. 0 = No reset from nRESET pin. 1 = Pin nRESET had issued the reset signal to reset the system. **Note:** 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 had 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.
[7]	**CRCRST**	**CRC Calculation Controller Reset (Write Protect)** 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.
[2]	**PDMARST**	**PDMA Controller Reset (Write Protect)** Setting this bit to 1 will generate a reset signal to the PDMA. User needs to set this bit to 0 to release from reset state. 0 = PDMA controller normal operation. 1 = PDMA controller reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[1]	**CPURST**	**Processor Core One-shot Reset (Write Protect)** Setting this bit will only reset the processor core and Flash Memory Controller(FMC), and this bit will automatically return to 0 after the 2 clock cycles. 0 = Processor core normal operation. 1 = Processor core one-shot reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[0]	**CHIPRST**	**Chip One-shot Reset (Write Protect)** Setting this bit will reset the whole chip, including Processor core and all peripherals, and this bit will automatically return to 0 after the 2 clock cycles. The CHIPRST is same as the POR reset, all the chip controllers is reset and the chip setting from Flash are also reload. About the difference between CHIPRST and SYSRESETREQ(AIRCR[2]), please refer to section 7.2.2 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			EADCRST	Reserved			
23	22	21	20	19	18	17	16
Reserved		UART5RST	UART4RST	UART3RST	UART2RST	UART1RST	UART0RST
15	14	13	12	11	10	9	8
Reserved	SPI1RST	SPI0RST	Reserved			I2C1RST	I2C0RST
7	6	5	4	3	2	1	0
ACMP01RST	Reserved	TMR3RST	TMR2RST	TMR1RST	TMR0RST	GPIORST	Reserved

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28]	**EADCRST**	**EADC Controller Reset** 0 = EADC controller normal operation. 1 = EADC controller reset.
[27:22]	**Reserved**	Reserved.
[21]	**UART5RST**	**UART5 Controller Reset** 0 = UART5 controller normal operation. 1 = UART5 controller reset.
[20]	**UART4RST**	**UART4 Controller Reset** 0 = UART4 controller normal operation. 1 = UART4 controller reset.
[19]	**UART3RST**	**UART3 Controller Reset** 0 = UART3 controller normal operation. 1 = UART3 controller reset.
[18]	**UART2RST**	**UART2 Controller Reset** 0 = UART2 controller normal operation. 1 = UART2 controller reset.
[17]	**UART1RST**	**UART1 Controller Reset** 0 = UART1 controller normal operation. 1 = UART1 controller reset.
[16]	**UART0RST**	**UART0 Controller Reset** 0 = UART0 controller normal operation. 1 = UART0 controller reset.

[15]	**Reserved**	Reserved.
[14]	**SPI1RST**	**SPI1 Controller Reset** 0 = SPI1 controller normal operation. 1 = SPI1 controller reset.
[13]	**SPI0RST**	**SPI0 Controller Reset** 0 = SPI0 controller normal operation. 1 = SPI0 controller reset.
[12:10]	**Reserved**	Reserved.
[9]	**I2C1RST**	**I^2C1 Controller Reset** 0 = I^2C1 controller normal operation. 1 = I^2C1 controller reset.
[8]	**I2C0RST**	**I^2C0 Controller Reset** 0 = I^2C0 controller normal operation. 1 = I^2C0 controller reset.
[7]	**ACMP01RST**	**Analog Comparator 0/1 Controller Reset** 0 = Analog Comparator 0/1 controller normal operation. 1 = Analog Comparator 0/1 controller reset.
[6]	**Reserved**	Reserved.
[5]	**TMR3RST**	**Timer3 Controller Reset** 0 = Timer3 controller normal operation. 1 = Timer3 controller reset.
[4]	**TMR2RST**	**Timer2 Controller Reset** 0 = Timer2 controller normal operation. 1 = Timer2 controller reset.
[3]	**TMR1RST**	**Timer1 Controller Reset** 0 = Timer1 controller normal operation. 1 = Timer1 controller reset.
[2]	**TMR0RST**	**Timer0 Controller Reset** 0 = Timer0 controller normal operation. 1 = Timer0 controller reset.
[1]	**GPIORST**	**GPIO Controller Reset** 0 = GPIO controller normal operation. 1 = GPIO controller reset.
[0]	**Reserved**	Reserved.

Peripheral Reset Control Register 2 (SYS_IPRST2)

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

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

31	30	29	28	27	26	25	24
Reserved							PRNGRST
23	22	21	20	19	18	17	16
Reserved				BPWM1RST	BPWM0RST	EPWM1RST	EPWM0RST
15	14	13	12	11	10	9	8
CIR0RST	Reserved		DACRST	Reserved			
7	6	5	4	3	2	1	0
Reserved							

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**PRNGRST**	**PRNG Controller Reset** 0 = PRNG controller normal operation. 1 = PRNG controller reset.
[23:20]	**Reserved**	Reserved.
[19]	**BPWM1RST**	**BPWM1 Controller Reset** 0 = BPWM1 controller normal operation. 1 = BPWM1 controller reset.
[18]	**BPWM0RST**	**BPWM0 Controller Reset** 0 = BPWM0 controller normal operation. 1 = BPWM0 controller reset.
[17]	**EPWM1RST**	**EPWM1 Controller Reset** 0 = EPWM1 controller normal operation. 1 = EPWM1 controller reset.
[16]	**EPWM0RST**	**EPWM0 Controller Reset** 0 = EPWM0 controller normal operation. 1 = EPWM0 controller reset.
[15]	**CIR0RST**	**CIR0 Controller Reset** 0 = CIR0 controller normal operation. 1 = CIR0 controller reset.
[14:13]	**Reserved**	Reserved.
[12]	**DACRST**	**DAC Controller Reset**

		0 = DAC controller normal operation. 1 = DAC controller reset.
[11:0]	**Reserved**	Reserved.

Brown-out Detector Control Register (SYS_BODCTL)

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

Register	Offset	R/W	Description	Reset Value
SYS_BODCTL	SYS_BA+0x18	R/W	Brown-out Detector Control Register	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	LVRDGSEL			Reserved	BODDGSEL		
7	6	5	4	3	2	1	0
LVREN	BODOUT	BODLPM	BODIF	BODRSTEN	Reserved		BODEN

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17:16]	**BODVL**	**Brown-out Detector Threshold Voltage Selection (Write Protect)** The default value is set by Flash controller user configuration register CBOV (CONFIG0 [22:21]). 00 = Brown-Out Detector threshold voltage is 2.4V. 01 = Brown-Out Detector threshold voltage is 2.7V. 10 = Brown-Out Detector threshold voltage is 3.7V. 11 = Brown-Out Detector threshold voltage is 4.4V. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[15]	**Reserved**	Reserved.
[14:12]	**LVRDGSEL**	**LVR Output De-glitch Time Select (Write Protect)** 000 = Without de-glitch function. 001 = 4 system clock (HCLK). 010 = 8 system clock (HCLK). 011 = 16 system clock (HCLK). 100 = 32 system clock (HCLK). 101 = 64 system clock (HCLK). 110 = 128 system clock (HCLK). 111 = 256 system clock (HCLK). **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.
[11]	**Reserved**	Reserved.

Bits	Description	
[10:8]	**BODDGSEL**	**Brown-out Detector Output De-glitch Time Select (Write Protect)** 000 = BOD output is sampled by RC10K clock. 001 = 4 system clock (HCLK). 010 = 8 system clock (HCLK). 011 = 16 system clock (HCLK). 100 = 32 system clock (HCLK). 101 = 64 system clock (HCLK). 110 = 128 system clock (HCLK). 111 = 256 system clock (HCLK). **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.
[7]	**LVREN**	**Low Voltage Reset Enable Bit (Write Protect)** The LVR function resets the chip when the input power voltage is lower than LVR circuit setting. LVR function is enabled by default. 0 = Low Voltage Reset function Disabled. 1 = Low Voltage Reset function Enabled. **Note 1:** After enabling the bit, the LVR function will be active with 100us delay for LVR output stable (default). **Note 2:** This bit is write protected. Refer to the SYS_REGLCTL register.
[6]	**BODOUT**	**Brown-out Detector Output Status** 0 = Brown-out Detector output status is 0. It means the detected voltage is higher than BODVL setting or BODEN is 0. 1 = Brown-out Detector output status is 1. It means the detected voltage is lower than BODVL setting. If the BODEN is 0, BOD function disabled, this bit always responds 0.
[5]	**BODLPM**	**Brown-out Detector Low Power Mode (Write Protect)** 0 = BOD operate in normal mode (default). 1 = BOD Low Power mode Enabled. **Note 1:** The BOD consumes about 100uA in normal mode, the low power mode can reduce the current to about 1/10 but slow the BOD response. **Note 2:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note 3:**BOD enable stable time is 50us, period enable time need larger than it
[4]	**BODIF**	**Brown-out Detector Interrupt Flag** 0 = Brown-out Detector does not detect any voltage draft at V_{DD} down through or up through the voltage of BODVL setting. 1 = When Brown-out Detector detects the V_{DD} is dropped down through the voltage of BODVL setting or the V_{DD} is raised up through the voltage of BODVL setting, this bit is set to 1 and the brown-out interrupt is requested if brown-out interrupt is enabled. **Note:** Write 1 to clear this bit to 0.

Bits	Description	
[3]	**BODRSTEN**	**Brown-out Reset Enable Bit (Write Protect)** The default value is set by Flash controller user configuration register CBORST(CONFIG0[20]) bit. 0 = Brown-out "INTERRUPT" function Enabled. 1 = Brown-out "RESET" function Enabled. **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 be kept till to the 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:** This bit is write protected. Refer to the SYS_REGLCTL register.
[2:1]	**Reserved**	Reserved.
[0]	**BODEN**	**Brown-out Detector Enable Bit (Write Protect)** 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:** This bit is write protected. Refer to the SYS_REGLCTL register.

Internal Voltage Source Control Register (SYS_IVSCTL)

Register	Offset	R/W	Description	Reset Value
SYS_IVSCTL	SYS_BA+0x1C	R/W	Internal Voltage Source Control Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**VTEMPEN**	**Temperature Sensor Enable Bit** This bit is used to enable/disable temperature sensor function. 0 = Temperature sensor function Disabled (default). 1 = Temperature sensor function Enabled.

V_{REF} Control Register (SYS_VREFCTL)

Register	Offset	R/W	Description	Reset Value
SYS_VREFCTL	SYS_BA+0x28	R/W	V_{REF} Control Register	0x0000_0000

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

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**VBGFEN**	**Chip Internal Voltage Band-gap Force Enable Bit(Write Only)** 0 = Chip internal voltage band-gap controlled by ADC/ACMP if source selected. 1 = Chip internal voltage band-gap force enable. **Note:** If user want to read the value of this bit, please read bit[25].
[23:7]	**Reserved**	Reserved.
[6]	**PRELOADEN**	**Pre-load Function Enable Bit (Write Protect)** This bit should be enabled and keep during Tstable when VREFCTL(SYS_VREFCTL[4:0]) change setting(except set to 00000). Tstable depends on different situations has different requirement, please refer to Datasheet. 0 = V_{REF} Pre-load function Disabled. (Default). 1 = V_{REF} Pre-load function Enabled. **Note 1:** This bit is write protected. Refer to the SYS_REGLCTL register.
[5]	**Reserved**	Reserved.
[4:0]	**VREFCTL**	**V_{REF} Control Bits (Write Protect)** 00000 = V_{REF} is from external pin. 00010 = V_{REF} is from internal reference voltage 2.048V. 00110 = V_{REF} is from internal reference voltage 2.56V. 01010 = V_{REF} is from internal reference voltage 3.072V. 01110 = V_{REF} is from internal reference voltage 4.096V. Others = Reserved. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

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	0x0000_0000

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

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

GPIOB High Byte Multiple Function Control Register (SYS_GPB_MFPH)

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

31	30	29	28	27	26	25	24
PB15MFP				PB14MFP			
23	22	21	20	19	18	17	16
PB13MFP				PB12MFP			
15	14	13	12	11	10	9	8
PB11MFP				PB10MFP			
7	6	5	4	3	2	1	0
PB9MFP				PB8MFP			

Bits	Description	
[31:28]	PB15MFP	PB.15 Multi-function Pin Selection
[27:24]	PB14MFP	PB.14 Multi-function Pin Selection
[23:20]	PB13MFP	PB.13 Multi-function Pin Selection
[19:16]	PB12MFP	PB.12 Multi-function Pin Selection
[15:12]	PB11MFP	PB.11 Multi-function Pin Selection
[11:8]	PB10MFP	PB.10 Multi-function Pin Selection
[7:4]	PB9MFP	PB.9 Multi-function Pin Selection
[3:0]	PB8MFP	PB.8 Multi-function Pin Selection

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
Reserved				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]	Reserved	Reserved
[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_0000

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

GPIOE Low Byte Multiple Function Control Register (SYS_GPE_MFPL)

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

31	30	29	28	27	26	25	24
PE7MFP				PE6MFP			
23	22	21	20	19	18	17	16
PE5MFP				PE4MFP			
15	14	13	12	11	10	9	8
PE3MFP				PE2MFP			
7	6	5	4	3	2	1	0
PE1MFP				PE0MFP			

Bits	Description	
[31:28]	PE7MFP	PE.7 Multi-function Pin Selection
[27:24]	PE6MFP	PE.6 Multi-function Pin Selection
[23:20]	PE5MFP	PE.5 Multi-function Pin Selection
[19:16]	PE4MFP	PE.4 Multi-function Pin Selection
[15:12]	PE3MFP	PE.3 Multi-function Pin Selection
[11:8]	PE2MFP	PE.2 Multi-function Pin Selection
[7:4]	PE1MFP	PE.1 Multi-function Pin Selection
[3:0]	PE0MFP	PE.0 Multi-function Pin Selection

GPIOE High Byte Multiple Function Control Register (SYS_GPE_MFPH)

Register	Offset	R/W	Description	Reset Value
SYS_GPE_MFPH	SYS_BA+0x54	R/W	GPIOE High Byte Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PE15MFP				PE14MFP			
23	22	21	20	19	18	17	16
PE13MFP				PE12MFP			
15	14	13	12	11	10	9	8
PE11MFP				PE10MFP			
7	6	5	4	3	2	1	0
PE9MFP				PE8MFP			

Bits	Description	
[31:28]	PE15MFP	PE.15 Multi-function Pin Selection
[27:24]	PE14MFP	PE.14 Multi-function Pin Selection
[23:20]	PE13MFP	PE.13 Multi-function Pin Selection
[19:16]	PE12MFP	PE.12 Multi-function Pin Selection
[15:12]	PE11MFP	PE.11 Multi-function Pin Selection
[11:8]	PE10MFP	PE.10 Multi-function Pin Selection
[7:4]	PE9MFP	PE.9 Multi-function Pin Selection
[3:0]	PE8MFP	PE.8 Multi-function Pin Selection

GPIOF Low Byte Multiple Function Control Register (SYS_GPF_MFPL)

Register	Offset	R/W	Description	Reset Value
SYS_GPF_MFPL	SYS_BA+0x58	R/W	GPIOF Low Byte Multiple Function Control Register	0x0000_00EE

31	30	29	28	27	26	25	24
PF7MFP				PF6MFP			
23	22	21	20	19	18	17	16
PF5MFP				PF4MFP			
15	14	13	12	11	10	9	8
PF3MFP				PF2MFP			
7	6	5	4	3	2	1	0
PF1MFP				PF0MFP			

Bits	Description	
[31:28]	PF7MFP	PF.7 Multi-function Pin Selection
[27:24]	PF6MFP	PF.6 Multi-function Pin Selection
[23:20]	PF5MFP	PF.5 Multi-function Pin Selection
[19:16]	PF4MFP	PF.4 Multi-function Pin Selection
[15:12]	PF3MFP	PF.3 Multi-function Pin Selection
[11:8]	PF2MFP	PF.2 Multi-function Pin Selection
[7:4]	PF1MFP	PF.1 Multi-function Pin Selection
[3:0]	PF0MFP	PF.0 Multi-function Pin Selection

GPIOF High Byte Multiple Function Control Register (SYS_GPF_MFPH)

Register	Offset	R/W	Description	Reset Value
SYS_GPF_MFPH	SYS_BA+0x5C	R/W	GPIOF High Byte Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PF15MFP				PF14MFP			
23	22	21	20	19	18	17	16
PF13MFP				PF12MFP			
15	14	13	12	11	10	9	8
PF11MFP				PF10MFP			
7	6	5	4	3	2	1	0
PF9MFP				PF8MFP			

Bits	Description	
[31:28]	PF15MFP	PF.15 Multi-function Pin Selection
[27:24]	PF14MFP	PF.14 Multi-function Pin Selection
[23:20]	PF13MFP	PF.13 Multi-function Pin Selection
[19:16]	PF12MFP	PF.12 Multi-function Pin Selection
[15:12]	PF11MFP	PF.11 Multi-function Pin Selection
[11:8]	PF10MFP	PF.10 Multi-function Pin Selection
[7:4]	PF9MFP	PF.9 Multi-function Pin Selection
[3:0]	PF8MFP	PF.8 Multi-function Pin Selection

GPIOG Low Byte Multiple Function Control Register (SYS_GPG_MFPL)

Register	Offset	R/W	Description	Reset Value
SYS_GPG_MFPL	SYS_BA+0x60	R/W	GPIOG Low Byte Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				PG4MFP			
15	14	13	12	11	10	9	8
PG3MFP				PG2MFP			
7	6	5	4	3	2	1	0
Reserved							

Bits	Description	
[31:20]	Reserved	Reserved.
[19:16]	PG4MFP	PG.4 Multi-function Pin Selection
[15:12]	PG3MFP	PG.3 Multi-function Pin Selection
[11:8]	PG2MFP	PG.2 Multi-function Pin Selection
[7:0]	Reserved	Reserved.

GPIOG High Byte Multiple Function Control Register (SYS_GPG_MFPH)

Register	Offset	R/W	Description	Reset Value
SYS_GPG_MFPH	SYS_BA+0x64	R/W	GPIOG High Byte Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PG15MFP				PG14MFP			
23	22	21	20	19	18	17	16
PG13MFP				PG12MFP			
15	14	13	12	11	10	9	8
PG11MFP				PG10MFP			
7	6	5	4	3	2	1	0
PG9MFP				PG8MFP			

Bits	Description	
[31:28]	PG15MFP	PG.15 Multi-function Pin Selection
[27:24]	PG14MFP	PG.14 Multi-function Pin Selection
[23:20]	PG13MFP	PG.13 Multi-function Pin Selection
[19:16]	PG12MFP	PG.12 Multi-function Pin Selection
[15:12]	PG11MFP	PG.11 Multi-function Pin Selection
[11:8]	PG10MFP	PG.10 Multi-function Pin Selection
[7:4]	PG9MFP	PG.9 Multi-function Pin Selection
[3:0]	PG8MFP	PG.8 Multi-function Pin Selection

GPIOH Low Byte Multiple Function Control Register (SYS_GPH_MFPL)

Register	Offset	R/W	Description	Reset Value
SYS_GPH_MFPL	SYS_BA+0x68	R/W	GPIOH Low Byte Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PH7MFP				PH6MFP			
23	22	21	20	19	18	17	16
PH5MFP				PH4MFP			
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							

Bits	Description	
[31:28]	PH7MFP	PH.7 Multi-function Pin Selection
[27:24]	PH6MFP	PH.6 Multi-function Pin Selection
[23:20]	PH5MFP	PH.5 Multi-function Pin Selection
[19:16]	PH4MFP	PH.4 Multi-function Pin Selection
[15:0]	Reserved	Reserved.

GPIOH High Byte Multiple Function Control Register (SYS_GPH_MFPH)

Register	Offset	R/W	Description	Reset Value
SYS_GPH_MFPH	SYS_BA+0x6C	R/W	GPIOH High Byte Multiple Function 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
PH11MFP				PH10MFP			
7	6	5	4	3	2	1	0
PH9MFP				PH8MFP			

Bits	Description	
[31:16]	Reserved	Reserved
[15:12]	PH11MFP	PH.11 Multi-function Pin Selection
[11:8]	PH10MFP	PH.10 Multi-function Pin Selection
[7:4]	PH9MFP	PH.9 Multi-function Pin Selection
[3:0]	PH8MFP	PH.8 Multi-function Pin Selection

GPIOI Low Byte Multiple Function Control Register (SYS_GPI_MFPL)

Register	Offset	R/W	Description	Reset Value
SYS_GPI_MFPL	SYS_BA+0x70	R/W	GPIOI Low Byte Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
PI5MFP				PI4MFP			
15	14	13	12	11	10	9	8
PI3MFP				PI2MFP			
7	6	5	4	3	2	1	0
PI1MFP				PI0MFP			

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:20]	**PI5MFP**	**PI.5 Multi-function Pin Selection**
[19:16]	**PI4MFP**	**PI.4 Multi-function Pin Selection**
[15:12]	**PI3MFP**	**PI.3 Multi-function Pin Selection**
[11:8]	**PI2MFP**	**PI.2 Multi-function Pin Selection**
[7:4]	**PI1MFP**	**PI.1 Multi-function Pin Selection**
[3:0]	**PI0MFP**	**PI.0 Multi-function Pin Selection**

GPIO A-H Multiple Function Output Select Register (SYS_GPx_MFOS)

Register	Offset	R/W	Description	Reset Value
SYS_GPA_MFOS	SYS_BA+0x80	R/W	GPIOA Multiple Function Output Select Register	0x0000_0000
SYS_GPB_MFOS	SYS_BA+0x84	R/W	GPIOB Multiple Function Output Select Register	0x0000_0000
SYS_GPC_MFOS	SYS_BA+0x88	R/W	GPIOC Multiple Function Output Select Register	0x0000_0000
SYS_GPD_MFOS	SYS_BA+0x8C	R/W	GPIOD Multiple Function Output Select Register	0x0000_0000
SYS_GPE_MFOS	SYS_BA+0x90	R/W	GPIOE Multiple Function Output Select Register	0x0000_0000
SYS_GPF_MFOS	SYS_BA+0x94	R/W	GPIOF Multiple Function Output Select Register	0x0000_0000
SYS_GPG_MFOS	SYS_BA+0x98	R/W	GPIOG Multiple Function Output Select Register	0x0000_0000
SYS_GPH_MFOS	SYS_BA+0x9C	R/W	GPIOH Multiple Function Output Select Register	0x0000_0000
SYS_GPI_MFOS	SYS_BA+0xA0	R/W	GPIOI Multiple Function Output 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
MFOS							
7	6	5	4	3	2	1	0
MFOS							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**MFOS**	**GPIOA-h Pin[n] Multiple Function Pin Output Mode Select** This bit used to select multiple function pin output mode type for Px.n pin 0 = Multiple function pin output mode type is Push-pull mode. 1 = Multiple function pin output mode type is Open-drain mode. **Note:** Max. n=5 for port I. The PC15/PG0/PG1/PG5/PG6/PG7/PG8/PH0/PH1/PH2/PH3/PH12/PH13/PH14/PH15 pins are not available.

Modulation Control Register (SYS_MODCTL)

Register	Offset	R/W	Description	Reset Value
SYS_MODCTL	SYS_BA+0xB0	R/W	Modulation Control Register	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:4]	**MODPWMSEL**	**EPWM0 Channel Select for Modulation** Select the EPWM0 channel to modulate with the UART0_TXD. 0000: EPWM0 Channel 0 modulate with UART0_TXD. 0001: EPWM0 Channel 1 modulate with UART0_TXD. 0010: EPWM0 Channel 2 modulate with UART0_TXD. 0011: EPWM0 Channel 3 modulete with UART0_TXD. 0100: EPWM0 Channel 4 modulete with UART0_TXD. 0101: EPWM0 Channel 5 modulete with UART0_TXD. 0110: Reserved. 0111: Reserved. 1000: Reserved. 1001: Reserved. 1010: Reserved. 1011: Reserved. 1100: Reserved. 1101: Reserved. 1110: Reserved. 1111: Reserved. **Note:** This bis is valid while MODEN (SYS_MODCTL[0]) is set to 1.
[3:2]	**Reserved**	Reserved.
[1]	**MODH**	**Modulation at Data High** Select modulation pulse(EPWM0) at high or low of UART0_TXD. 0: Modulation pulse at UART0_TXD low. 1: Modulation pulse at UART0_TXD high.
[0]	**MODEN**	**Modulation Function Enable Bit** This bit enables modulation funcion by modulating with EPWM0 channel output and

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

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.
[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.
[0]	PERRIF	**SRAM Parity Check Error Flag** This bit indicates the System SRAM parity error occurred. Write 1 to clear this 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.

System SRAM BIST Test Control Register (SYS_SRAM_BISTCTL)

Register	Offset	R/W	Description	Reset Value
SYS_SRAM_BISTCTL	SYS_BA+0xD0	R/W	System SRAM BIST Test 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
PDMABIST	Reserved				CRBIST	SRBIST1	SRBIST0

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**PDMABIST**	**PDMA BIST Enable Bit (Write Protect)** This bit enables BIST test for PDMA RAM 0 = system PDMA BIST Disabled. 1 = system PDMA BIST Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[6:3]	**Reserved**	Reserved.
[2]	**CRBIST**	**CACHE BIST Enable Bit (Write Protect)** This bit enables BIST test for CACHE RAM 0 = system CACHE BIST Disabled. 1 = system CACHE BIST Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[1]	**SRBIST1**	**SRAM Bank1 BIST Enable Bit (Write Protect)** This bit enables BIST test for SRAM bank1. 0 = system SRAM bank1 BIST Disabled. 1 = system SRAM bank1 BIST Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[0]	**SRBIST0**	**SRAM Bank0 BIST Enable Bit (Write Protect)** This bit enables BIST test for SRAM bank0. 0 = system SRAM bank0 BIST Disabled. 1 = system SRAM bank0 BIST Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

System SRAM BIST Test Status Register (SYS_SRAM_BISTSTS)

Register	Offset	R/W	Description	Reset Value
SYS_SRAM_BISTSTS	SYS_BA+0xD4	R	System SRAM BIST Test Status Register	0x00xx_00xx

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
PDMABEND	Reserved				CRBEND	SRBEND1	SRBEND0
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PDMABEF	Reserved				CRBISTEF	SRBISTEF1	SRBISTEF0

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23]	**PDMABEND**	**PDMA SRAM BIST Test Finish** 0 = PDMA SRAM BIST is active. 1 = PDMA SRAM BIST test finish.
[22:19]	**Reserved**	Reserved.
[18]	**CRBEND**	**CACHE SRAM BIST Test Finish** 0 = System CACHE RAM BIST is active. 1 = System CACHE RAM BIST test finish.
[17]	**SRBEND1**	**2nd SRAM BIST Test Finish** 0 = 2nd system SRAM BIST is active. 1 = 2nd system SRAM BIST finish.
[16]	**SRBEND0**	**1st SRAM BIST Test Finish** 0 = 1st system SRAM BIST active. 1 =1st system SRAM BIST finish.
[15:8]	**Reserved**	Reserved.
[7]	**PDMABEF**	**PDMA SRAM BIST Fail Flag** 0 = PDMA SRAM BIST test pass. 1 = PDMA SRAM BIST test fail.
[6:3]	**Reserved**	Reserved.
[2]	**CRBISTEF**	**CACHE SRAM BIST Fail Flag** 0 = System CACHE RAM BIST test pass. 1 = System CACHE RAM BIST test fail.
[1]	**SRBISTEF1**	**2nd System SRAM BIST Fail Flag** 0 = 2nd system SRAM BIST test pass.

		1 = 2nd system SRAM BIST test fail.
[0]	**SRBISTEF0**	**1st System SRAM BIST Fail Flag** 0 = 1st system SRAM BIST test pass. 1 = 1st system SRAM BIST test fail.

HIRC Trim Control Register (SYS_IRCTCTL)

Register	Offset	R/W	Description	Reset Value
SYS_IRCTCTL	SYS_BA+0xF0	R/W	HIRC Trim Control Register	0x0008_0000

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

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20:16]	**BOUNDARY**	**Boundary Selection** Fill the boundary range from 0x1 to 0x31, 0x0 is reserved. This field shows the update value range. If the difference between the current trim value and the new update trim value is smaller than BOUNDARY value, then the trim value will be update to RC or it will keep the current trim value. **Note 1:** When the RC clock shift over 0.25% due to temperature condition in lock state, rc_trim circuit will increase or decrease at least 2 to trim value. **Note 2:** Only when the difference of the current frequency count and the target frequency count is smaller than 0.25% in unlock state, rc_trim circuit will increase or decrease 1 to trim value. **Note**: This field is effective only when the BOUNDEN(SYS_IRCTCTL[9]) is enabled.
[15:11]	**Reserved**	Reserved.
[10]	**REFCKSEL**	**Reference Clock Selection** 0 = HIRC trim reference clock is from LXT (32.768 kHz). 1 = Reserved. **Note:** The HIRC trim reference clock is 20 kHz in test mode.
[9]	**BOUNDEN**	**Boundary Enable Bit** 0 = Boundary function Disabled. 1 = Boundary function Enabled.
[8]	**CESTOPEN**	**Clock Error Stop Enable Bit** 0 = The trim operation keeps going if clock is inaccurate. 1 = The trim operation is stopped if clock is inaccurate.
[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 is locked, the internal trim value update counter will be reset.

		If the trim value update counter reaches this limitation value and the frequency of HIRC is still not locked, 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.
[1:0]	**FREQSEL**	**Trim Frequency Selection** This field indicates the target frequency of 48 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. 00 = Disable HIRC auto trim function. 01 = Enable HIRC auto trim function and trim HIRC to 48 MHz. 10 = Reserved. 11 = Reserved. **Note:** Only when the LXT and HIRC is stable, then the FREQSEL setting will be load to HIRC auto trim circuit.

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.
[2]	**CLKEIEN**	**Clock Error Interrupt Enable Bit** This bit controls if CPU would get an interrupt while clock is inaccurate 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 inaccurate. 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.

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				OVBDIF	CLKERRIF	TFAILIF	FREQLOCK

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**OVBDIF**	**Over Boundary Status** When the over boundary function is set, if there occurs the over boundary condition, this flag will be set. 0 = Over boundary coundition did not occur. 1 = Over boundary coundition occurred. **Note 1:** Write 1 to clear this flag.
[2]	**CLKERRIF**	**Clock Error Interrupt Status** When the frequency of 32.768 kHz external low speed crystal oscillator (LXT) or 48 MHz internal high speed RC oscillator (HIRC) is shift larger to unreasonable value, this bit will be set and to be an indicate that clock frequency is inaccurate. Once this bit is set to 1, the auto trim operation stopped and FREQSEL(SYS_IRCTCTL[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 inaccurate. Write 1 to clear this to 0. 0 = Clock frequency is accurate. 1 = Clock frequency is inaccurate.
[1]	**TFAILIF**	**Trim Failure Interrupt Status** This bit indicates that HIRC trim value update limitation count is reached and the HIRC clock frequency is still not 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 is not reached. 1 = Trim value update limitation count is reached and HIRC frequency is still not locked.
[0]	**FREQLOCK**	**HIRC Frequency Lock Status** This bit indicates the HIRC frequency is locked. This is a status bit and does not trigger any interrupt Write 1 to clear this to 0. This bit will be set automatically, if the frequency is locked and the RC_TRIM is enabled.

		0 = The internal high-speed oscillator frequency is not locked at 48 MHz yet. 1 = The internal high-speed oscillator frequency is locked at 48 MHz.

Register Lock Control Register (SYS_REGLCTL)

Some of the system control registers need to be protected to avoid inadvertent write and disturb the chip operation. These system control registers are protected after the power-on reset till user disables register protection. For user to program these protected registers, a register protection disable sequence needs to be followed by a special programming. The register protection disable sequence is writing the data “59h”, “16h” “88h” to the register SYS_REGLCTL address at 0x4000_0100 continuously. Any different data value, different sequence or any other write to other address during these three data writing will abort the whole sequence.

After the protection is disabled, user can check the protection disable bit at address 0x4000_0100 bit0, 1 is protection disable, and 0 is protection enable. Then user can update the target protected register value and then write any data to the address “0x4000_0100” to enable register protection.

This register is written to disable/enable register protection and read for the REGLCTL status.

Register	Offset	R/W	Description	Reset Value
SYS_REGLCTL	SYS_BA+0x100	R/W	Register Lock Control Register	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**REGLCTL**	**Register Lock Control Code (Write Only)** Some registers have write-protection function. Writing these registers have to disable the protected function by writing the sequence value “59h”, “16h”, “88h” to this field. After this sequence is completed, the REGLCTL bit will be set to 1 and write-protection registers can be normal write. **REGLCTL[0]** **Register Lock Control Disable Index (Read Only)** 0 = Write-protection Enabled for writing protected registers. Any write to the protected register is ignored. 1 = Write-protection Disabled for writing protected registers.

Analog POR Disable Control Register (SYS_PORDISAN)

Register	Offset	R/W	Description	Reset Value
SYS_PORDISAN	SYS_BA+0x1EC	R/W	Analog POR Disable 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
POROFFAN							
7	6	5	4	3	2	1	0
POROFFAN							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**POROFFAN**	**Power-on Reset Enable Bit (Write Protect)** After powered on, user can turn off internal analog POR circuit to save power by writing 0x5AA5 to this field. The analog POR circuit 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.

Power Level Control Register (SYS_PLCTL)

Register	Offset	R/W	Description	Reset Value
SYS_PLCTL	SYS_BA+0x1F8	R/W	Power Level Control Register	0x0000_0001

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

Bits	Description	
[31:24]	**LVSPRD**	**LDO Voltage Scaling Period(Write Protect)** The LVSPRD value is the period of each LDO voltage rising step. LDO voltage scaling period = (LVSPRD + 1) * 1us.
[23:21]	**Reserved**	Reserved.
[20:16]	**LVSSTEP**	**LDO Voltage Scaling Step(Write Protect)** The LVSSTEP value is LDO voltage rising step. LDO voltage scaling step = (LVSSTEP + 1) * 20mV.
[15:2]	**Reserved**	Reserved.
[1:0]	**PLSEL**	**Power Level Select(Write Protect)** These bits indicate the status of power level. 00 = Power level is PL0. 01 = Power level is PL1. Others = Reserved. **Note:** Refer to section 6.2.5 for Power Modes and Power Level Transition.

Power Level Status Register (SYS_PLSTS)

Register	Offset	R/W	Description	Reset Value
SYS_PLSTS	SYS_BA+0x1FC	R	Power Level Status Register	0x0000_0100

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

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:8]	**PLSTATUS**	**Power Level Status (Read Only)** This bit indicates the status of power level. 00 = Power level is PL0. 01 = Power level is PL1. Others = Reserved.
[7:1]	**Reserved**	Reserved.
[0]	**PLCBUSY**	**Power Level Change Busy Bit (Read Only)** This bit is set by hardware when power level is changing. After power level change is completed, this bit will be cleared automatically by hardware. 0 = Core voltage change is completed. 1 = Core voltage change is ongoing.

AHB Bus Matrix Priority Control Register (SYS_AHBMCTL)

Register	Offset	R/W	Description	Reset Value
SYS_AHBMCTL	SYS_BA+0x400	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.
[0]	**INTACTEN**	**Highest AHB Bus Priority of Cortex®-M4 Core Enable Bit (Write Protect)** 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 occurred. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

6.2.15 System Timer (SysTick)

The Cortex®-M4 includes an integrated system timer, SysTick, which provides a simple, 24-bit clear-on-write, decrementing, wrap-on-zero counter with a flexible control mechanism. The counter can be used as a Real Time Operating System (RTOS) tick timer or as a simple counter.

When system timer is enabled, it will count down from the value in the SysTick Current Value Register (SYST_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 transitions to zero, the COUNTFLAG status bit is set. The COUNTFLAG bit clears on reads.

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*".

6.2.16 System Timer Control Register Map

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

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

6.2.16.1 System Timer Control Register Description

SysTick Control and Status Register (SYST_CTRL)

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

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

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

SysTick Reload Value Register (SYST_LOAD)

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

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**RELOAD**	**System Tick Reload Value** Value to load into the Current Value register when the counter reaches 0.

SysTick Current Value Register (SYST_VAL)

Register	Offset	R/W	Description	Reset Value
SYST_VAL	SCS_BA+0x18	R/W	SysTick Current Value Register	0xXXXX_XXXX

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

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

6.2.17 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.

6.2.17.1 Exception Model and System Interrupt Map

Table 6.2-8 lists the exception model supported by M471V/M471K/M471C 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 6.2-8 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	FMC_INT	FMC ISP interrupt
22	6	RTC_INT	Real time clock interrupt
23	7	TAMP_INT	Tamper interrupt
24	8	WDT_INT	Watchdog Timer interrupt
25	9	WWDT_INT	Window Watchdog Timer interrupt
26	10	EINT0	External interrupt from PA.6 or PB.5 pins
27	11	EINT1	External interrupt from PA.7, PB.4 or PD.15pins
28	12	EINT2	External interrupt from PB.3 or PC.6 pin
29	13	EINT3	External interrupt from PB.2 or PC.7 pin
30	14	EINT4	External interrupt from PA.8, PB.6 or PF.15 pin
31	15	EINT5	External interrupt from PB.7, PD.12 or PF.14 pin
32	16	GPA_INT	External interrupt from PA[15:0] pin
33	17	GPB_INT	External interrupt from PB[15:0] pin
34	18	GPC_INT	External interrupt from PC[15:0] pin
35	19	GPD_INT	External interrupt from PD[15:0] pin
36	20	GPE_INT	External interrupt from PE[15:0] pin
37	21	GPF_INT	External interrupt from PF[15:0] pin

38	22	Reserved	Reserved
39	23	SPI0_INT	SPI0 interrupt
40	24	BRAKE0_INT	EPWM0 brake interrupt
41	25	EPWM0_P0_INT	EPWM0 pair 0 interrupt
42	26	EPWM0_P1_INT	EPWM0 pair 1 interrupt
43	27	EPWM0_P2_INT	EPWM0 pair 2 interrupt
44	28	BRAKE1_INT	EPWM1 brake interrupt
45	29	EPWM1_P0_INT	EPWM1 pair 0 interrupt
46	30	EPWM1_P1_INT	EPWM1 pair 1 interrupt
47	31	EPWM1_P2_INT	EPWM1 pair 2 interrupt
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	UART1_INT	UART1 interrupt
54	38	I2C0_INT	I^2C0 interrupt
55	39	I2C1_INT	I^2C1 interrupt
56	40	PDMA_INT	PDMA interrupt
57	41	DAC_INT	DAC interrupt
58	42	EADC0_INT0	EADC0 interrupt source 0
59	43	EADC0_INT1	EADC0 interrupt source 1
60	44	ACMP01_INT	ACMP0 and ACMP1 interrupt
61	45	Reserved	Reserved
62	46	EADC0_INT2	EADC0 interrupt source 2
63	47	EADC0_INT3	EADC0 interrupt source 3
64	48	UART2_INT	UART2 interrupt
65	49	UART3_INT	UART3 interrupt
66	50	Reserved	Reserved
67	51	SPI1_INT	SPI1 interrupt
68	52	Reserved	Reserved
69	53	Reserved	Reserved
70	54	Reserved	Reserved
71	55	Reserved	Reserved
72	56	Reserved	Reserved

73	57	Reserved	Reserved
74	58	Reserved	Reserved
75	59	Reserved	Reserved
76	60	Reserved	Reserved
77	61	Reserved	Reserved
78	62	Reserved	Reserved
80	64	Reserved	Reserved
81	65	Reserved	Reserved
82	66	Reserved	Reserved
83	67	Reserved	Reserved
84	68	Reserved	Reserved
85	69	Reserved	Reserved
86	70	Reserved	Reserved
87	71	PRNG_INT	PRNG interrupt
88	72	GPG_INT	External interrupt from PG[15:0] pin
89	73	EINT6	External interrupt from PB.8 or PD.11 pin
90	74	UART4_INT	UART4 interrupt
91	75	UART5_INT	UART5 interrupt
92	76	Reserved	Reserved
93	77	Reserved	Reserved
94	78	BPWM0_INT	BPWM0 interrupt
95	79	BPWM1_INT	BPWM1 interrupt
96	80	Reserved	Reserved
97	81	Reserved	Reserved
98	82	Reserved	Reserved
99	83	Reserved	Reserved
100	84	Reserved	Reserved
101	85	Reserved	Reserved
102	86	Reserved	Reserved
103	87	Reserved	Reserved
105	88	GPH_INT	External interrupt from PH[11:0] pin
105	89	EINT7	External interrupt from PB.9 or PD.10 pin
106	90	Reserved	Reserved
107	91	Reserved	Reserved
108	92	Reserved	Reserved

109	93	Reserved	Reserved
110	94	Reserved	Reserved
111	95	Reserved	Reserved
112	96	Reserved	Reserved
113	97	Reserved	Reserved
114	98	Reserved	Reserved
115	99	Reserved	Reserved
116	100	Reserved	Reserved
117	101	Reserved	Reserved
118	102	Reserved	Reserved
119	103	Reserved	Reserved
120	104	Reserved	Reserved
121	105	Reserved	Reserved
122	106	Reserved	Reserved
123	107	Reserved	Reserved
124	108	Reserved	Reserved
125	109	Reserved	Reserved
126	110	GPI_INT	External interrupt from PI[5:0] pin
127	111	CIR0_INT	CIR0 interrupt

Table 6.2-9 Interrupt Number Table

6.2.17.2 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.

6.2.17.3 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	NVIC_BA+0x00	R/W	IRQ0 ~ IRQ111 Set-enable Control Register	0x0000_0000
NVIC_ISER1	NVIC_BA+0x04	R/W	IRQ0 ~ IRQ111 Set-enable Control Register	0x0000_0000
NVIC_ISER2	NVIC_BA+0x08	R/W	IRQ0 ~ IRQ111 Set-enable Control Register	0x0000_0000
NVIC_ISER3	NVIC_BA+0x0C	R/W	IRQ0 ~ IRQ111 Set-enable Control Register	0x0000_0000
NVIC_ICER0	NVIC_BA+0x80	R/W	IRQ0 ~ IRQ111 Clear-enable Control Register	0x0000_0000
NVIC_ICER1	NVIC_BA+0x84	R/W	IRQ0 ~ IRQ111 Clear-enable Control Register	0x0000_0000
NVIC_ICER2	NVIC_BA+0x88	R/W	IRQ0 ~ IRQ111 Clear-enable Control Register	0x0000_0000
NVIC_ICER3	NVIC_BA+0x8C	R/W	IRQ0 ~ IRQ111 Clear-enable Control Register	0x0000_0000
NVIC_ISPR0	NVIC_BA+0x100	R/W	IRQ0 ~ IRQ111 Set-pending Control Register	0x0000_0000
NVIC_ISPR1	NVIC_BA+0x104	R/W	IRQ0 ~ IRQ111 Set-pending Control Register	0x0000_0000
NVIC_ISPR2	NVIC_BA+0x108	R/W	IRQ0 ~ IRQ111 Set-pending Control Register	0x0000_0000
NVIC_ISPR3	NVIC_BA+0x10C	R/W	IRQ0 ~ IRQ111 Set-pending Control Register	0x0000_0000
NVIC_ICPR0	NVIC_BA+0x180	R/W	IRQ0 ~ IRQ111 Clear-pending Control Register	0x0000_0000
NVIC_ICPR1	NVIC_BA+0x184	R/W	IRQ0 ~ IRQ111 Clear-pending Control Register	0x0000_0000
NVIC_ICPR2	NVIC_BA+0x188	R/W	IRQ0 ~ IRQ111 Clear-pending Control Register	0x0000_0000
NVIC_ICPR3	NVIC_BA+0x18C	R/W	IRQ0 ~ IRQ111 Clear-pending Control Register	0x0000_0000
NVIC_IABR0	NVIC_BA+0x200	R/W	IRQ0 ~ IRQ111 Active Bit Register	0x0000_0000
NVIC_IABR1	NVIC_BA+0x204	R/W	IRQ0 ~ IRQ111 Active Bit Register	0x0000_0000
NVIC_IABR2	NVIC_BA+0x208	R/W	IRQ0 ~ IRQ111 Active Bit Register	0x0000_0000
NVIC_IABR3	NVIC_BA+0x20C	R/W	IRQ0 ~ IRQ111 Active Bit Register	0x0000_0000
NVIC_IPRn **n=0,1..28**	NVIC_BA+0x300 +0x4*n	R/W	IRQ0 ~ IRQ111 Priority Control Register	0x0000_0000
STIR	NVIC_BA+0xE00	R/W	Software Trigger Interrupt Registers	0x0000_0000

IRQ0 ~ IRQ111 Set-enable Control Register (NVIC_ISER0)

Register	Offset	R/W	Description	Reset Value
NVIC_ISER0	NVIC_BA+0x00	R/W	IRQ0 ~ IRQ111 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_ISER0 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 ~ IRQ111 Set-enable Control Register (NVIC_ISER1)

Register	Offset	R/W	Description	Reset Value
NVIC_ISER1	NVIC_BA+0x04	R/W	IRQ0 ~ IRQ111 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_ISER1 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 ~ IRQ111 Set-enable Control Register (NVIC_ISER2)

Register	Offset	R/W	Description	Reset Value
NVIC_ISER2	NVIC_BA+0x08	R/W	IRQ0 ~ IRQ111 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 ~ IRQ111 Set-enable Control Register (NVIC_ISER3)

Register	Offset	R/W	Description	Reset Value
NVIC_ISER3	NVIC_BA+0x0C	R/W	IRQ0 ~ IRQ111 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_ISER3 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 ~ IRQ111 Clear-enable Control Register (NVIC_ICER0)

Register	Offset	R/W	Description	Reset Value
NVIC_ICER0	NVIC_BA+0x80	R/W	IRQ0 ~ IRQ111 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_ICER0 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 ~ IRQ111 Clear-enable Control Register (NVIC_ICER1)

Register	Offset	R/W	Description	Reset Value
NVIC_ICER1	NVIC_BA+0x84	R/W	IRQ0 ~ IRQ111 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_ICER1 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 ~ IRQ111 Clear-enable Control Register (NVIC_ICER2)

Register	Offset	R/W	Description	Reset Value
NVIC_ICER2	NVIC_BA+0x88	R/W	IRQ0 ~ IRQ111 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 ~ IRQ111 Clear-enable Control Register (NVIC_ICER3)

Register	Offset	R/W	Description	Reset Value
NVIC_ICER3	NVIC_BA+0x8C	R/W	IRQ0 ~ IRQ111 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_ICER3 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 ~ IRQ111 Set-pending Control Register (NVIC_ISPR0)

Register	Offset	R/W	Description	Reset Value
NVIC_ISPR0	NVIC_BA+0x100	R/W	IRQ0 ~ IRQ111 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_ISPR0 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 ~ IRQ111 Set-pending Control Register (NVIC_ISPR1)

Register	Offset	R/W	Description	Reset Value
NVIC_ISPR1	NVIC_BA+0x104	R/W	IRQ0 ~ IRQ111 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_ISPR1 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 ~ IRQ111 Set-pending Control Register (NVIC_ISPR2)

Register	Offset	R/W	Description	Reset Value
NVIC_ISPR2	NVIC_BA+0x108	R/W	IRQ0 ~ IRQ111 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 ~ IRQ111 Set-pending Control Register (NVIC_ISPR3)

Register	Offset	R/W	Description	Reset Value
NVIC_ISPR3	NVIC_BA+0x10C	R/W	IRQ0 ~ IRQ111 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_ISPR3 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 ~ IRQ111 Clear-pending Control Register (NVIC_ICPR0)

Register	Offset	R/W	Description	Reset Value
NVIC_ICPR0	NVIC_BA+0x180	R/W	IRQ0 ~ IRQ111 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_ICPR0 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 ~ IRQ111 Clear-pending Control Register (NVIC_ICPR1)

Register	Offset	R/W	Description	Reset Value
NVIC_ICPR1	NVIC_BA+0x184	R/W	IRQ0 ~ IRQ111 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_ICPR1 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 ~ IRQ111 Clear-pending Control Register (NVIC_ICPR2)

Register	Offset	R/W	Description	Reset Value
NVIC_ICPR2	NVIC_BA+0x188	R/W	IRQ0 ~ IRQ111 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 ~ IRQ111 Clear-pending Control Register (NVIC_ICPR3)

Register	Offset	R/W	Description	Reset Value
NVIC_ICPR3	NVIC_BA+0x18C	R/W	IRQ0 ~ IRQ111 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_ICPR3 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 ~ IRQ111 Active Bit Register (NVIC_IABR0)

Register	Offset	R/W	Description	Reset Value
NVIC_IABR0	NVIC_BA+0x200	R/W	IRQ0 ~ IRQ111 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_IABR0 registers indicate which interrupts are active. 0 = interrupt not active. 1 = interrupt active.

IRQ0 ~ IRQ111 Active Bit Register (NVIC_IABR1)

Register	Offset	R/W	Description	Reset Value
NVIC_IABR1	NVIC_BA+0x204	R/W	IRQ0 ~ IRQ111 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_IABR1 registers indicate which interrupts are active. 0 = interrupt not active. 1 = interrupt active.

IRQ0 ~ IRQ111 Active Bit Register (NVIC_IABR2)

Register	Offset	R/W	Description	Reset Value
NVIC_IABR2	NVIC_BA+0x208	R/W	IRQ0 ~ IRQ111 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 ~ IRQ111 Active Bit Register (NVIC_IABR3)

Register	Offset	R/W	Description	Reset Value
NVIC_IABR3	NVIC_BA+0x20C	R/W	IRQ0 ~ IRQ111 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_IABR3 registers indicate which interrupts are active. 0 = interrupt not active. 1 = interrupt active.

IRQ0 ~ IRQ111 Interrupt Priority Register (NVIC_IPRn)

Register	Offset	R/W	Description	Reset Value
NVIC_IPRn **n=0,1..28**	NVIC_BA+0x300 +0x4*n	R/W	IRQ0 ~ IRQ111 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.
[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.
[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.
[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.

Software Trigger Interrupt Register (STIR)

Register	Offset	R/W	Description	Reset Value
STIR	NVIC_BA+0xE00	R/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.
[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.

6.2.17.4 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**
UART1_INT	**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	
[31:16]	**Reserved**	Reserved.
[15]	**UART1_INT**	**UART1 NMI Source Enable (Write Protect)** 0 = UART1 NMI source Disabled. 1 = UART1 NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[14]	**UART0_INT**	**UART0 NMI Source Enable (Write Protect)** 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 From PB.7, PD.12 or PF.14 Pin NMI Source Enable (Write Protect)** 0 = External interrupt from PB.7, PD.12 or PF.14 pin NMI source Disabled. 1 = External interrupt from PB.7, PD.12 or PF.14 pin NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[12]	**EINT4**	**External Interrupt From PA.8, PB.6 or PF.15 Pin NMI Source Enable (Write Protect)** 0 = External interrupt from PA.8, PB.6 or PF.15 pin NMI source Disabled. 1 = External interrupt from PA.8, PB.6 or PF.15 pin NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[11]	**EINT3**	**External Interrupt From PB.2 or PC.7 Pin NMI Source Enable (Write Protect)** 0 = External interrupt from PB.2 or PC.7 pin NMI source Disabled. 1 = External interrupt from PB.2 or PC.7 pin NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[10]	**EINT2**	**External Interrupt From PB.3 or PC.6 Pin NMI Source Enable (Write Protect)** 0 = External interrupt from PB.3 or PC.6 pin NMI source Disabled. 1 = External interrupt from PB.3 or PC.6 pin NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[9]	**EINT1**	**External Interrupt From PA.7, PB.4 or PD.15 NMI Source Enable (Write Protect)**

		0 = External interrupt from PA.7, PB.4 or PD.15 pin NMI source Disabled. 1 = External interrupt from PA.7, PB.4 or PD.15 pin NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[8]	**EINT0**	**External Interrupt From PA.6 or PB.5 Pin NMI Source Enable (Write Protect)** 0 = External interrupt from PA.6 or PB.5 pin NMI source Disabled. 1 = External interrupt from PA.6 or PB.5 pin NMI source Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[7]	**Reserved**	Reserved.
[6]	**RTC_INT**	**RTC NMI Source Enable (Write Protect)** 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.
[4]	**CLKFAIL**	**Clock Fail Detected NMI Source Enable (Write Protect)** 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 ParityCheck Error NMI Source Enable (Write Protect)** 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 Protect)** 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 Protect)** 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 Protect)** 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
UART1_INT	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	
[31:16]	**Reserved**	Reserved.
[15]	**UART1_INT**	**UART1 Interrupt Flag (Read Only)** 0 = UART1 interrupt is deasserted. 1 = UART1 interrupt is asserted.
[14]	**UART0_INT**	**UART0 Interrupt Flag (Read Only)** 0 = UART0 interrupt is deasserted. 1 = UART0 interrupt is asserted.
[13]	**EINT5**	**External Interrupt From PB.7, PD.12 or PF.14 Pin Interrupt Flag (Read Only)** 0 = External Interrupt from PB.7, PD.12 or PF.14 interrupt is deasserted. 1 = External Interrupt from PB.7, PD.12 or PF.14 interrupt is asserted.
[12]	**EINT4**	**External Interrupt From PA.8, PB.6 or PF.15 Pin Interrupt Flag (Read Only)** 0 = External Interrupt from PA.8, PB.6 or PF.15 interrupt is deasserted. 1 = External Interrupt from PA.8, PB.6 or PF.15 interrupt is asserted.
[11]	**EINT3**	**External Interrupt From PB.2 or PC.7 Pin Interrupt Flag (Read Only)** 0 = External Interrupt from PB.2 or PC.7 interrupt is deasserted. 1 = External Interrupt from PB.2 or PC.7 interrupt is asserted.
[10]	**EINT2**	**External Interrupt From PB.3 or PC.6 Pin Interrupt Flag (Read Only)** 0 = External Interrupt from PB.3 or PC.6 interrupt is deasserted. 1 = External Interrupt from PB.3 or PC.6 interrupt is asserted.
[9]	**EINT1**	**External Interrupt From PA.7, PB.4 or PD.15 Pin Interrupt Flag (Read Only)** 0 = External Interrupt from PA.7, PB.4 or PD.15 interrupt is deasserted. 1 = External Interrupt from PA.7, PB.4 or PD.15 interrupt is asserted.
[8]	**EINT0**	**External Interrupt From PA.6 or PB.5 Pin Interrupt Flag (Read Only)** 0 = External Interrupt from PA.6 or PB.5 interrupt is deasserted. 1 = External Interrupt from PA.6 or PB.5 interrupt is asserted.

[7]	**Reserved**	Reserved.
[6]	**RTC_INT**	**RTC Interrupt Flag (Read Only)** 0 = RTC interrupt is deasserted. 1 = RTC interrupt is asserted.
[5]	**Reserved**	Reserved.
[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 ParityCheck 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.

6.2.17.5 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.
[0]	**INTACTEN**	**Highest AHB Bus Priority of Cortex®-M4 Core Enable Bit (Write Protect)** Enable Cortex®-M4 Core With Highest AHB Bus Priority In AHB Bus Matrix 0 = Run robin mode. 1 = Cortex®-M4 CPU with highest bus priority when interrupt occusr. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

6.2.18 System Control Register

The Cortex®-M4 status and operation mode control are managed by System Control Registers. Including CPUID, Cortex®-M4 interrupt priority and Cortex®-M4 power management can be controlled through these system control registers.

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

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 = Change NMI exception state to pending. Read Operation: 0 = NMI exception is not pending. 1 = NMI exception is pending. **Note:** Because NMI is the highest-priority exception, normally the processor enters the NMI exception handler as soon as it detects a write of 1 to this bit. Entering the handler then clears this bit to 0. This means a read of this bit by the NMI exception handler returns 1 only if the NMI signal is reasserted while the processor is executing that handler.
[30:29]	Reserved	Reserved.
[28]	PENDSVSET	**PendSV Set-pending Bit** Write Operation: 0 = No effect. 1 = Change 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 = Remove 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 = Change 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 = Remove 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.
[23]	**ISRPREEMPT**	**Interrupt Preempt Bit (Read Only)** If set, a pending exception will be serviced on exit from the debug halt state.
[22]	**ISRPENDING**	**Interrupt Pending Flag, Excluding NMI and Faults (Read Only)** 0 = Interrupt not pending. 1 = Interrupt pending.
[21:18]	**Reserved**	Reserved.
[17:12]	**VECTPENDING**	**Number of the Highest Pended Exception** Indicates the Exception Number of the Highest Priority Pending Enabled Exception 0 = No pending exceptions. Non-zero = 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** Indicates 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.
[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.
[10:8]	**PRIGROUP**	**Interrupt Priority Grouping** This field determines the Split Of Group priority from subpriority,
[7:3]	**Reserved**	Reserved.
[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.

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 6.2-10 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.
[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.
[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 = Enter 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.

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.
[23:20]	**PRI_6**	Priority of system handler 6, UsageFault
[20:16]	**Reserved**	Reserved.
[15:12]	**PRI_5**	Priority of system handler 5, BusFault
[11:8]	**Reserved**	Reserved.
[7:4]	**PRI_4**	Priority of system handler 4, MemManage
[3:0]	**Reserved**	Reserved.

System Handler Priority Register 2 (SHPR2)

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

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

Bits	Description	
[31:28]	**PRI_11**	**Priority of System Handler 11 – SVCall** “0” denotes the highest priority and “3” denotes the lowest priority.
[27:0]	**Reserved**	Reserved.

System Handler Priority Register 3 (SHPR3)

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

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

Bits	Description	
[31:28]	**PRI_15**	**Priority of System Handler 15 – SysTick** "0" denotes the highest priority and "3" denotes the lowest priority.
[27:24]	**Reserved**	Reserved.
[23:20]	**PRI_14**	**Priority of System Handler 14 – PendSV** "0" denotes the highest priority and "3" denotes the lowest priority.
[19:0]	**Reserved**	Reserved.

6.3 Clock Controller

6.3.1 Overview

The clock controller generates clocks for the whole chip, including system clocks and all peripheral clocks. The clock controller also implements the power control function with the individually clock ON/OFF control, clock source selection and a clock divider. The chip will not enter Power-down mode until CPU sets the Power-down enable bit PDEN(CLK_PWRCTL[7]) and Cortex®-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 MHz external high speed crystal (HXT) and 48 MHz internal high speed RC oscillator (HIRC) to reduce the overall system power consumption. Figure 6.3-1 shows the clock generator and the overview of the clock source control.

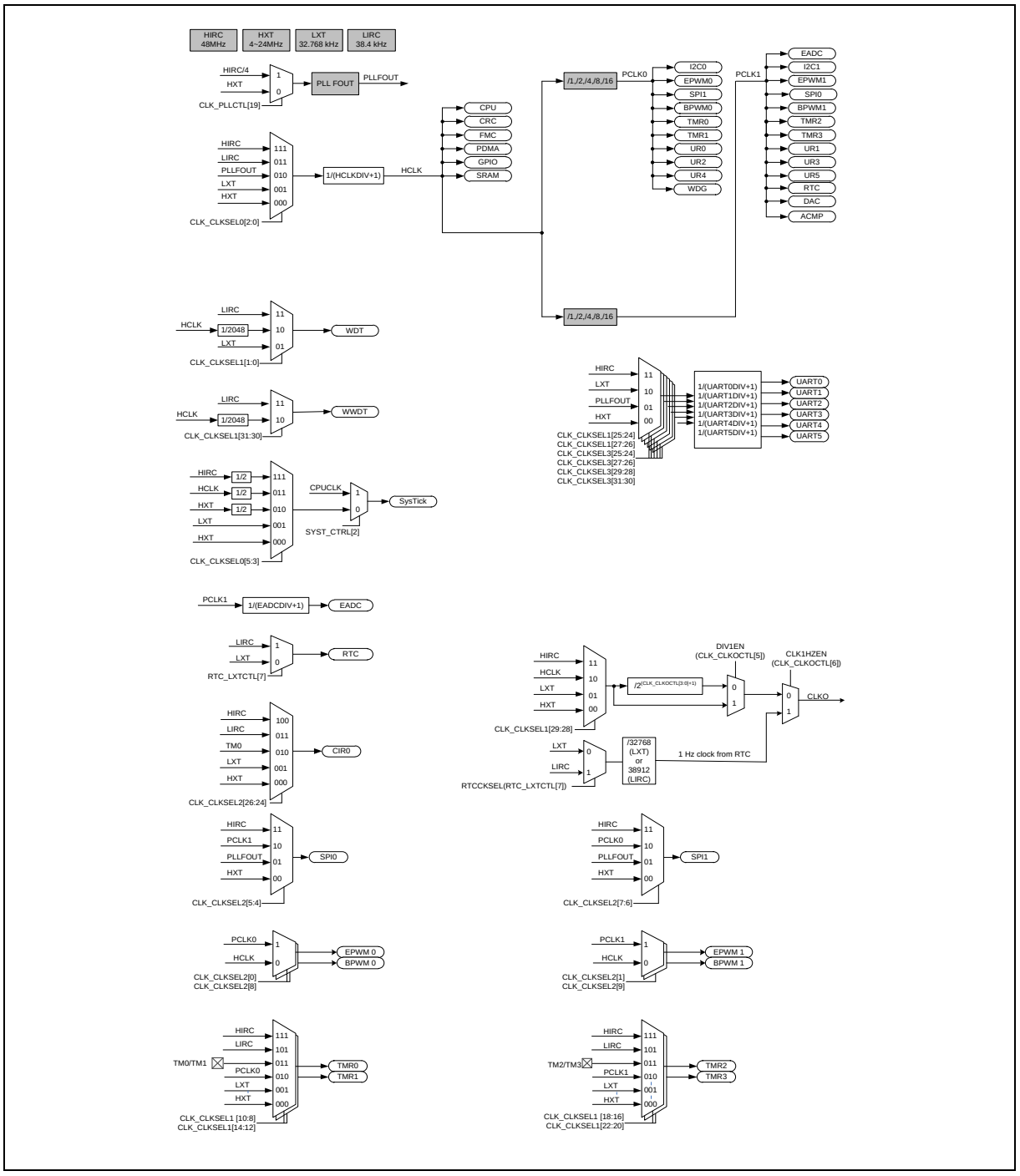


Figure 6.3-1 Clock Generator Global View Diagram

6.3.2 Clock Generator

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

- 32.768 kHz external low speed crystal oscillator (LXT)
- 4~24 MHz external high speed crystal oscillator (HXT)
- Programmable PLL output clock frequency (PLLFOUT), PLL source can be selected from

external 4~24 MHz external high speed crystal (HXT) or 48 MHz internal high speed oscillator divided by 4 (HIRC/4)

- 48 MHz internal high speed RC oscillator (HIRC)
- 38.4 kHz internal low speed RC oscillator (LIRC)

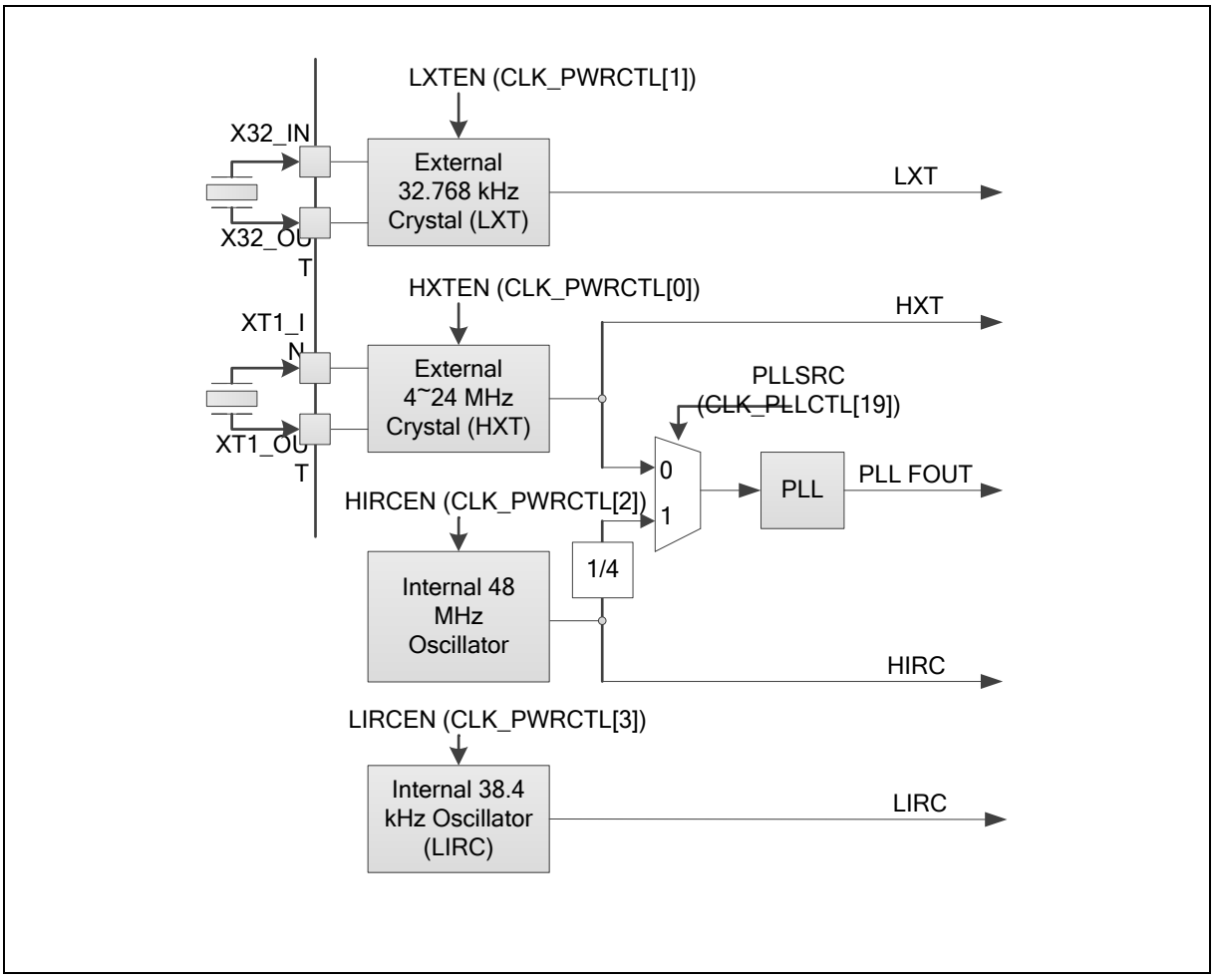


Figure 6.3-2 Clock Generator Block Diagram

6.3.3 System Clock and SysTick Clock

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

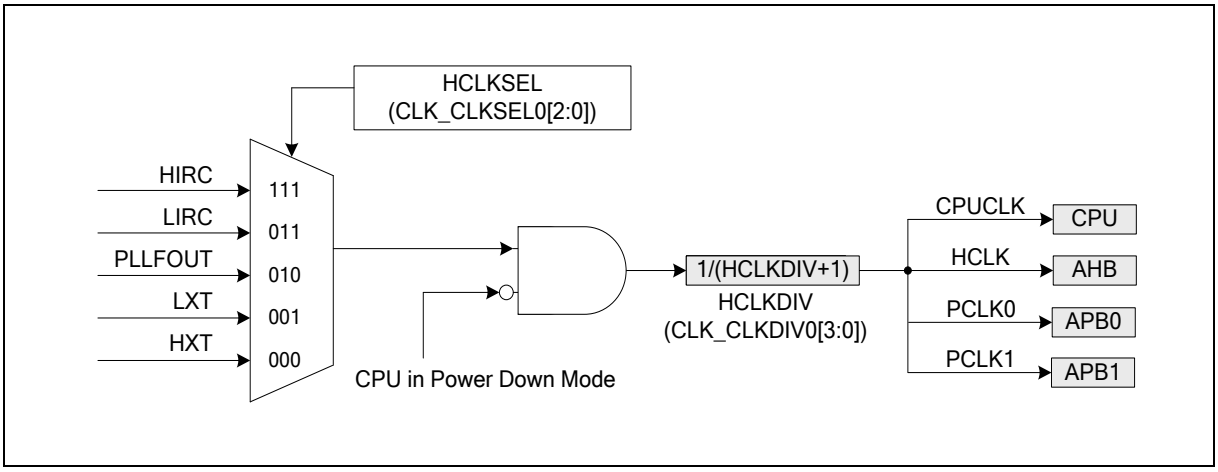


Figure 6.3-3 System Clock Block Diagrams

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

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

Figure 6.3-4 shows The HXT clock stops detection and system clock switches to HIRC procedure

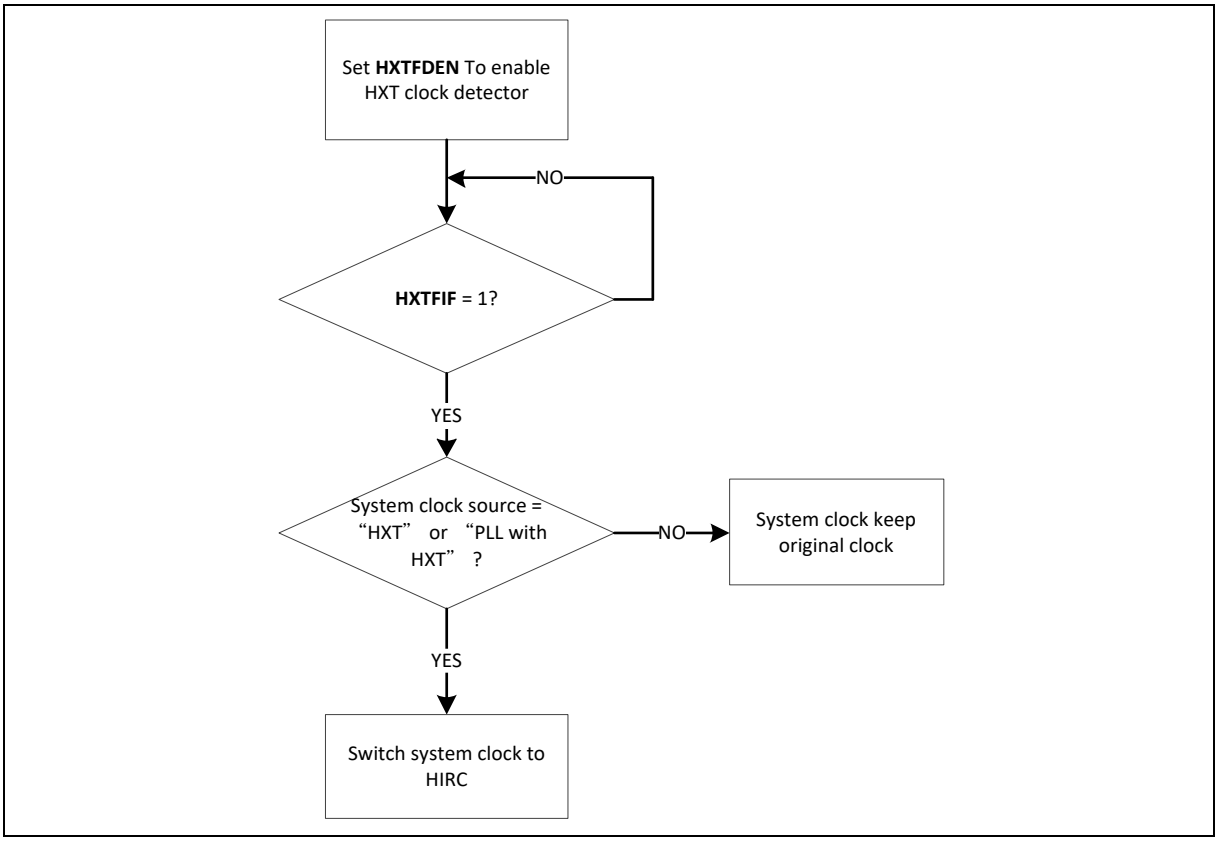


Figure 6.3-4 HXT Stop Protect Procedure

The formula of UPERBD and LOWERBD as below

- HIRC_period * 2048 < HXT_period * UPERBD
- HIRC_period* 2048 > HXT_period* LOWERBD

The clock source of SysTick in Cortex®-M4 core can use CPU clock or external clock (SYST_CTRL[2]). If using external clock, the SysTick clock (STCLK) has 5 clock sources. The clock source switch depends on the setting of the register STCLKSEL (CLK_CLKSEL0[5:3]). The block diagram is shown in Figure 6.3-5.

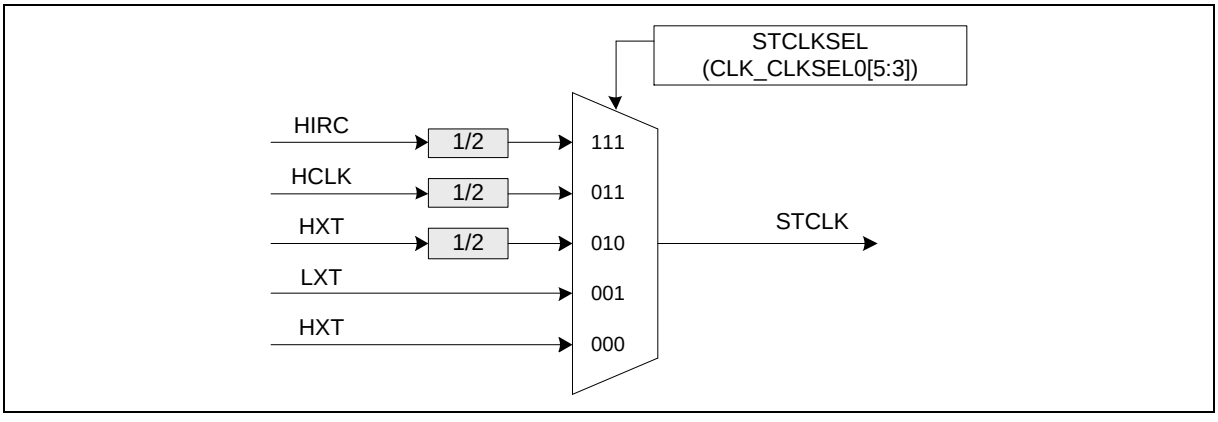


Figure 6.3-5 SysTick Clock Control Block Diagram

6.3.4 Peripherals Clock

Each peripheral clock has its own clock source selection. Refer to the CLK_CLKSEL1, CLK_CLKSEL2 and CLK_CLKSEL3 register.

6.3.5 Power-down Mode Clock

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

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

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

6.3.6 Clock Output

This device is equipped with a power-of-2 frequency divider which is composed 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.

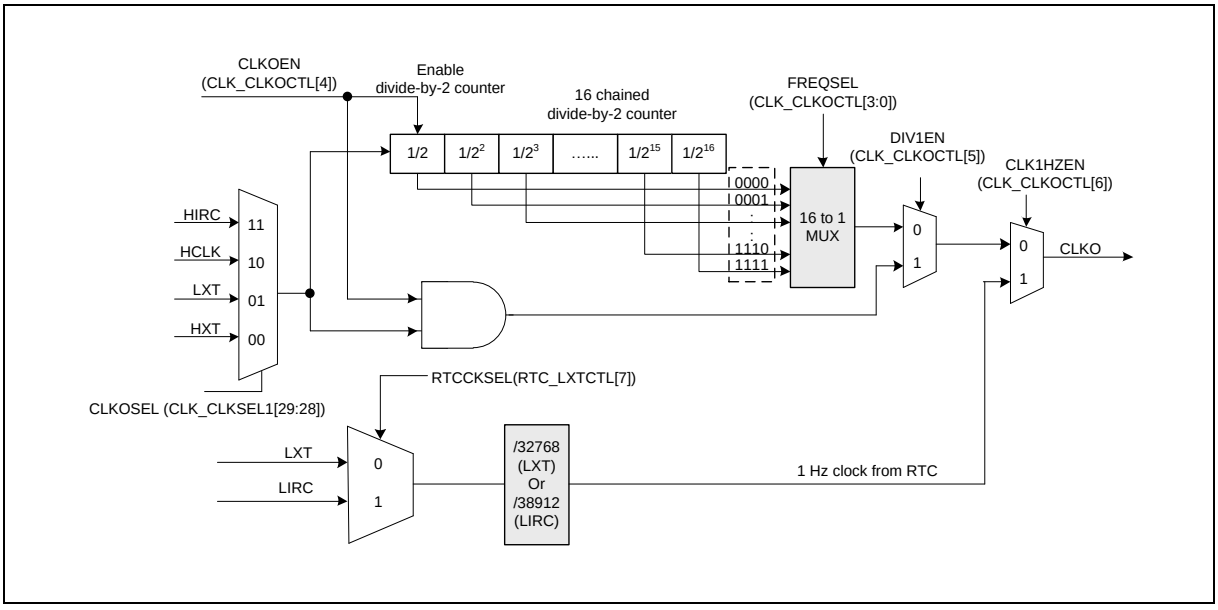


Figure 6.3-6 Clock Output Block Diagram

6.3.7 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_001C
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	0xBF77_7703
CLK_CLKSEL2	CLK_BA+0x18	R/W	Clock Source Select Control Register 2	0x0400_03A3
CLK_CLKSEL3	CLK_BA+0x1C	R/W	Clock Source Select Control Register 3	0xFF00_0000
CLK_CLKDIV0	CLK_BA+0x20	R/W	Clock Divider Number Register 0	0x0000_0003
CLK_CLKDIV4	CLK_BA+0x30	R/W	Clock Divider Number 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_C49E
CLK_STATUS	CLK_BA+0x50	R	Clock Status Monitor Register	0x0000_00XX
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_0000

6.3.8 Register Description

System Power-down Control Register (CLK_PWRCTL)

Register	Offset	R/W	Description	Reset Value
CLK_PWRCTL	CLK_BA+0x00	R/W	System Power-down Control Register	0x0000_001C

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

Bits	Description	
[31]	**HXTMD**	**HXT Bypass Mode (Write Protect)** This is a protected register. Please refer to open lock sequence to program it. 0 = HXT work as crystal mode. PF.2 and PF.3 are configured as external high speed crystal (HXT) pins. 1 = HXT works as external clock mode. PF.3 is configured as external clock input pin. **Note 1:** When HXTMD = 1, PF.3 MFP should be setting as GPIO mode. The DC characteristic of XT1_IN is the same as GPIO. **Note 2:** This bit is write protected. Refer to the SYS_REGCTL register.
[30:23]	**Reserved**	Reserved.
[22:20]	**HXTGAIN**	**HXT Gain Control Bit (Write Protect)** This is a protected register. Please refer to open lock sequence to program it. Gain control is used to enlarge the gain of crystal to make sure crystal work normally. If gain control is enabled, crystal will consume more power than gain control off. 000 = HXT frequency is from 4 MHz to 8 MHz. 001 = HXT frequency is from 8 MHz to 12 MHz. 010 = HXT frequency is from 12 MHz to 16 MHz. 011 = HXT frequency is from 16 MHz to 24 MHz. 000 = HXT frequency is from 4 MHz to 8 MHz. (Crystal) 001 = HXT frequency is from 8 MHz to 12 MHz. (Crystal) 010 = HXT frequency is from 12 MHz to 16 MHz. (Crystal) 011 = HXT frequency is from 16 MHz to 24 MHz. (Crystal) 100 = HXT frequency is from 4 MHz to 8 MHz. (Resonator) 101 = HXT frequency is from 8 MHz to 12 MHz. (Resonator) 110 = HXT frequency is from 12 MHz to 16 MHz. (Resonator) 111 = HXT frequency is from 16 MHz to 24 MHz. (Resonator) Others: Reserved **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[19:18]	**Reserved**	Reserved.

[17:16]	**HIRCSTBS**	**HIRC Stable Count Select (Write Protect)** 00 = HIRC stable count = 512 clocks. 01 = HIRC stable count = 1024 clocks. 10 = HIRC stable count = 2048 clocks. 11 = HIRC stable count = 256 clocks. Others: Reserved **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[15:8]	**Reserved**	Reserved.
[7]	**PDEN**	**System Power-down Enable (Write Protect)** When this bit is set to 1, Power-down mode is enabled and chip keeps active till the CPU sleep mode is also active and then the chip enters Power-down mode. When chip wakes up from Power-down mode, this bit is auto cleared. Users need to set this bit again for next Power-down. In Power-down mode, HXT and the HIRC will be disabled in this mode, but LXT and LIRC are not controlled by Power-down mode. In Power-down mode, the PLL and system clock are disabled, and ignored the clock source selection. The clocks of peripheral are not controlled by Power-down mode, if the peripheral clock source is from LXT or LIRC. 0 = Chip 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. **Note 1:** Write 1 to clear the bit to 0. **Note 2:** This bit works only if PDWKIEN (CLK_PWRCTL[5]) is set to 1.
[5]	**PDWKIEN**	**Power-down Mode Wake-up Interrupt Enable Bit (Write Protect)** 0 = Power-down mode wake-up interrupt Disabled. 1 = Power-down mode wake-up interrupt Enabled. **Note 1:** The interrupt will occur when both PDWKIF and PDWKIEN are high. **Note 2:** This bit is write protected. Refer to the SYS_REGLCTL register.
[4]	**PDWKDLY**	**Enable the Wake-up Delay Counter (Write Protect)** When the chip wakes up from Power-down mode, the clock control will delay certain clock cycles to wait system clock stable. The delayed clock cycle is 4096 clock cycles when chip works at 4~24 MHz external high speed crystal oscillator (HXT), and 256 clock cycles when chip works at 48 MHz internal high speed RC oscillator (HIRC). 0 = Clock cycles delay Disabled. 1 = Clock cycles delay Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[3]	**LIRCEN**	**LIRC Enable Bit (Write Protect)** 0 = 38.4 kHz internal low speed RC oscillator (LIRC) Disabled. 1 = 38.4 kHz internal low speed RC oscillator (LIRC) Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[2]	**HIRCEN**	**HIRC Enable Bit (Write Protect)** 0 = 48 MHz internal high speed RC oscillator (HIRC) Disabled. 1 = 48 MHz internal high speed RC oscillator (HIRC) Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

[1]	**LXTEN**	**LXT Enable Bit (Write Protect)** 0 = 32.768 kHz external low speed crystal (LXT) Disabled. 1 = 32.768 kHz external low speed crystal (LXT) Enabled. **Note 1:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note 2:** When LXT is enabled, GPF.4 and GPF.5 must be set as input mode.
[0]	**HXTEN**	**HXT Enable Bit (Write Protect)** 0 = 4~24 MHz external high speed crystal (HXT) Disabled. 1 = 4~24 MHz external high speed crystal (HXT) Enabled. **Note 1:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note 2:** When HXT is enabled, GPF.2 and GPF.3 must be set as input mode.

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
GPHCKEN	GPGCKEN	GPFCKEN	GPECKEN	GPDCKEN	GPCCKEN	GPBCKEN	GPACKEN
23	22	21	20	19	18	17	16
GPICKEN	Reserved			TRACECKEN	Reserved		
15	14	13	12	11	10	9	8
FMCIDLE	Reserved						
7	6	5	4	3	2	1	0
CRCCKEN	Reserved		STCLKEN	Reserved	ISPCKEN	PDMACKEN	Reserved

Bits	Description	
[31]	GPHCKEN	**GPIOH Clock Enable Bit** 0 = GPIOH port clock Disabled. 1 = GPIOH port clock Enabled.
[30]	GPGCKEN	**GPIOG Clock Enable Bit** 0 = GPIOG port clock Disabled. 1 = GPIOG port clock Enabled.
[29]	GPFCKEN	**GPIOF Clock Enable Bit** 0 = GPIOF port clock Disabled. 1 = GPIOF port clock Enabled.
[28]	GPECKEN	**GPIOE Clock Enable Bit** 0 = GPIOE port clock Disabled. 1 = GPIOE port clock Enabled.
[27]	GPDCKEN	**GPIOD Clock Enable Bit** 0 = GPIOD port clock Disabled. 1 = GPIOD port clock Enabled.
[26]	GPCCKEN	**GPIOC Clock Enable Bit** 0 = GPIOC port clock Disabled. 1 = GPIOC port clock Enabled.
[25]	GPBCKEN	**GPIOB Clock Enable Bit** 0 = GPIOB port clock Disabled. 1 = GPIOB port clock Enabled.
[24]	GPACKEN	**GPIOA Clock Enable Bit** 0 = GPIOA port clock Disabled. 1 = GPIOA port clock Enabled.

[23]	**GPICKEN**	**GPIOI Clock Enable Bit** 0 = GPIOI port clock Disabled. 1 = GPIOI port clock Enabled.
[22:20]	**Reserved**	Reserved.
[19]	**TRACECKEN**	**TRACE Clock Enable Bit** 0 = TRACE clock Disabled. 1 = TRACE clock Enabled.
[18:16]	**Reserved**	Reserved.
[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.
[7]	**CRCCKEN**	**CRC Generator Controller Clock Enable Bit** 0 = CRC peripheral clock Disabled. 1 = CRC peripheral clock Enabled.
[6:5]	**Reserved**	Reserved.
[4]	**STCLKEN**	**Cortex®-M4 SysTick Clock Enable Bit** 0 = Cortex®-M4 sys tick clock Disabled. 1 = Cortex®-M4 sys tick clock Enabled.
[3]	**Reserved**	Reserved.
[2]	**ISPCKEN**	**Flash ISP Controller Clock Enable Bit** 0 = Flash ISP peripheral clock Disabled. 1 = Flash ISP peripheral clock Enabled.
[1]	**PDMACKEN**	**PDMA Controller Clock Enable Bit** 0 = PDMA peripheral clock Disabled. 1 = PDMA peripheral clock Enabled.
[0]	**Reserved**	Reserved.

APB Devices Clock Enable Control Register 0 (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			EADCCKEN	Reserved			
23	22	21	20	19	18	17	16
Reserved		UART5CKEN	UART4CKEN	UART3CKEN	UART2CKEN	UART1CKEN	UART0CKEN
15	14	13	12	11	10	9	8
Reserved	SPI1CKEN	SPI0CKEN	Reserved			I2C1CKEN	I2C0CKEN
7	6	5	4	3	2	1	0
ACMP01CKEN	CLKOCKEN	TMR3CKEN	TMR2CKEN	TMR1CKEN	TMR0CKEN	RTCCKEN	WDTCKEN

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28]	**EADCCKEN**	**Enhanced Analog-digital-converter (EADC) Clock Enable Bit** 0 = EADC clock Disabled. 1 = EADC clock Enabled.
[27:22]	**Reserved**	Reserved.
[21]	**UART5CKEN**	**UART5 Clock Enable Bit** 0 = UART5 clock Disabled. 1 = UART5 clock Enabled.
[20]	**UART4CKEN**	**UART4 Clock Enable Bit** 0 = UART4 clock Disabled. 1 = UART4 clock Enabled.
[19]	**UART3CKEN**	**UART3 Clock Enable Bit** 0 = UART3 clock Disabled. 1 = UART3 clock Enabled.
[18]	**UART2CKEN**	**UART2 Clock Enable Bit** 0 = UART2 clock Disabled. 1 = UART2 clock Enabled.
[17]	**UART1CKEN**	**UART1 Clock Enable Bit** 0 = UART1 clock Disabled. 1 = UART1 clock Enabled.
[16]	**UART0CKEN**	**UART0 Clock Enable Bit** 0 = UART0 clock Disabled. 1 = UART0 clock Enabled.

[15]	**Reserved**	Reserved.
[14]	**SPI1CKEN**	**SPI1 Clock Enable Bit** 0 = SPI1 clock Disabled. 1 = SPI1 clock Enabled.
[13]	**SPI0CKEN**	**SPI0 Clock Enable Bit** 0 = SPI0 clock Disabled. 1 = SPI0 clock Enabled.
[12:10]	**Reserved**	Reserved.
[9]	**I2C1CKEN**	**I^2C1 Clock Enable Bit** 0 = I^2C1 clock Disabled. 1 = I^2C1 clock Enabled.
[8]	**I2C0CKEN**	**I^2C0 Clock Enable Bit** 0 = I^2C0 clock Disabled. 1 = I^2C0 clock Enabled.
[7]	**ACMP01CKEN**	**Analog Comparator 0/1 Clock Enable Bit** 0 = Analog comparator 0/1 clock Disabled. 1 = Analog comparator 0/1 clock Enabled.
[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. 0 = RTC clock Disabled. 1 = RTC clock Enabled.
[0]	**WDTCKEN**	**Watchdog Timer Clock Enable Bit (Write Protect)** 0 = Watchdog timer clock Disabled. 1 = Watchdog timer clock Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

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							PRNGCKEN
23	22	21	20	19	18	17	16
Reserved				BPWM1CKEN	BPWM0CKEN	EPWM1CKEN	EPWM0CKEN
15	14	13	12	11	10	9	8
CIR0CKEN	Reserved		DACCKEN	Reserved			
7	6	5	4	3	2	1	0
Reserved							

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**PRNGCKEN**	**PRNG Clock Enable Bit** 0 = PRNG clock Disabled. 1 = PRNG clock Enabled.
[23:20]	**Reserved**	Reserved.
[19]	**BPWM1CKEN**	**BPWM1 Clock Enable Bit** 0 = BPWM1 clock Disabled. 1 = BPWM1 clock Enabled.
[18]	**BPWM0CKEN**	**BPWM0 Clock Enable Bit** 0 = BPWM0 clock Disabled. 1 = BPWM0 clock Enabled.
[17]	**EPWM1CKEN**	**EPWM1 Clock Enable Bit** 0 = EPWM1 clock Disabled. 1 = EPWM1 clock Enabled.
[16]	**EPWM0CKEN**	**EPWM0 Clock Enable Bit** 0 = EPWM0 clock Disabled. 1 = EPWM0 clock Enabled.
[15]	**CIR0CKEN**	**CIR0 Clock Enable Bit** 0 = CIR0 clock Disabled. 1 = CIR0 clock Enabled.
[14:13]	**Reserved**	Reserved.
[12]	**DACCKEN**	**DAC Clock Enable Bit** 0 = DAC clock Disabled. 1 = DAC clock Enabled.

[11:0]	**Reserved**	Reserved.

Clock Source Select Control Register 0 (CLK_CLKSEL0)

Register	Offset	R/W	Description	Reset Value
CLK_CLKSEL0	CLK_BA+0x10	R/W	Clock Source Select Control Register 0	0x0000_003X

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

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:3]	**STCLKSEL**	**Cortex®-M4 SysTick Clock Source Selection (Write Protect)** 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. **Note 1:** if SysTick clock source is not from HCLK (i.e. SYST_CTRL[2] = 0), SysTick clock source must less than or equal to HCLK/2. **Note 2:** This bit is write protected. Refer to the SYS_REGLCTL register.
[2:0]	**HCLKSEL**	**HCLK Clock Source Selection (Write Protect)** Before clock switching, the related clock sources (both pre-select and new-select) must be turned on. The default value is reloaded from the value of CFOSC (CONFIG0[26:24]) in user configuration register of Flash controller by any reset. Therefore the default value is either 000b or 111b. 000 = Clock source from HXT. 001 = Clock source from LXT. 010 = Clock source from PLL. 011 = Clock source from LIRC. 111= Clock source from HIRC. Other = Reserved. **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	0xBF77_7703

31	30	29	28	27	26	25	24
WWDTSEL		CLKOSEL		UART1SEL		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.
[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]	**UART1SEL**	**UART1 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).
[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.
[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.
[19]	**Reserved**	Reserved.
[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.
[15]	**Reserved**	Reserved.
[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.
[11]	**Reserved**	Reserved.
[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.
[7:2]	**Reserved**	Reserved.
[1:0]	**WDTSEL**	**Watchdog Timer Clock Source Selection (Write Protect)** 00 = Reserved. 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	0x0400_03A3

31	30	29	28	27	26	25	24
Reserved					CIR0SEL		
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						BPWM1SEL	BPWM0SEL
7	6	5	4	3	2	1	0
SPI1SEL		SPI0SEL		Reserved		EPWM1SEL	EPWM0SEL

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:24]	**CIR0SEL**	**CIR0 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 Timer0 clock output (TM0). 011 = Clock source from internal low speed RC oscillator (LIRC). 100 = Clock source from internal high speed RC oscillator (HIRC). Others = Reserved.
[23:10]	**Reserved**	Reserved.
[9]	**BPWM1SEL**	**BPWM1 Clock Source Selection** The peripheral clock source of BPWM1 is defined by BPWM1SEL. 0 = Clock source from HCLK. 1 = Clock source from PCLK1.
[8]	**BPWM0SEL**	**BPWM0 Clock Source Selection** The peripheral clock source of BPWM0 is defined by BPWM0SEL. 0 = Clock source from HCLK. 1 = Clock source from PCLK0.
[7:6]	**SPI1SEL**	**SPI1 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]	**SPI0SEL**	**SPI0 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]	**Reserved**	Reserved.
[1]	**EPWM1SEL**	**EPWM1 Clock Source Selection** The peripheral clock source of EPWM1 is defined by EPWM1SEL. 0 = Clock source from HCLK. 1 = Clock source from PCLK1.
[0]	**EPWM0SEL**	**EPWM0 Clock Source Selection** The peripheral clock source of EPWM0 is defined by EPWM0SEL. 0 = Clock source from HCLK. 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	0xFF00_0000

31	30	29	28	27	26	25	24
UART5SEL		**UART4SEL**		**UART3SEL**		**UART2SEL**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:30]	**UART5SEL**	**UART5 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).
[29:28]	**UART4SEL**	**UART4 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).
[27:26]	**UART3SEL**	**UART3 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).
[25:24]	**UART2SEL**	**UART2 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).
[23:0]	**Reserved**	Reserved.

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	0x0000_0003

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
UART1DIV				UART0DIV			
7	6	5	4	3	2	1	0
Reserved				HCLKDIV			

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**EADCDIV**	**EADC Clock Divide Number From EADC Clock Source** EADC clock frequency = (EADC clock source frequency) / (EADCDIV + 1).
[15:12]	**UART1DIV**	**UART1 Clock Divide Number From UART1 Clock Source** UART1 clock frequency = (UART1 clock source frequency) / (UART1DIV + 1).
[11:8]	**UART0DIV**	**UART0 Clock Divide Number From UART0 Clock Source** UART0 clock frequency = (UART0 clock source frequency) / (UART0DIV + 1).
[7:4]	**Reserved**	Reserved.
[3:0]	**HCLKDIV**	**HCLK Clock Divide Number From HCLK Clock Source** HCLK clock frequency = (HCLK clock source frequency) / (HCLKDIV + 1).

Clock Divider Number Register 4 (CLK_CLKDIV4)

Register	Offset	R/W	Description	Reset Value
CLK_CLKDIV4	CLK_BA+0x30	R/W	Clock Divider Number Register 4	0x0000_0000

31	30	29	28	27	26	25	24
TRACEDIV							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
UART5DIV				UART4DIV			
7	6	5	4	3	2	1	0
UART3DIV				UART2DIV			

Bits	Description	
[31:24]	**TRACEDIV**	**Cortex® M4 ETM Trace Clock Divide Number From ETM Trace Clock Source** Cortex® M4 ETM TRACE clock frequency = (HCLK clock source frequency) / (TRACEDIV + 1).
[23:16]	**Reserved**	Reserved.
[15:12]	**UART5DIV**	**UART5 Clock Divide Number From UART5 Clock Source** UART5 clock frequency = (UART5 clock source frequency) / (UART5DIV + 1).
[11:8]	**UART4DIV**	**UART4 Clock Divide Number From UART4 Clock Source** UART4 clock frequency = (UART4 clock source frequency) / (UART4DIV + 1).
[7:4]	**UART3DIV**	**UART3 Clock Divide Number From UART3 Clock Source** UART3 clock frequency = (UART3 clock source frequency) / (UART3DIV + 1).
[3:0]	**UART2DIV**	**UART2 Clock Divide Number From UART2 Clock Source** UART2 clock frequency = (UART2 clock source frequency) / (UART2DIV + 1).

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.
[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.
[3]	**Reserved**	Reserved.
[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.

PLL Control Register (CLK_PLLCTL)

Register	Offset	R/W	Description	Reset Value
CLK_PLLCTL	CLK_BA+0x40	R/W	PLL Control Register	0x0005_C49E

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23]	**STBSEL**	**PLL Stable Counter Selection (Write Protect)** 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 16128 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.
[19]	**PLLSRC**	**PLL Source Clock Selection (Write Protect)** 0 = PLL source clock from 4~24 MHz external high-speed crystal oscillator (HXT). 1 = PLL source clock from 48 MHz internal high-speed oscillator divided by 4 (HIRC/4). **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[18]	**OE**	**PLL OE (FOUT Enable) Pin Control (Write Protect)** 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 Protect)** 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 Protect)** 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 Protect)** 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 Protect)** 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 Protect)** Refer to the formulas below the table. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

Output Clock Frequency Setting

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

Constraint:

1. $3.2MHz < FIN < 150MHz$
2. $800kHz < \frac{FIN}{2 * NR} < 8MHz$
3. $200MHz < FCO = FIN * \frac{NF}{NR} < 500MHz,$
   $FCO > 250MHz \ \ is \ preferred$

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

Table 6.3-1 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_00XX

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**CLKSFAIL**	**Clock Switching Fail Flag (Read Only)** This bit is updated when software switches system clock source. If switch target clock is stable, this bit will be set to 0. If switch target clock is not stable, this bit will be set to 1. 0 = Clock switching success. 1 = Clock switching failure. **Note:** Write 1 to clear the bit to 0.
[6:5]	**Reserved**	Reserved.
[4]	**HIRCSTB**	**HIRC Clock Source Stable Flag (Read Only)** 0 = 48 MHz internal high speed RC oscillator (HIRC) clock is not stable or disabled. 1 = 48 MHz internal high speed RC oscillator (HIRC) clock is stable and enabled.
[3]	**LIRCSTB**	**LIRC Clock Source Stable Flag (Read Only)** 0 = 38.4 kHz internal low speed RC oscillator (LIRC) clock is not stable or disabled. 1 = 38.4 kHz internal low speed RC oscillator (LIRC) clock is stable and enabled.
[2]	**PLLSTB**	**Internal PLL Clock Source Stable Flag (Read Only)** 0 = Internal PLL clock is not stable or disabled. 1 = Internal PLL clock is stable and enabled.
[1]	**LXTSTB**	**LXT Clock Source Stable Flag (Read Only)** 0 = 32.768 kHz external low speed crystal oscillator (LXT) clock is not stable or disabled. 1 = 32.768 kHz external low speed crystal oscillator (LXT) clock is stabled and enabled.
[0]	**HXTSTB**	**HXT Clock Source Stable Flag (Read Only)** 0 = 4~24 MHz external high speed crystal oscillator (HXT) clock is not stable or disabled. 1 = 4~24 MHz external high speed crystal oscillator (HXT) clock is stable and enabled.

Clock 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.
[6]	**CLK1HZEN**	**Clock Output 1Hz Enable Bit** 0 = 1 Hz clock output for 32.768 kHz or 38 kHz frequency compensation Disabled. 1 = 1 Hz clock output for 32.768 kHz or 38 kHz frequency compensation Enabled. **Note:** Output for 32.768 kHz(LXT) or 38 kHz(LIRC) based on RTCCKSEL(RTC_LXTCTL[7]).
[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.
[17]	**HXTFQIEN**	**HXT Clock Frequency Range Detector Interrupt Enable Bit** 0 = 4~24 MHz external high speed crystal oscillator (HXT) clock frequency range detector fail interrupt Disabled. 1 = 4~24 MHz external high speed crystal oscillator (HXT) clock frequency range detector fail interrupt Enabled.
[16]	**HXTFQDEN**	**HXT Clock Frequency Range Detector Enable Bit** 0 = 4~24 MHz external high speed crystal oscillator (HXT) clock frequency range detector Disabled. 1 = 4~24 MHz external high speed crystal oscillator (HXT) clock frequency range detector Enabled.
[15:14]	**Reserved**	Reserved.
[13]	**LXTFIEN**	**LXT Clock Fail Interrupt Enable Bit** 0 = 32.768 kHz external low speed crystal oscillator (LXT) clock fail interrupt Disabled. 1 = 32.768 kHz external low speed crystal oscillator (LXT) clock fail interrupt Enabled.
[12]	**LXTFDEN**	**LXT Clock Fail Detector Enable Bit** 0 = 32.768 kHz external low speed crystal oscillator (LXT) clock fail detector Disabled. 1 = 32.768 kHz external low speed crystal oscillator (LXT) clock fail detector Enabled.
[11:6]	**Reserved**	Reserved.
[5]	**HXTFIEN**	**HXT Clock Fail Interrupt Enable Bit** 0 = 4~24 MHz external high speed crystal oscillator (HXT) clock fail interrupt Disabled. 1 = 4~24 MHz external high speed crystal oscillator (HXT) clock fail interrupt Enabled.
[4]	**HXTFDEN**	**HXT Clock Fail Detector Enable Bit** 0 = 4~24 MHz external high speed crystal oscillator (HXT) clock fail detector Disabled. 1 = 4~24 MHz external high speed crystal oscillator (HXT) clock fail detector Enabled.
[3:0]	**Reserved**	Reserved.

Clock Fail Detector Status Register (CLK_CLKDSTS)

Register	Offset	R/W	Description	Reset Value
CLK_CLKDSTS	CLK_BA+0x74	R/W	Clock Fail Detector Status Register	0x0000_0000

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

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

Clock Frequency Range 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	
[31:11]	Reserved	Reserved.
[10:0]	UPERBD	**HXT Clock Frequency Range Detector Upper Boundary Value** The bits define the maximum value of frequency range detector window. When HXT frequency higher than this maximum frequency value, the HXT Clock Frequency Range Detector Interrupt Flag will be set to 1.

Clock Frequency Range 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	
[31:11]	**Reserved**	Reserved.
[10:0]	**LOWERBD**	**HXT Clock Frequency Range Detector Lower Boundary Value** The bits define the minimum value of frequency range detector window. When HXT frequency lower than this minimum frequency value, the HXT Clock Frequency Range Detector Interrupt Flag will be 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_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**PDMSEL**	**Power-down Mode Selection (Write Protect)** This is a protected bit. Please refer to open lock sequence to program it. These bits control chip Power-down mode grade selection when CPU executes WFI/WFE instruction. 000 = Power-down mode is selected. (NPD) 001 = Reserved. 010 = Reserved. 011 = Reserved. 100 = Reserved. 101 = Reserved.. 110 = Reserved. 111 = Reserved. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

6.4 Flash Memeory Controller (FMC)

6.4.1 Overview

The FMC is equipped with 256/512 Kbytes on-chip embedded Flash for application and configurable Data Flash to store some application dependent data. Thus, the total size of application rom (APROM) is 256/512 Kbytes. A User Configuration block provides for system initiation. A 4 Kbytes loader ROM (LDROM) is used for In-System-Programming (ISP) function. A 4 Kbytes cache with zero wait cycle is used to improve Flash access performance. This chip also supports In-Application-Programming (IAP) function. User switches the code executing without chip reset after the embedded Flash is updated.

6.4.2 Features

- Supports dual-bank Flash macro for safe firmware upgrade
- Supports 256/512 Kbytes application ROM (APROM)
- Supports 4 Kbytes loader ROM (LDROM)
- Supports 4 XOM (eXecution Only Memory) regions to conceal user program in APROM.
- Supports 16 bytes User Configuration block to control system initiation
- Supports 2 Kbytes page erase for all embedded Flash
- Supports bank erase for APROM, except for XOM regions
- 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 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 (4K cache 1 way)

6.4.3 Block Diagram

The Flash memory controller (FMC) consists of AHB slave interface, cache memory controller, boot loader, Flash control registers, Flash initialization controller, Flash operation control and embedded Flash memory. The block diagram of Flash memory controller is shown as follows.

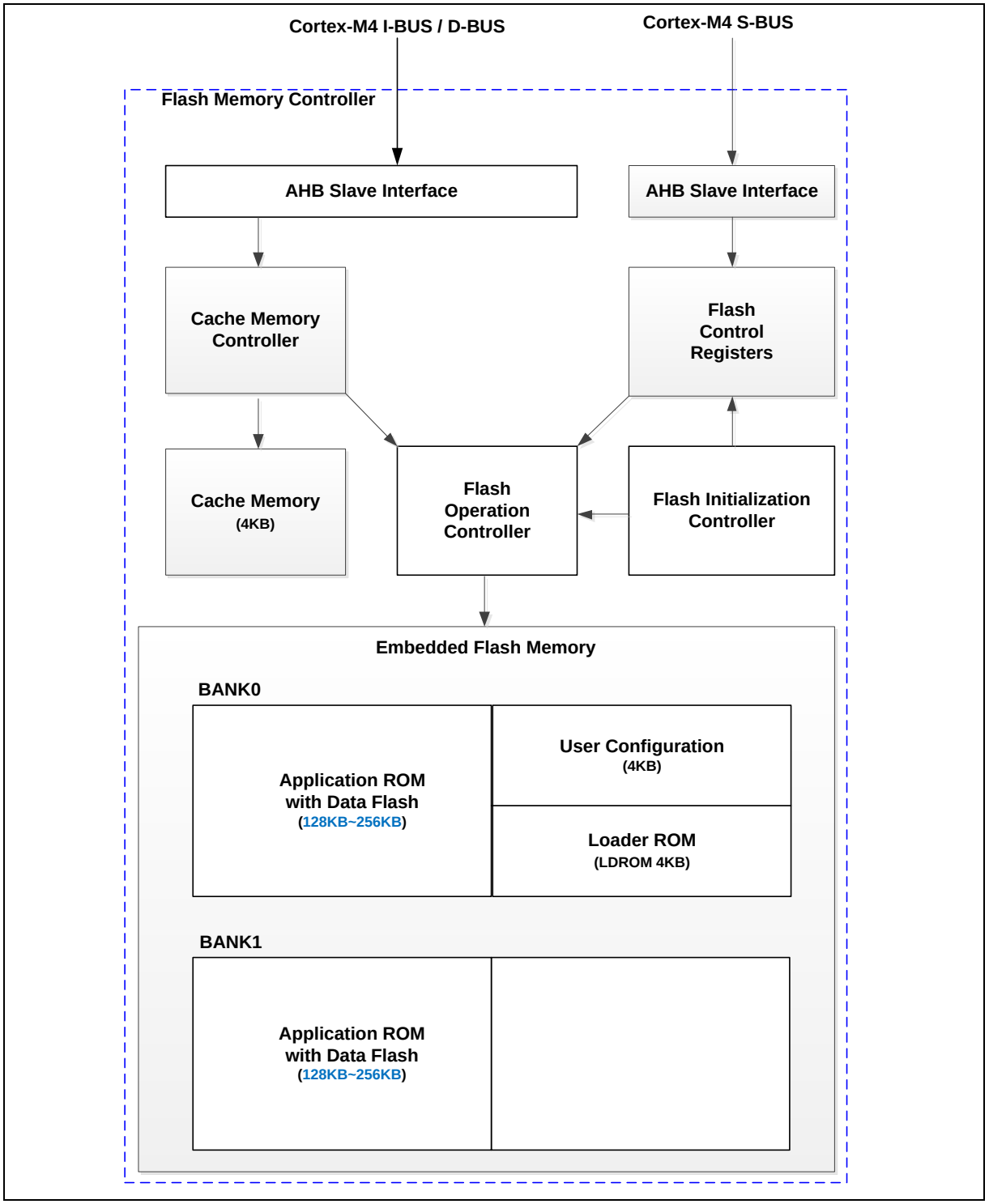


Figure 6.4-1 Flash Memory Controller Block Diagram

AHB Slave Interface

There are two AHB slave interfaces in Flash memory controller, one is from both Cortex®-M4 I-Bus and D-Bus for the 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 with zero wait cycle is implemented between Cortex®-M4 CPU and embedded Flash memory. This cache memory controller 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 detail registers description is in the Register Description section

Flash Initialization Controller

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

Flash Operation Controller

The Flash operations, such as Flash erase, Flash program, and Flash read operation, have specific control timing for embedded Flash memory. The Flash operation controller generates those control timing by 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 Kbytes LDROM, two 4 XOM setting pages, and 256KB/512KB APROM. The page erase Flash size is 2KB, and minimum program bit size is 32 bits.

6.4.4 Functional Description

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

6.4.4.1 Memory Organization

The FMC memory consists of the embedded Flash memory and boot loader. The embedded Flash memory is programmable, and includes APROM, LDROM, XOM setting page, Data Flash, and the User Configuration block. The address map includes Flash memory map and five system address maps: LDROM with IAP, LDROM without IAP, APROM with IAP, APROM without IAP, and Boot Loader with IAP functions.

BANK	Flash Memory Block	Address Range
0	APROM with 512KB	0x00_0000 ~ 0x03_ffff
	APROM with 256KB	0x00_0000 ~ 0x01_ffff
	User Configuration	0x30_0000 ~ 0x30_000f
	LDROM	0x10_0000 ~ 0x10_0fff
1	APROM with 512KB	0x04_0000 ~ 0x07_ffff
	APROM with 256KB	0x02_0000 ~ 0x03_ffff

Flash Memory Block	Address Range
APROM with 256KB	0x00_0000 ~ 0x01_ffff
APROM with 128KB	0x00_0000 ~ 0x00_ffff
User Configuration	0x30_0000 ~ 0x30_000f
LDROM	0x10_0000 ~ 0x10_0fff
XOM setting	0x20_0000 ~ 0x20_0fff

Table 6.4-1 Block Address Range

6.4.4.2 LDROM APROM

LDROM is designed for a loader to implement In-System-Programming (ISP) function by user. LDROM is a 4KB embedded Flash memory, the Flash address range is from 0x0010_0000 to 0x0010_0FFF. APROM is main memory for user applications. APROM size is 256KB/512KB. . All of the embedded Flash memory is 2KB page erased.

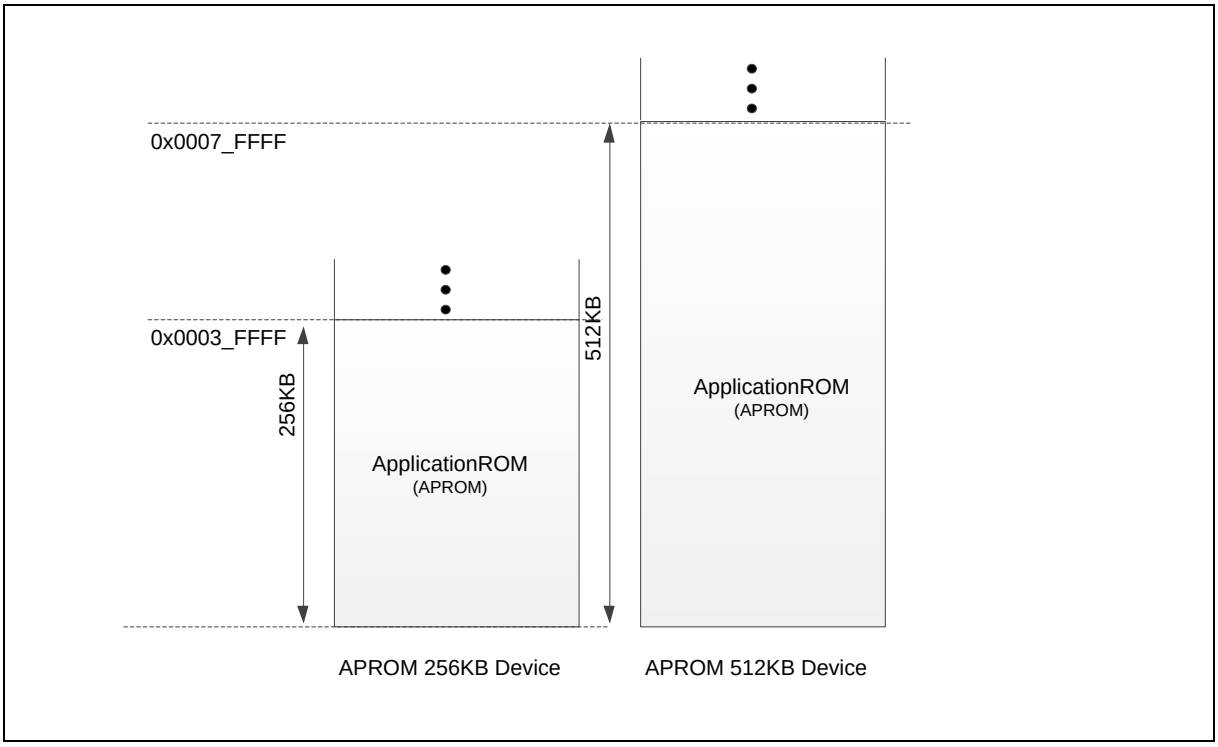


Figure 6.4-2 APROM Examples (256KB/512KB)

6.4.4.3 User Configuration Block

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

CONFIG0 (Address = 0x0030_0000)

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

Bits	Description	
[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 force at LIRC and LIRC can't be disabled. **CWDTEN[2:0]** is CONFIG0[31][4][3], 011 = WDT hardware enable function is active. WDT clock is always on except chip enters Power-down mode. When chip enters Power-down mode, WDT clock is always on if CWDTPDEN is 0 or WDT clock is controlled by LIRCEN (CLK_PWRCTL[3]) if CWDTPDEN is 1. Please refer to bit field description of CWDTPDEN. 111 = WDT hardware enable function is inactive. Others = WDT hardware enable function is active. WDT clock is always on.
[30]	**CWDTPDEN**	**Watchdog Clock Power-down Enable Bit** 0 = Watchdog Timer clock kept enabled when chip enters Power-down. 1 = Watchdog Timer clock is controlled by LIRCEN (CLK_PWRCTL[3]) when chip enters Power-down. **Note:** This bit only works if CWDTEN[2:0] is set to 011
[29:23]	**Reserved**	Reserved.
[22:21]	**CBOV**	**Brown-out Voltage Selection** 000 = Brown-out voltage is 2.4V. 001 = Brown-out voltage is 2.7V. 010 = Brown-out voltage is 3.7V. 011 = Brown-out voltage is 4.4V.
[20]	**CBORST**	**Brown-out Reset Enable Bit** 0 = Brown-out reset Enabled after powered on. 1 = Brown-out reset Disabled after powered on.
[19]	**CBODEN**	**Brown-out Detector Enable Bit** 0= Brown-out detect Enabled after powered on. 1= Brown-out detect Disabled after powered on.
[18:12]	**Reserved**	Reserved.

[11]	**ICELOCK**	**ICE Lock Bit** This bit only used to disable ICE function. User may use it with LOCK (CONFIG0[1]) bit or ALOCK (CONFIG2[7:0]) bits to increase security level. 0 = ICE function Disabled. 1 = ICE function Enabled.
[10]	**CIOINI**	**I/O Initial State Selection** 0 = All GPIO set as Quasi-bidirectional mode after chip powered on. 1 = All GPIO set as input tri-state mode after powered on.
[9:8]	**Reserved**	Reserved.
[7:6]	**CBS**	**Chip Booting Selection** When CBS[0] = 0, the LDROM base address is mapping to 0x100000 and APROM base address is mapping to 0x0. User could access both APROM and LDROM without boot switching. In other words, if IAP mode is supported, the code in LDROM and APROM can be called by each other. CBS[0] value will be assigned to 0 when MBS =0. 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 and MBS = 1. VECMAP (FMC_ISPSTS[23:9]) is only be used to remap 0x0~0x1ff when CBS[0] = 0 or MBS = 0.
[5]	**Reserved**	Reserved.
[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], 011 = WDT hardware enable function is active. WDT clock is always on except chip enters Power-down mode. When chip enters Power-down mode, WDT clock is always on if CWDTPDEN is 0 or WDT clock is controlled by LIRCEN (CLK_PWRCTL[3]) if CWDTPDEN is 1. Please refer to bit field description of CWDTPDEN. 111 = WDT hardware enable function is inactive. Others = WDT hardware enable function is active. WDT clock is always on.
[2]	**Reserved**	Reserved.
[1]	**LOCK**	**Security Lock Control** 0 = Flash memory content is locked. 1 = Flash memory content is not locked if ALOCK(CONFIG2[7:0]) and SBLOCK (CONFIG2[15:8]) is 0x5A. When Flash data is locked by LOCK,user can look the lock effect of FMC up in lock effect tables. To unlock system, user can use ISP command to disable LOCK bit or erase whole chip (Chip Erase) by ICP tool.
[0]	**Reserved**	Reserved.

Note: The config bits should be 1 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
Reserved							
7	6	5	4	3	2	1	0
ALOCK							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**ALOCK**	**Advance Security Lock Control** 0x5A = Flash memory content is unlocked if LOCK (CONFIG0[1]) is 1. The others = Flash memory content is locked. When Flash data is locked by ALOCK,user must look the lock effect of FMC up in lock effect tables. To unlock system, user can use ISP command to disable LOCK bit or erase whole chip (Chip Erase) by ICP tool.

Note: The config bits should be 1 if reserved.

6.4.4.4 Execution Only Memory (XOM)

The execution only memory (XOM) is used to store instructions for security application which are not allowed for data access via AHB-Bus. There are four XOM regions in APROM at most. To define a new XOM region, user must complete XOM settings in XOM setting page (0x20_0000 ~ 0x20_0008 for XOM region 0; 0x20_0010 ~ 0x20_0018 for XOM region 1 ; 0x20_0020 ~ 0x20_0028 for XOM region 2 ; 0x20_0030 ~ 0x20_0038 for XOM region 3) via In-System-Programming (ISP) function. When user setting XOM region,need to set Base address first then set XOM page size, as shown in Figure 6.4-3. User should not set XOM at region where chip can boot from.User should not set the XOMs overlap each other, because it would cause XOM can not normal active. After setting and restaring Flash initialization, user can check XOM status from FMC_XOMSTS and check each XOM range (i.e., base and size) from FMC_XOMR0STS~FMC_XOMR3STS. In ICP function or WRITER mode, user can execute command 0x0 to read XOM setting page and check XOM status. User need to do chip reset (Flash initial) after setting or erasing XOM.

When using XOM setting page and XOM regions, users must follow the limitations below:

- All XOM regions must be located in APROM with page size alignment.
- Any XOM region cannot be programmed after its XOM setting is active.
- Any XOM setting cannot be changed during runtime.
- Only use Flash 32-bit program command to program XOM setting page
- Clear XOM setting and XOM region by mass erase command or XOM page erase function (a special erase command for XOM)
- User must restart Flash initialization to activate any new XOM setting.
- ICE can not step in XOM only if debug mode.

Once the XOM region is active, user only can adopt mass erase command or XOM page erase function to clear the XOM setting of one XOM region. To execute XOM page erase function, user must perform FLASH Page Erase (0x22) by writing the base address of the target XOM region (address in APROM, not in XOM setting page) in FMC_ISPADDR and the constant value 0x0055aa03 in FMC_ISPDAT. Note thant, the valid commands in every active XOM region are chip erase, page erase, checksum and read CID/DID. The XOM setting page is not accessed, except for chip erase, word program and 32-bit read.

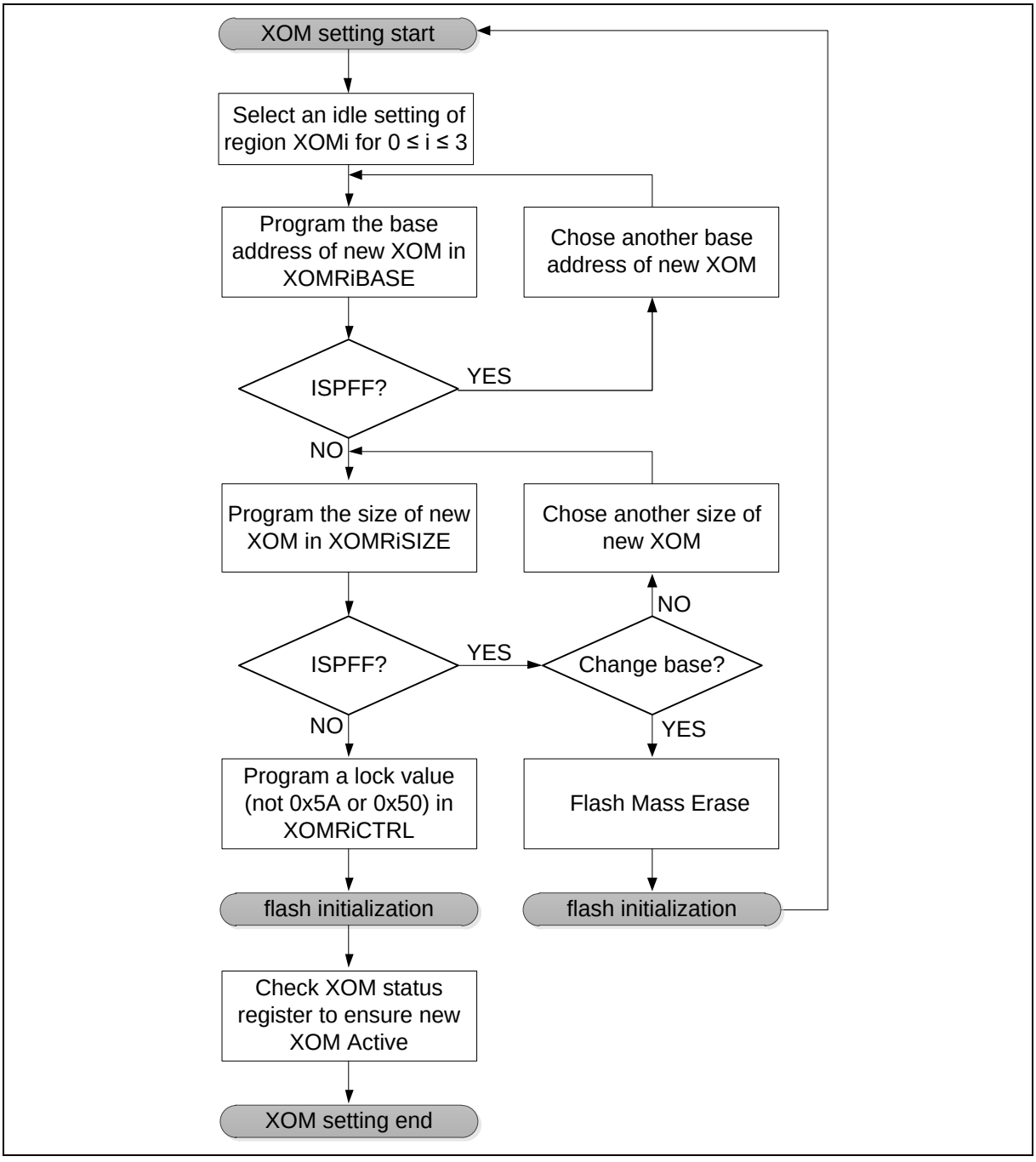


Figure 6.4-3 New XOM Setting Flow

XOMR0BASE (Address = 0x0020_0000)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
XOMR0BASE							
15	14	13	12	11	10	9	8
XOMR0BASE							
7	6	5	4	3	2	1	0
XOMR0BASE							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**XOMR0BASE**	**Base Address of XOM Region 0** XOMR0BASE must be page-aligned. **Note:** The page size is 2KB.

XOMR0SIZE (Address = 0x0020_0004)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
XOMR0SIZE							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**XOMR0SIZE**	**Page Number of XOM Region 0** XOMR0SIZE must be page-aligned and less than 63 pages. The XOMR0SIZE minimum value is 1 page. If XOMR0SIZE is written to 0, the register value will be set to 1. **Note:** The page size is 2KB.

XOMR0CTRL (Address = 0x0020_0008)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
XOMR0CTRL							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**XOMR0CTRL**	**Control of XOM Region 0** 0x5a = XOM region 0 is off. 0x50= XOM is in debug mode. The others = XOM region 0 is active. **Note 1:** But for Flash behavior,user can not write 0 to 1.User need to check the lock value is effective to change 0x5A(0101_1010). **Note 2:** The active state has come into effect after power-on or reset cycle.

XOMR1BASE (Address = 0x0020_0010)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
XOMR1BASE							
15	14	13	12	11	10	9	8
XOMR1BASE							
7	6	5	4	3	2	1	0
XOMR1BASE							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**XOMR1BASE**	**Base Address of XOM Region 1** XOMR1BASE must be page-aligned. **Note:** The page size is 2KB.

XOMR1SIZE (Address = 0x0020_0014)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
XOMR1SIZE							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**XOMR1SIZE**	**Page Number of XOM Region 1** XOMR1SIZE must be page-aligned and less than 63 pages. The XOMR1SIZE minimum value is 1 page. If XOMR1SIZE is written to 0, the register value will be set to 1. **Note:** The page size is 2KB.

XOMR1CTRL (Address = 0x0020_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
XOMR1CTRL							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**XOMR1CTRL**	**Control of XOM Region 1** 0x5a = XOM region 1 is off. 0x50= XOM is in debug mode. The others = XOM region 1 is active. **Note 1:** But for Flash behavior,user can not write 0 to 1.User need to check the lock value is effective to change 0x5A(0101_1010). **Note 2:** The active state has come into effect after power-on or reset cycle.

XOMR2BASE (Address = 0x0020_0020)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
XOMR2BASE							
15	14	13	12	11	10	9	8
XOMR2BASE							
7	6	5	4	3	2	1	0
XOMR2BASE							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**XOMR2BASE**	**Base Address of XOM Region 2** XOMR2BASE must be page-aligned. **Note:** The page size is 2KB.

XOMR2SIZE (Address = 0x0020_0024)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
XOMR2SIZE							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**XOMR2SIZE**	**Page Number of XOM Region 2** XOMR2SIZE must be page-aligned and less than 63 pages. The XOMR2SIZE minimum value is 1 page. If XOMR2SIZE is written to 0, the register value will be set to 1. **Note:** The page size is 2KB.

XOMR2CTRL (Address = 0x0020_0028)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
XOMR2CTRL							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**XOMR2CTRL**	**Control of XOM Region 2** 0x5a = XOM region 2 is off. 0x50= XOM is in debug mode. The others = XOM region 2 is active. **Note 1:** But for Flash behavior,user can not write 0 to 1.User need to check the lock value is effective to change 0x5A(0101_1010). **Note 2:** The active state has come into effect after power-on or reset cycle.

XOMR3BASE (Address = 0x0020_0030)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
XOMR3BASE							
15	14	13	12	11	10	9	8
XOMR3BASE							
7	6	5	4	3	2	1	0
XOMR3BASE							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**XOMR3BASE**	**Base Address of XOM Region 3** XOMR3BASE must be page-aligned. **Note:** The page size is 2KB.

XOMR3SIZE (Address = 0x0020_0034)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
XOMR3SIZE							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**XOMR3SIZE**	**Page Number of XOM Region 3** XOMR3SIZE must be page-aligned and less than 63 pages. The XOMR3SIZE minimum value is 1 page. If XOMR3SIZE is written to 0, the register value will be set to 1. **Note:** The page size is 2KB.

XOMR3CTRL (Address = 0x0020_0038)

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
XOMR3CTRL							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**XOMR3CTRL**	**Control of XOM Region 3** 0x5a = XOM region 3 is off. 0x50= XOM is in debug mode. The others = XOM region 3 is active. **Note 1:** But for Flash behavior,user can not write 0 to 1.User need to check the lock value is effective to change 0x5A(0101_1010). **Note 2:** The active state has come into effect after power-on or reset cycle.

6.4.4.5 Flash Memory Map

The Flash memory map is different from system memory map. The system memory map is used by CPU fetch code or data from FMC memory. The Flash memory map is used for ISP function to read, program or erase FMC memory. The Flash memory map is as Figure 6.4-4.

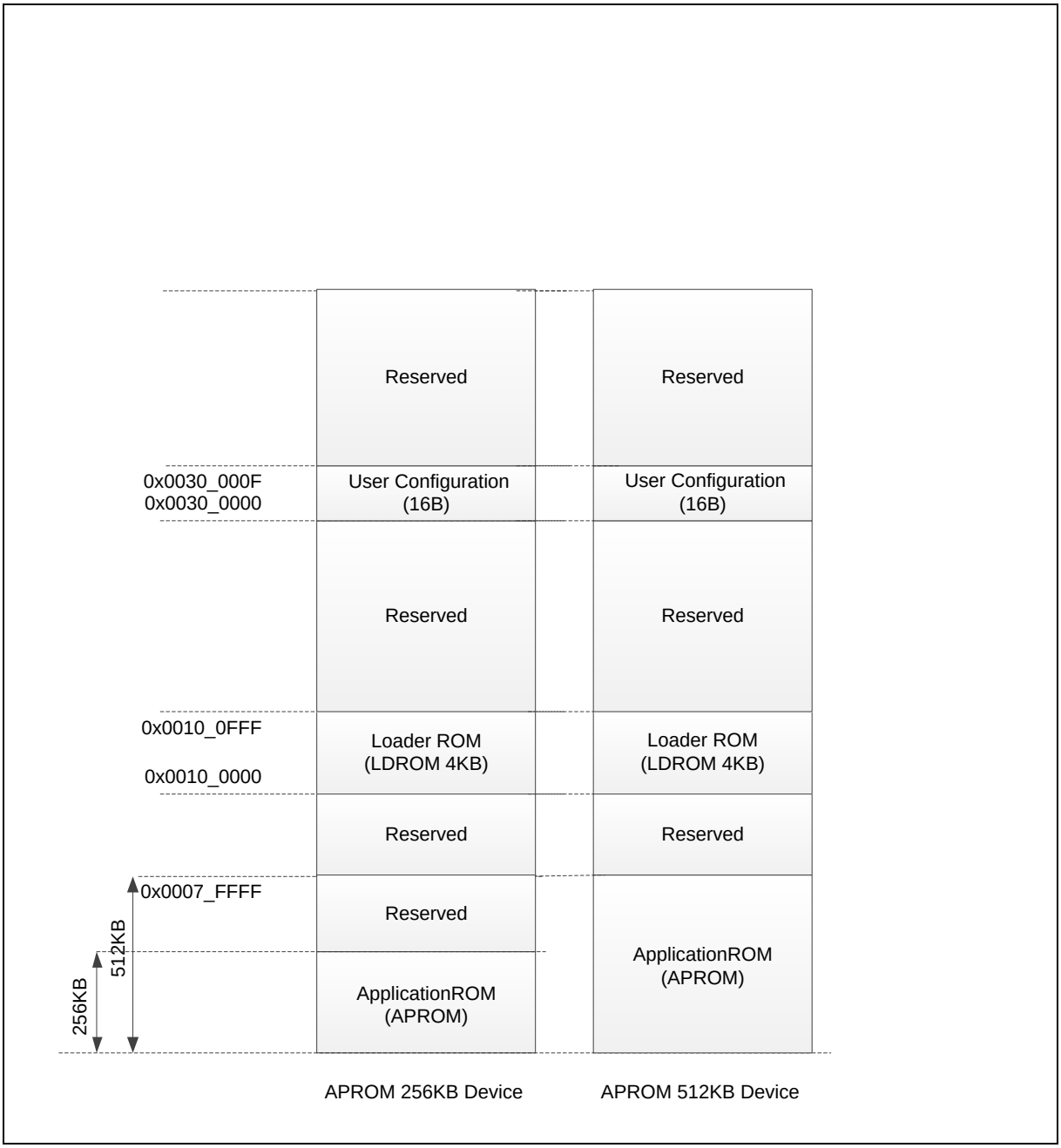


Figure 6.4-4 Flash Memory Map

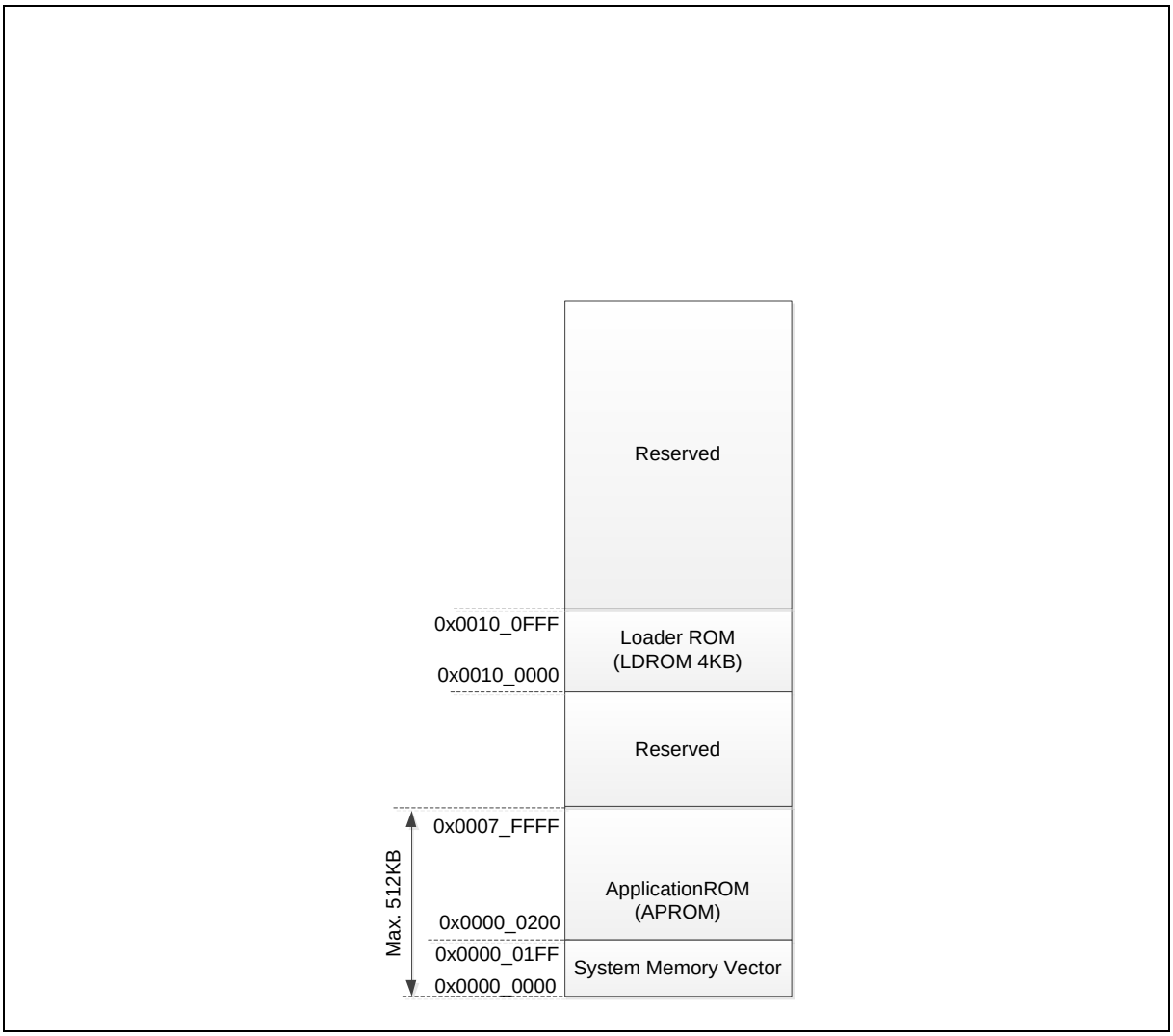
6.4.4.6 System Memory Map with IAP Mode

The system memory map is used by CPU to fetch code or data from FMC memory.And LDROM(0x0010_0000~0x0010_0FFF) address map are the same as in the Flash memory map.

The address from 0x0000_0000 to 0x0000_01FF is called system memory vector.

Figure 6.4-5 shows the system mamory map with IAP mode.

APROM, LDROM can map to the system memory vector for CPU start up. There are two kinds of system memory map with IAP mode when chip booting: (1) LDROM with IAP, (2) APROM with IAP.



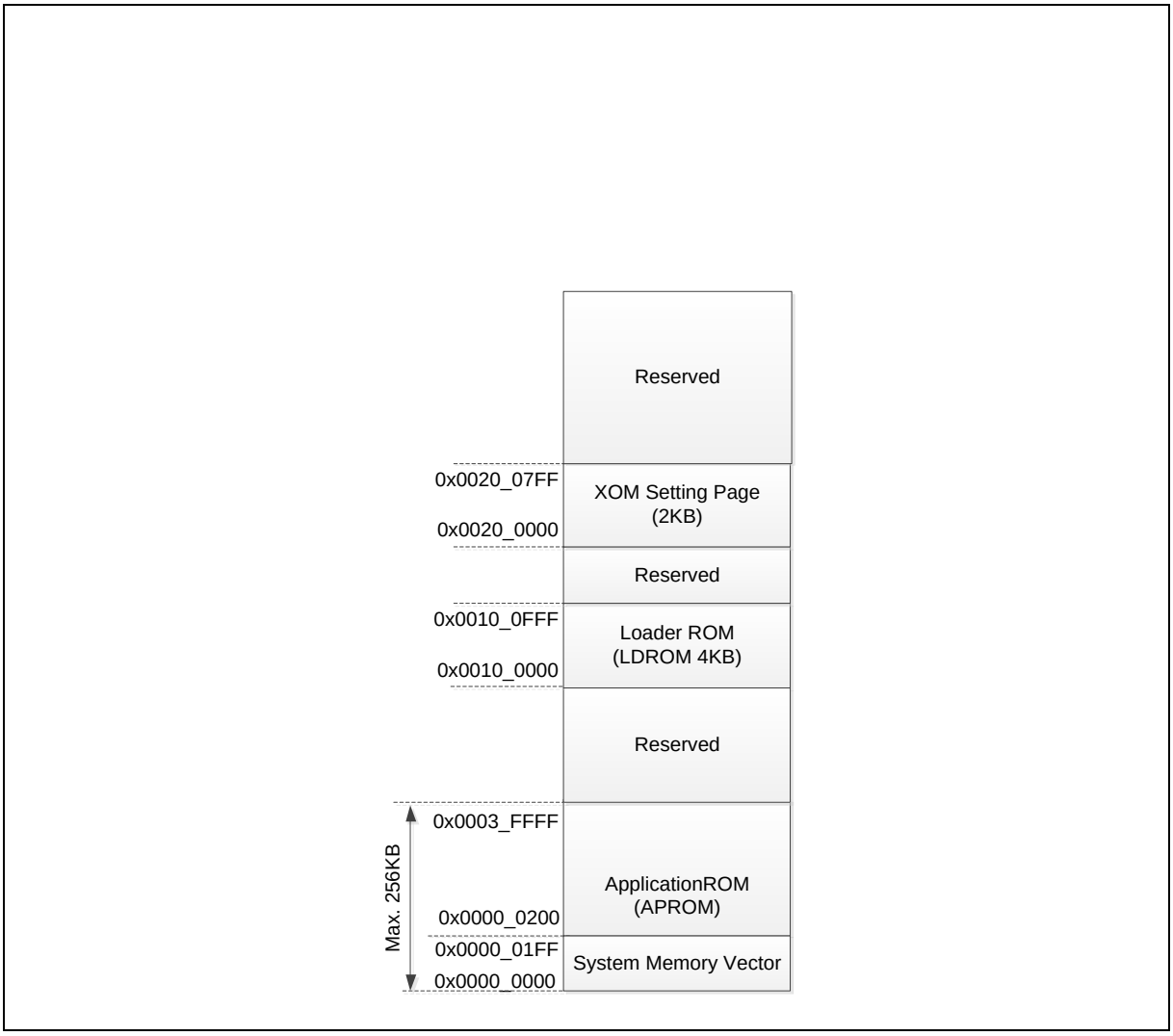


Figure 6.4-5 System Memory Map with IAP Mode

In LDROM with IAP mode, the LDROM (0x0010_0000~0x0010_01FF) is mapping to the system memory vector for Cortex®-M4 instruction or data access. Figure 6.4-6 shows the memory map of LDROM with IAP mode.

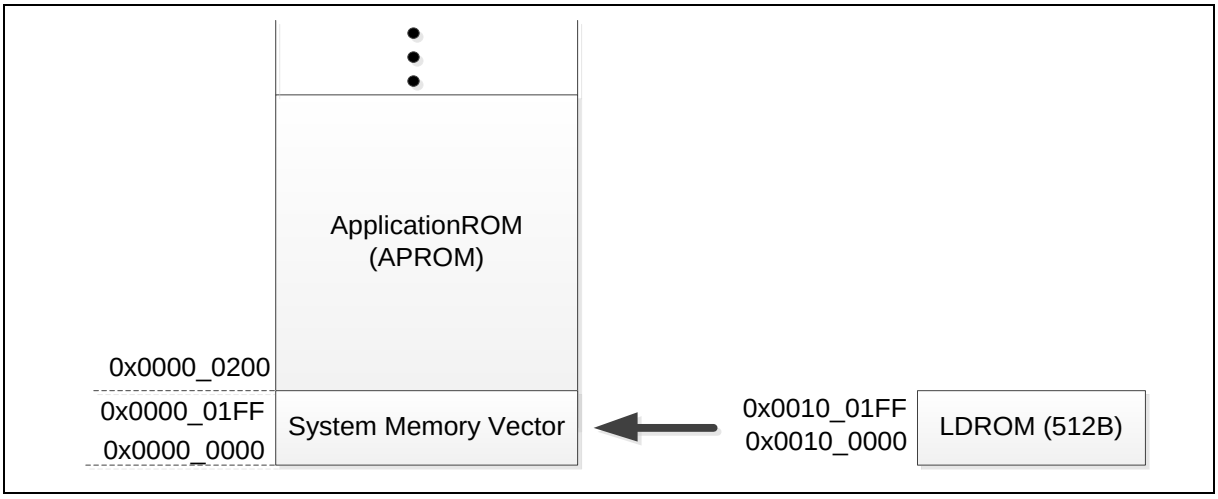


Figure 6.4-6 LDROM with IAP Mode

In APROM with IAP mode, the APROM (0x0000_0000~0x0000_01FF) is mapping to the system memory vector for Cortex®-M4 instruction or data access. Figure 6.4-7 shows the memory map of APROM with IAP mode.

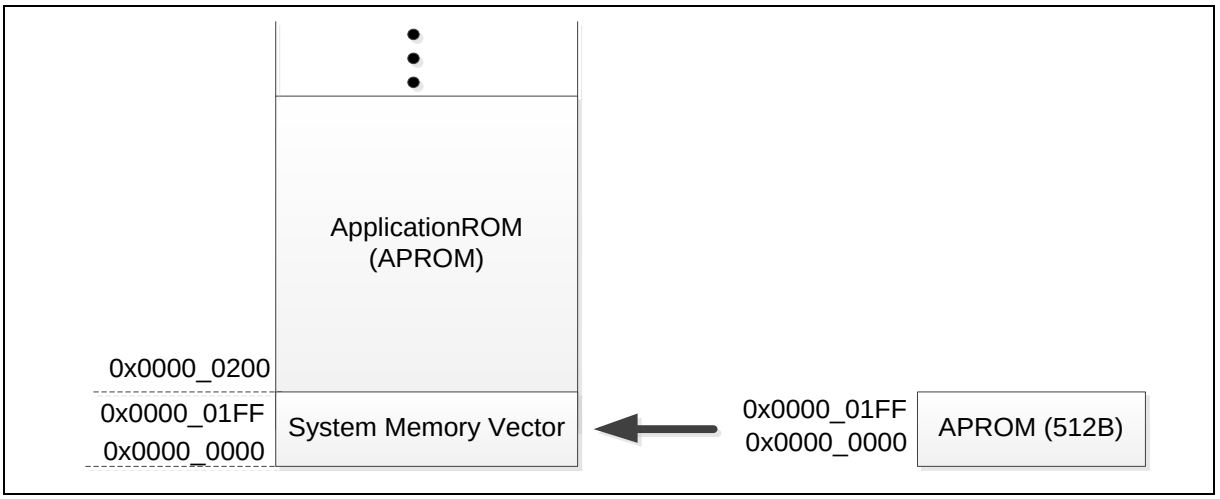


Figure 6.4-7 APROM with IAP Mode

In system memory map with IAP mode, APROM and LDROM can remap to the system memory vector when CPU running. User can write the target remap address to FMC_ISPADDR register and then trigger ISP procedure with the “Vector Remap” command (0x2E). In VECMAP (FMC_ISPSTS[23:9]), shows the finial system memory vector mapping address.

6.4.4.7 System Memory Map Without IAP Mode

There are two kinds of system memory map without IAP mode when chip booting: (1) LDROM without IAP, (2) APROM without IAP. In LDROM without IAP mode, LDROM base is mapping to 0x0000_0000. CPU program cannot run to access APROM. In APROM without IAP mode, APROM base is mapping to 0x0000_0000. CPU program cannot run to access LDROM.

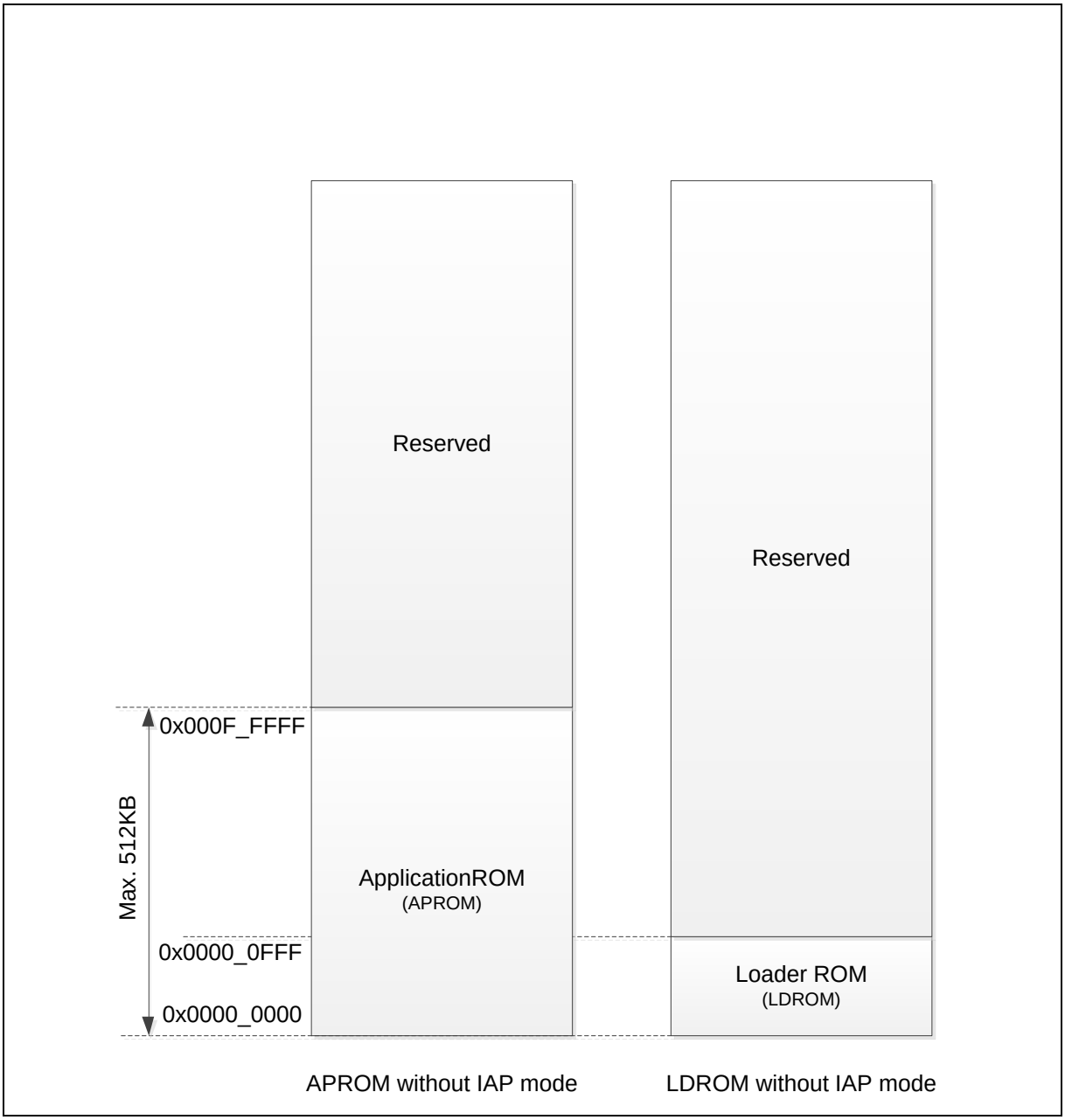


Figure 6.4-8 System Memory Map without IAP Mode

6.4.4.8 Boot Selection

The M471V/M471K/M471C series provides five booting sources for user to select, including LDROM with IAP, LDROM without IAP, APROM with IAP, APROM without IAP. The booting source and system memory map are setting by CBS (CONFIG0[7:6]).

The boot source selection diagram is shown in Figure 6.4-9 and each boot selection support vector mapping is shown in Table 6.4-2.

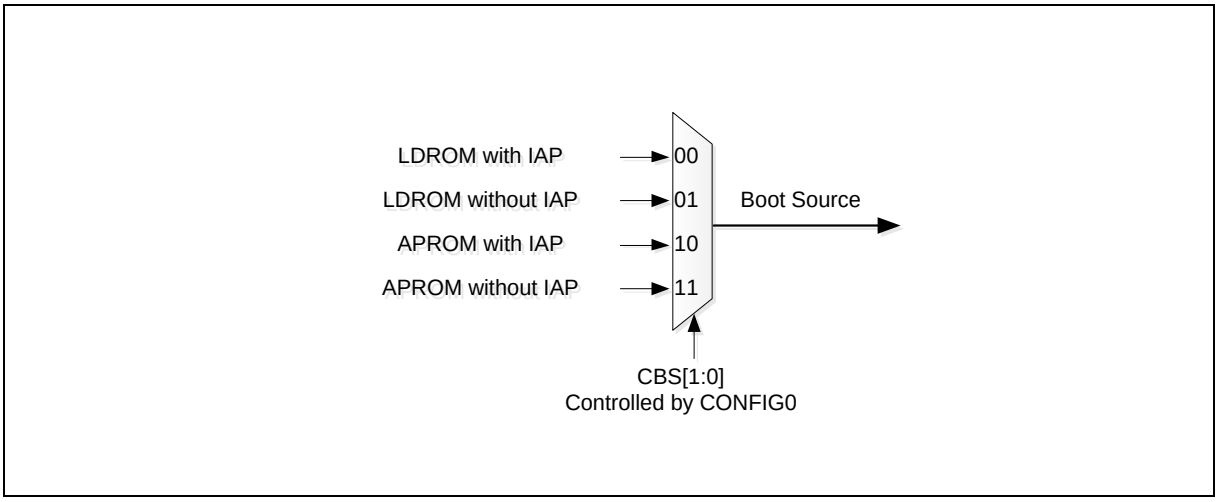


Figure 6.4-9 Boot Source Selection

MBS	CBS[1:0]	Boot Selection/System Memory Map	Vector Mapping Support
1	00	LDROM with IAP	√
1	01	LDROM without IAP	-
1	10	APROM with IAP	√
1	11	APROM without IAP	-

Table 6.4-2 Vector Mapping Support

6.4.4.9 In-Application-Programming (IAP)

The M471V/M471K/M471C series provides In-Application-Programming (IAP) function for user to switch the system memory vector code executing between APROM, LDROM and Boot Loader. User can enable the IAP function by booting chip and setting the chip boot selection bits in in CBS (CONFIG0[7:6]) as 10 or 00, the CBS[0] (FMC_ISPSTS[1]) registers will show 0 and ignore CONFIG0[6] setting).

When chip boots with IAP function is 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.

6.4.4.10 In-System-Programming (ISP)

The M471V/M471K/M471C series supports In-System-Programming (ISP) function allowing the embedded Flash memory to be reprogrammed under software control. ISP is performed without removing the microcontroller from the system through the firmware and on-chip connectivity interface, such as UART, I^2C, and SPI (depended on chip feature). The target Flash memory space that ISP function operates cannot be across banks. Table 6.4-3 lists all ISP commands.

The ISP provides the following functions for embedded Flash memory.

- Supports Flash page erase function
- Supports Flash data program function
- Supports Flash data read function
- Supports company ID read function
- Supports device ID read function
- Supports unique ID read function

- Supports memory CRC32 checksum calculation function
- Supports Flash all one verification function
- Supports system memory vector remap function

ISP Commands

ISP Command	FMC_ISPCMD	FMC_ISPADDR	FMC_ISPDAT FMC_MPDAT0~FMC_MPDAT3
Flash Page Erase	0x22	Valid address of Flash memory organization. It must be page (2 Kbytes) alignment. Note that FMC_ISPADDR[11:0] will be ignored.	N/A
Flash Bank Erase	0x23	Valid address of APROM of the target bank. Note that FMC_ISPADDR[15:0] will be ignored.	N/A
Flash Mass Erase (This command is only valid while MERASE(CONFIG0[13]) bit = 0.)	0x26	0x0000_0000	N/A
Flash 32-bit Program	0x21	Valid address of Flash memory organization It must be 4 bytes word alignment	FMC_ISPDAT :Programming Data
Flash 64-bit Program	0x61	Valid address of Flash memory organization It must be 8 bytes double word alignment	FMC_MPDAT0: LSB Programming Data FMC_MPDAT1: MSB Programming Data
Flash Multi-Word Program	0x27	Valid address of Flash memory organization in APROM and LDROM	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 It must be 4 bytes word alignment	FMC_ISPDAT: Return Data
Flash 64-bit Read	0x40	Valid address of Flash memory organization It must be 8 bytes double word alignment	FMC_ISPDAT: Return Data in FMC_ISPADDR FMC_MPDAT0: Return Data in FMC_ISPADDR FMC_MPDAT1: Return Data in FMC_ISPADDR+4
Read Company ID	0x0B	0x0000_0000	FMC_ISPDAT: 0x0000_00DA
Read Device ID	0x0C	0x0000_0000	FMC_ISPDAT: Return Device ID
		0x0000_0004	FMC_ISPDAT: Return Part ID
		0x0000_0008	FMC_ISPDAT: Return Data Verify Flag

Read CRC32 Checksum	0x0D	0x0000_0000	FMC_ISPDAT: Return Checksum
Run CRC32 Checksum Calculation	0x2D	Valid start address of memory organization It must be 2 Kbytes page alignment	FMC_ISPDAT: Size It must be 4 Kbytes alignment
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.
Run Flash All One Verification	0x28	Valid start address of memory organization It must be 2 Kbytes page alignment	FMC_ISPDAT: Size It must be 4 Kbytes alignment
Read Unique ID	0x04	0x0000_0000	FMC_ISPDAT: Unique ID Word 0
		0x0000_0004	FMC_ISPDAT: Unique ID Word 1
		0x0000_0008	FMC_ISPDAT: Unique ID Word 2
Vector Remap	0x2E	Valid address in APROM,LDROM or boot loader It must be 512 bytes alignment	N/A

Table 6.4-3 ISP Command List

ISP Procedure

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

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

Once the FMC_ISPCTL register is set properly, user can set FMC_ISPCMD (refer to Table 6.4-3 ISP command list) for specify operation. Set FMC_ISPADDR for target Flash memory based on Flash memory organization. FMC_ISPDAT can be used to set the data to program or used to return the read data according to FMC_ISPCMD. The ISP procedure flow is shown in Figure 6.4-10.

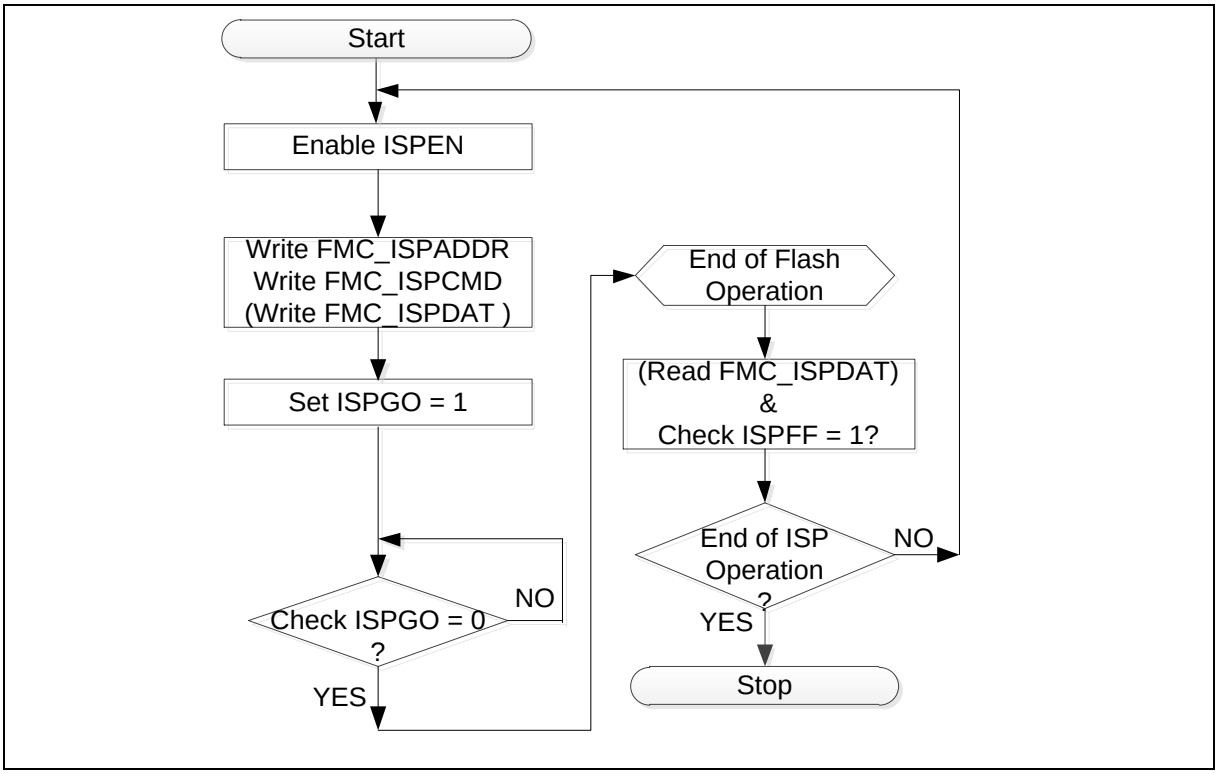


Figure 6.4-10 ISP Procedure Example

Finally, set the ISPGO (FMC_ISPTRG[0]) register to perform the relative ISP function. When ISP function is active, the ISPBUSY(FMC_ISPSTS[0]) and MPBUSY(FMC_MPSTS[0]) be set to 1. The ISPGO(FMC_ISPTRG[0]), ISPBUSY(FMC_ISPSTS[0]) and MPBUSY (FMC_MPSTS[0]) are self-cleared when ISP function has been done.

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

When the ISPGO(FMC_ISPTRG[0]) bit is set and then CPU access the same bank, CPU will wait for ISP operation to finish during this period; the peripheral still keeps working as usual. If any interrupt request occurs, CPU will not service it till ISP operation is finished. When ISP operation is finished, the ISPGO bit will be cleared by hardware automatically. User can check whether ISP operation is finished or not by the ISPGO(FMC_ISPTRG[0]) bit.

When CPU access operation and ISP command are executed in different bank, CPU and ISP command can operate in parallel.

6.4.4.11 Embedded Flash Memory Programming

The M471V/M471K/M471C series provides 32-bit, 64-bit and multi-word Flash memory programming function to speed up Flash updated procedure. Table 6.4-4 lists required FMC control registers in each embedded Flash programming function.

Register	Description	32-Bit Programming	64-Bit Programming	Multi-Word Programming
FMC_ISPCTL	ISP Control Register	√	√	√

FMC_ISPADDR	ISP Address Register	√	√	√
FMC_ISPDAT	ISP Data Register	√	N/A	N/A
FMC_ISPCMD	ISP Command Register	0x21	0x61	0x27
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 FMC Control Registers for Flash Programming

64-bit Programming

The M471V/M471K/M471C series 64-bit programming function is faster than 32-bit programming. FMC_ISPDAT is used for 32-bit programming data register. In 64-bit programming, there are two programming data registers, one is FMC_MPDAT0 for LSB word, and the other is FMC_MPDAT1 for MSB word, and ISP command is 0x61, the other registers are the same as 32-bit programming. Figure 6.4-11 and Figure 6.4-12 are the ISP 32-bit / 64-bit programing procedure flow.

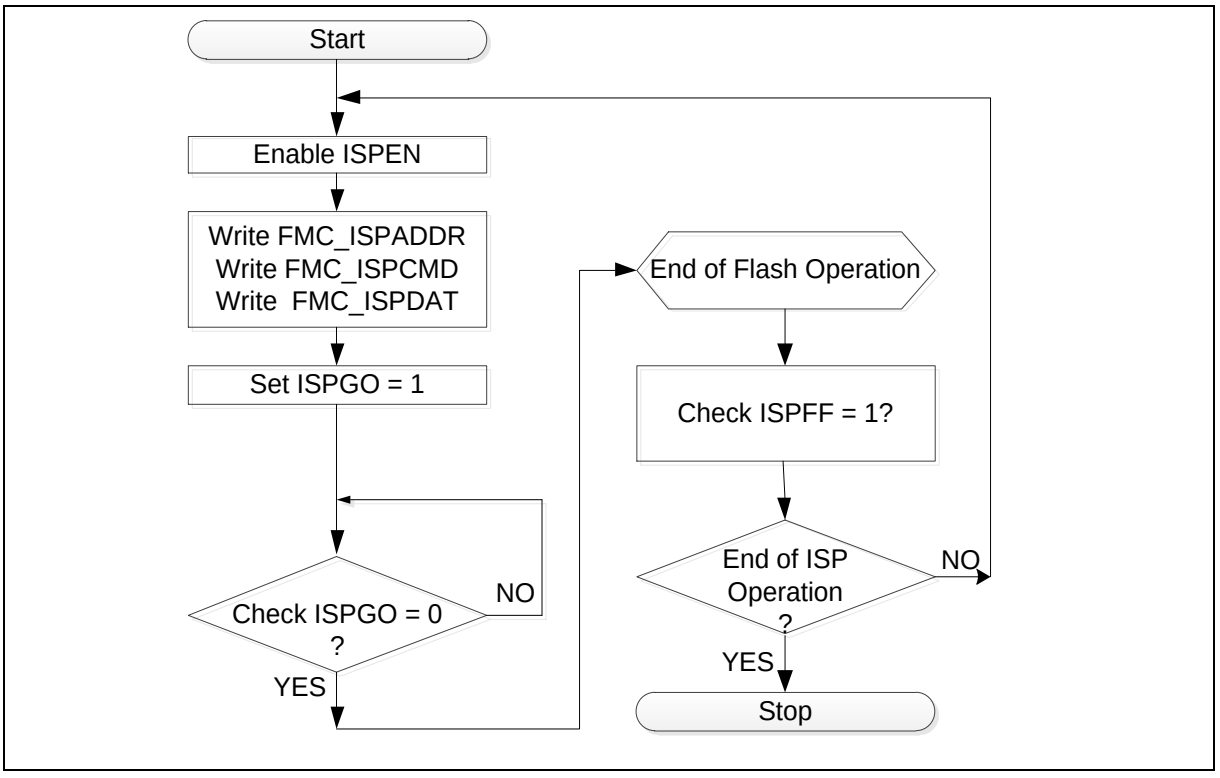


Figure 6.4-11 ISP 32-bit Programming Procedure

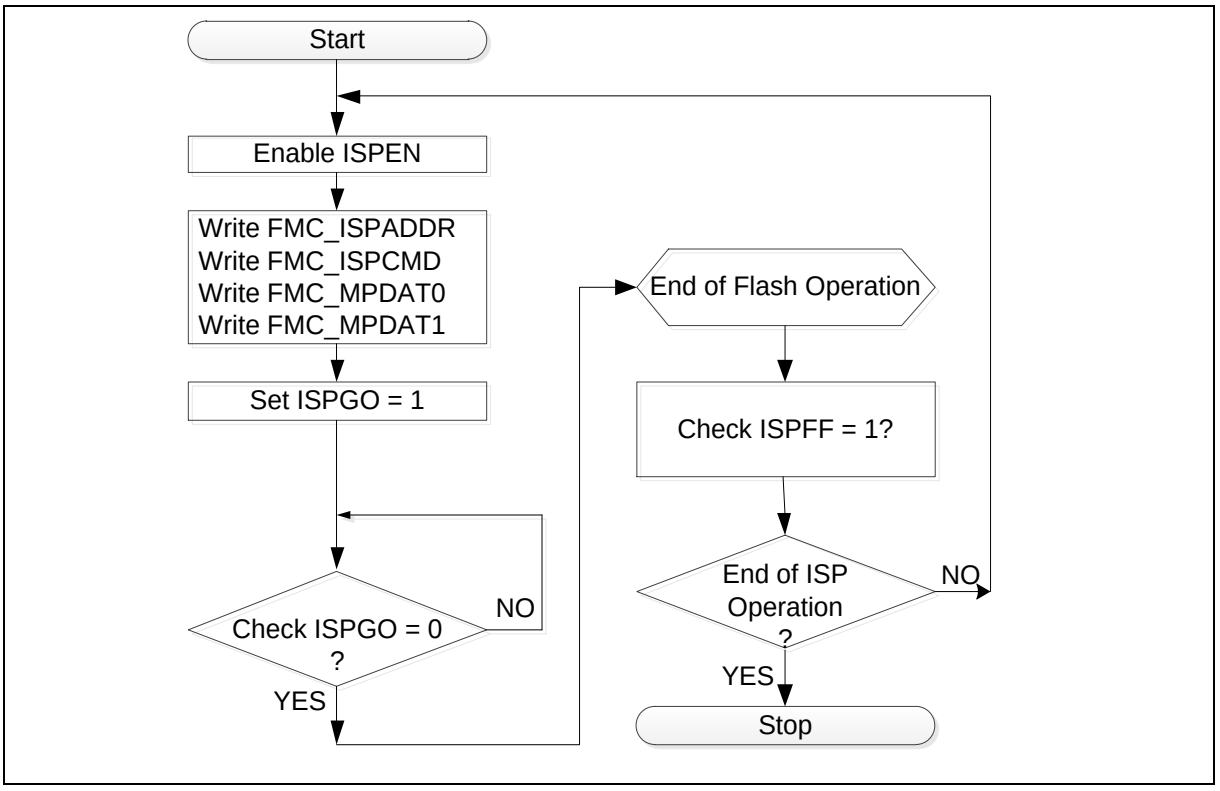


Figure 6.4-12 ISP 64-bit Programming Procedure

Multi-word Programming

The M471V/M471K/M471C series supports multi-word programming function to speed up Flash updated procedure. The maximum programming length is up to 256 bytes, and the minimum programming length is 8 bytes (2 words). The multi-word programming is the fastest programming function if the programming words more than 8 bytes, because only one set of Flash setup time and hold time needed for one time operation. Figure 6.4-13 compares each programming time.

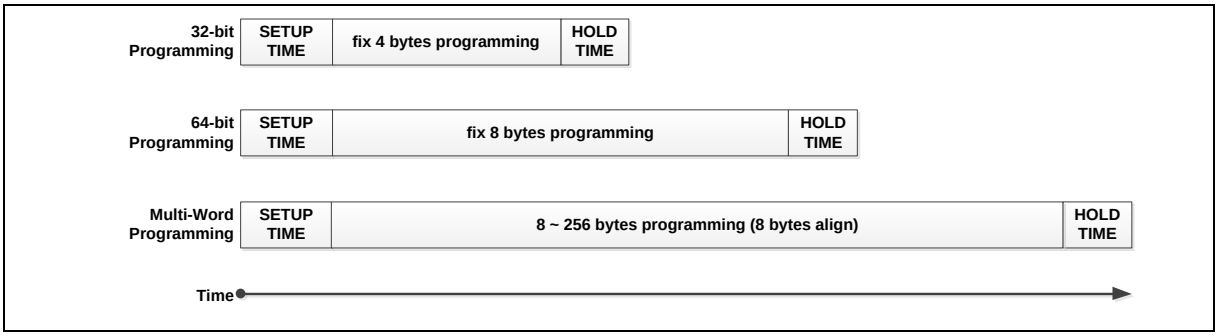


Figure 6.4-13 Multi-word Programming Time

In multi-word programming operation, Cortex®-M4 CPU has to monitor the empty status of the programming buffer. CPU has to prepare the next data for programming continuity. The multi-program firmware should not be located in APROM or LDROM, because CPU instruction fetch cannot be hold. The firmware has to be located in Boot loader or embedded SRAM of chip to avoid CPU hold.

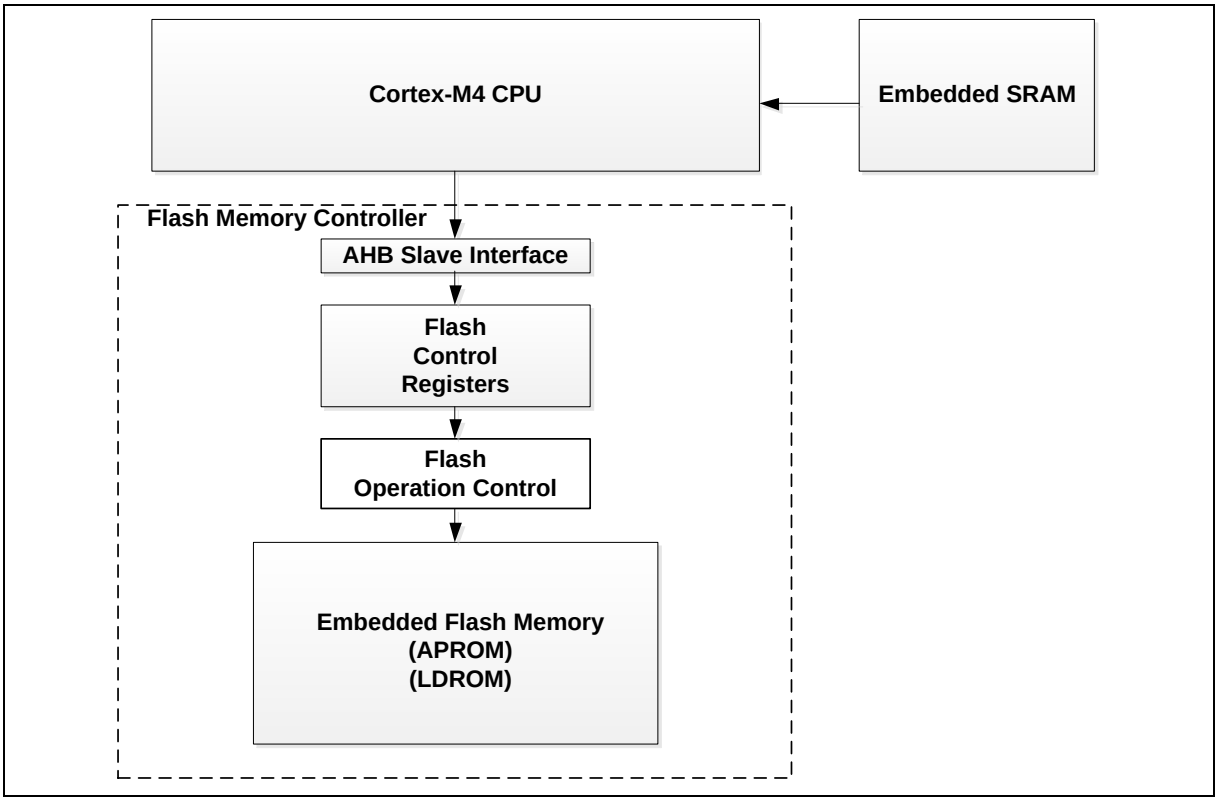


Figure 6.4-14 Firmware in SRAM for Multi-word Programming

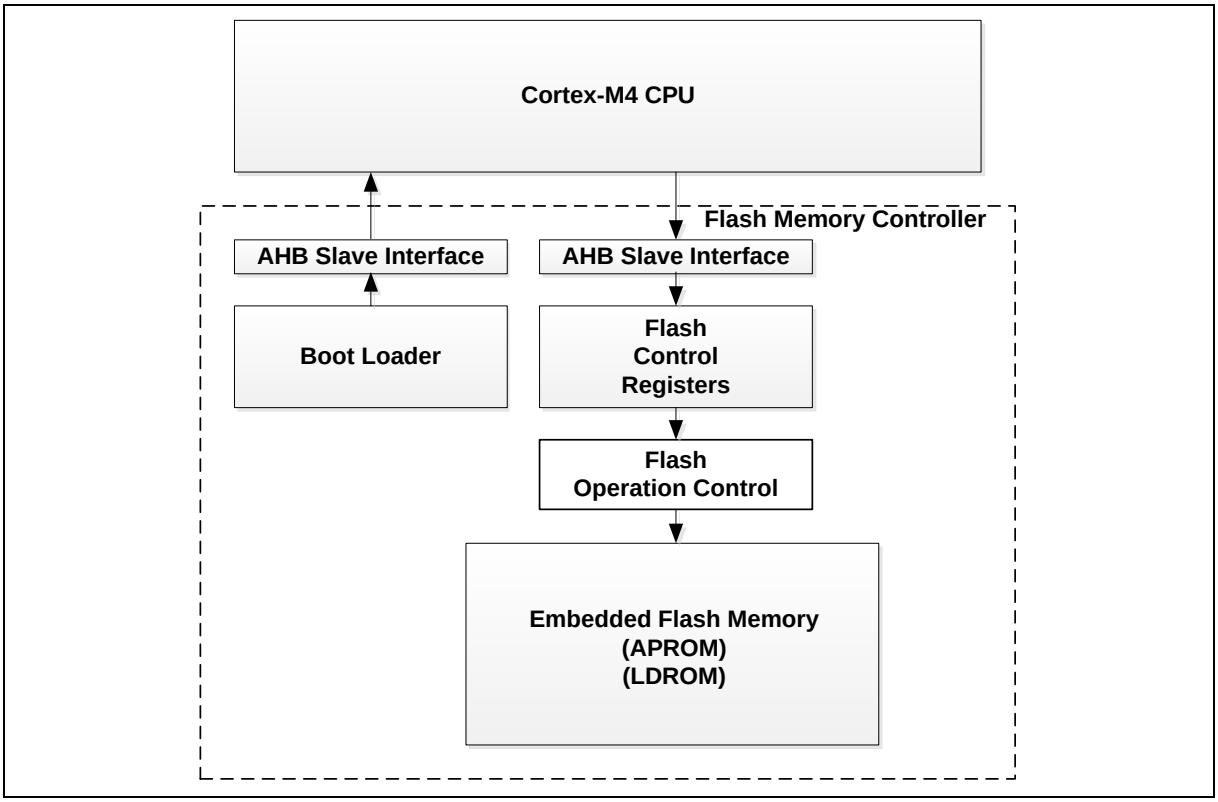


Figure 6.4-15 Firmware in Boot Loader for Multi-word Programming

The multi-word programming flow is shown in Figure 6.4-16. The starting ISP address (FMC_ISPADDR) has to be 8-byte align, FMC_ISPADDR[2:0] should be 0. ISPDAT0(FMC_MPDAT0) is the data word of the offset 0x0, ISPDAT1(FMC_MPDAT1) is the second word (offset 0x4), ISPDAT2(FMC_MPDAT2) is the third word (offset 0x8), and ISPDAT3(FMC_MPDAT3) is forth word (offset 0xC). If the starting ISP address FMC_ISPADDR [3] is 0, the 1st data word should put on ISPDAT0, and 2nd word is ISPDAT1, 3rd word is ISPDAT2, and 4th word is ISPDAT3. If the starting ISP address FMC_ISPADDR [3] is 1, the 1st data word should put on ISPDAT2, and 2nd word is ISPDAT3, 3rd word is ISPDAT0, and 4th word is ISPDAT1. The maximum programming size is 256 bytes and align to 256-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 (ISPDAT0, ISPDAT1, ISPDAT2 and ISPDAT3) 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.

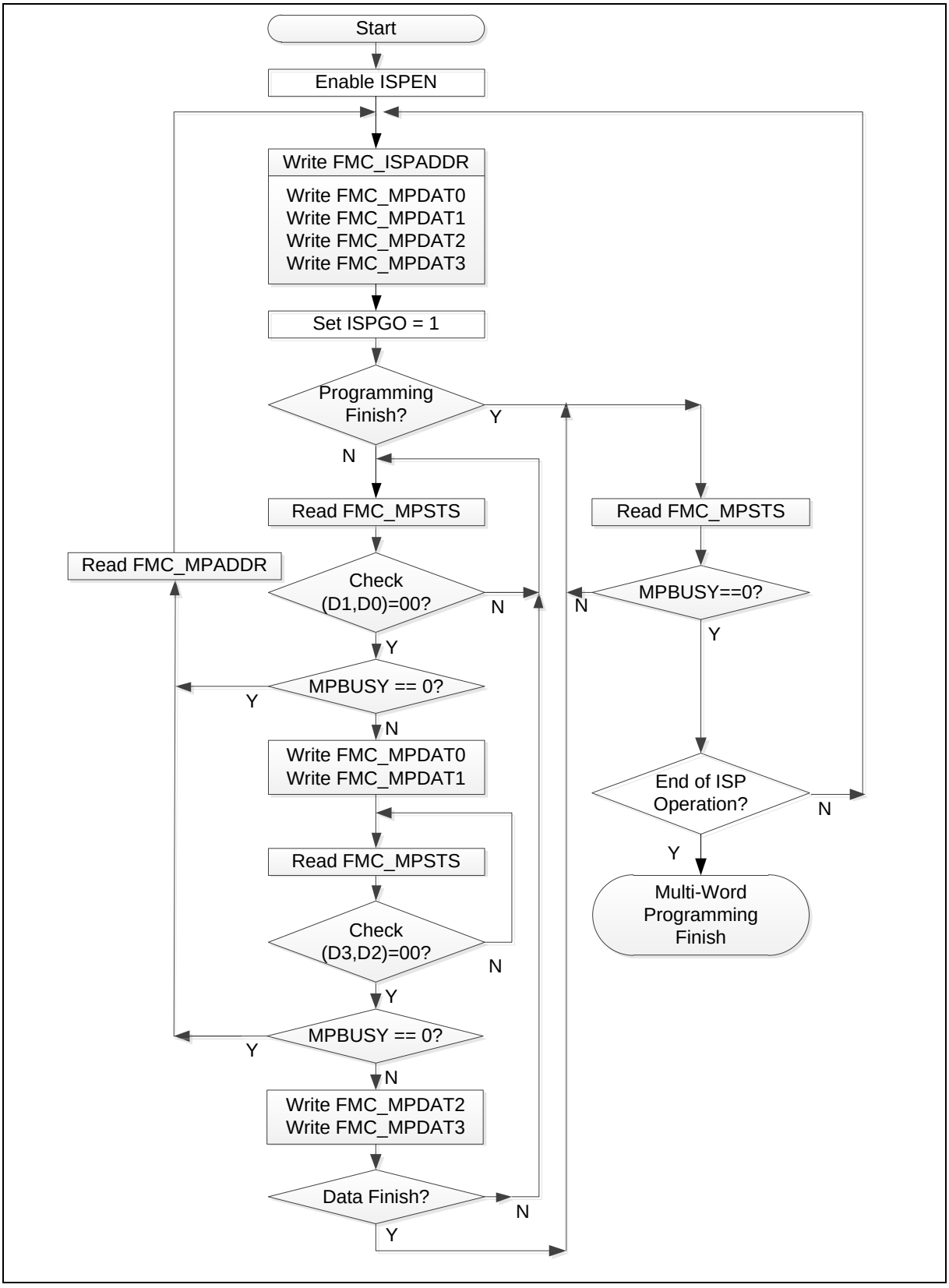


Figure 6.4-16 Multi-word Programming Flow

6.4.4.12 Fast Flash Programming Verification

In traditional Flash programming operation, the controller receives the programming trigger event then control the timing to perform the programming embedded Flash memory as show in Figure 6.4-17.

The M471V/M471K/M471C series supports the fast Flash programming verification function, which provides hardware verification for Flash programming to save time of the CPU read back and comparison. When data is programmed to the embedded Flash memory, the controller asserts the Flash read operation to read data out, and performs data comparison with data in. Finally, the comparison result is saved in PGFF (FMC_ISPSTS[5]). The PGFF is set to 1 if output data is not the same as the input programming data. The flag is kept until clear by software or a new erase operation. The fast Flash programming verification flow is shown in Figure 6.4-17.

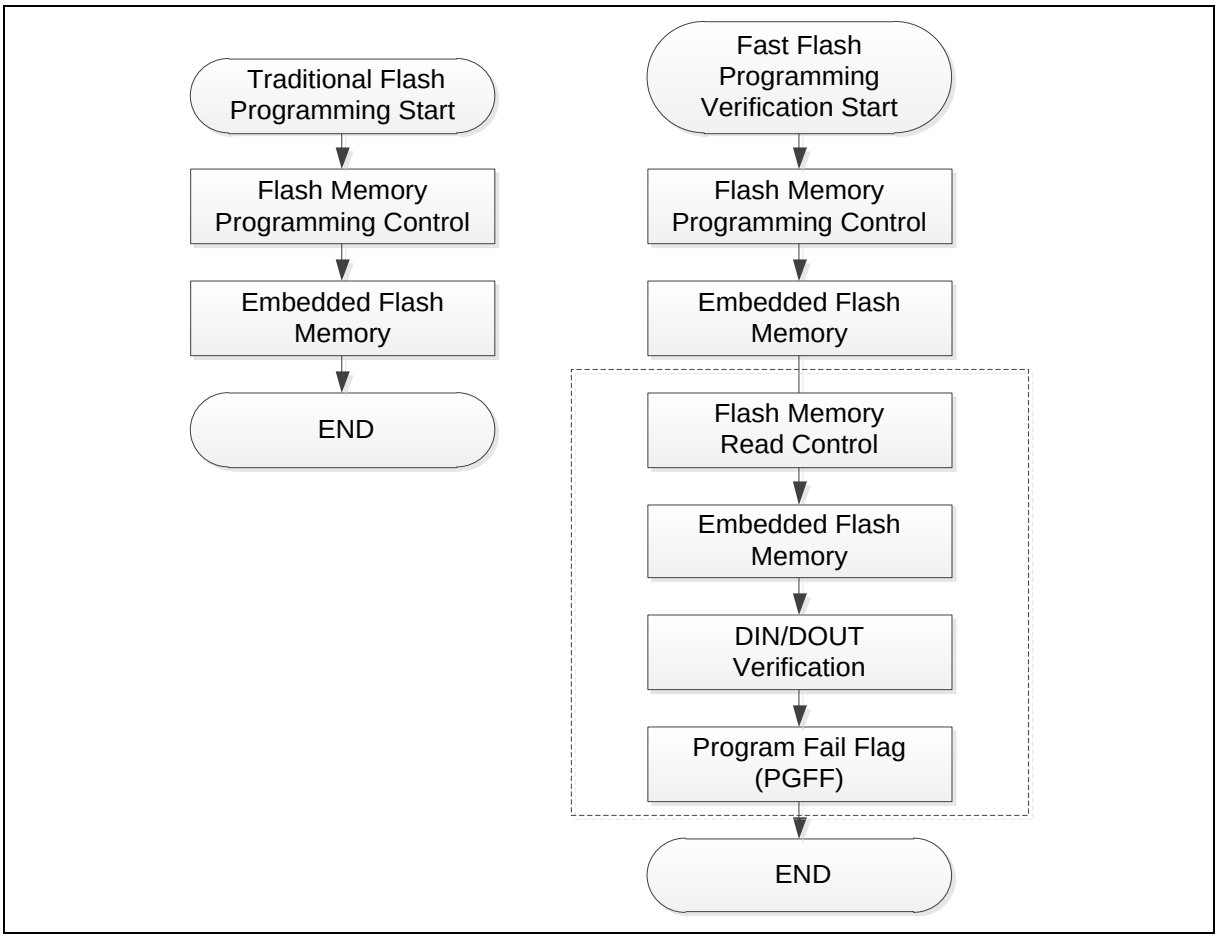


Figure 6.4-17 Fast Flash Programming Verification Flow

In traditional Flash updated operation, the Flash memory has to perform three steps to complete the Flash memory updated procedure, (1) Flash ERASE, (2) Flash PROGRAM, and (3) Flash READ back all of data to check the correction. In the M471V/M471K/M471C series, it only reads FMC_ISPSTS to check PGFF flag in Step (3) without reading data back to confirm. Figure 6.4-18 compares traditional programming verification flow and fast programming verification flow.

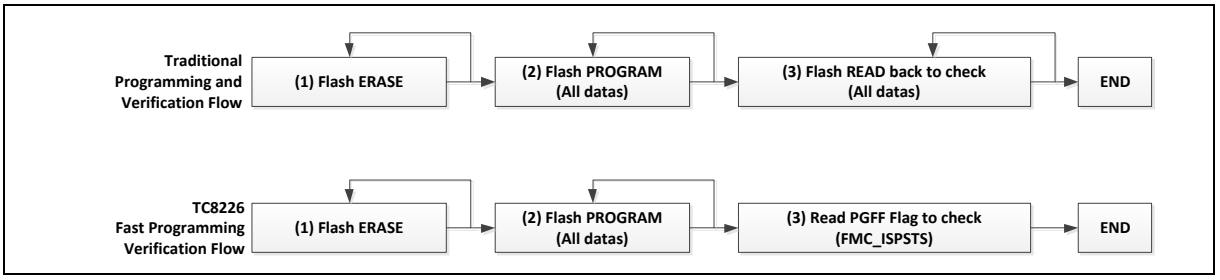


Figure 6.4-18 Verification Flow

The fast Flash programming verification function is released for 32-bit programming and 64-bit programming operation, but multi-word programming operation is not suitable due to the embedded Flash HV (High Voltage) of continue programming.

6.4.4.13 CRC32 Checksum Calculation

The M471V/M471K/M471C series supports the CRC32 checksum calculation function to help user quickly check the memory content includes APROM, LDROM and Boot Loader. 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 6.4-19.

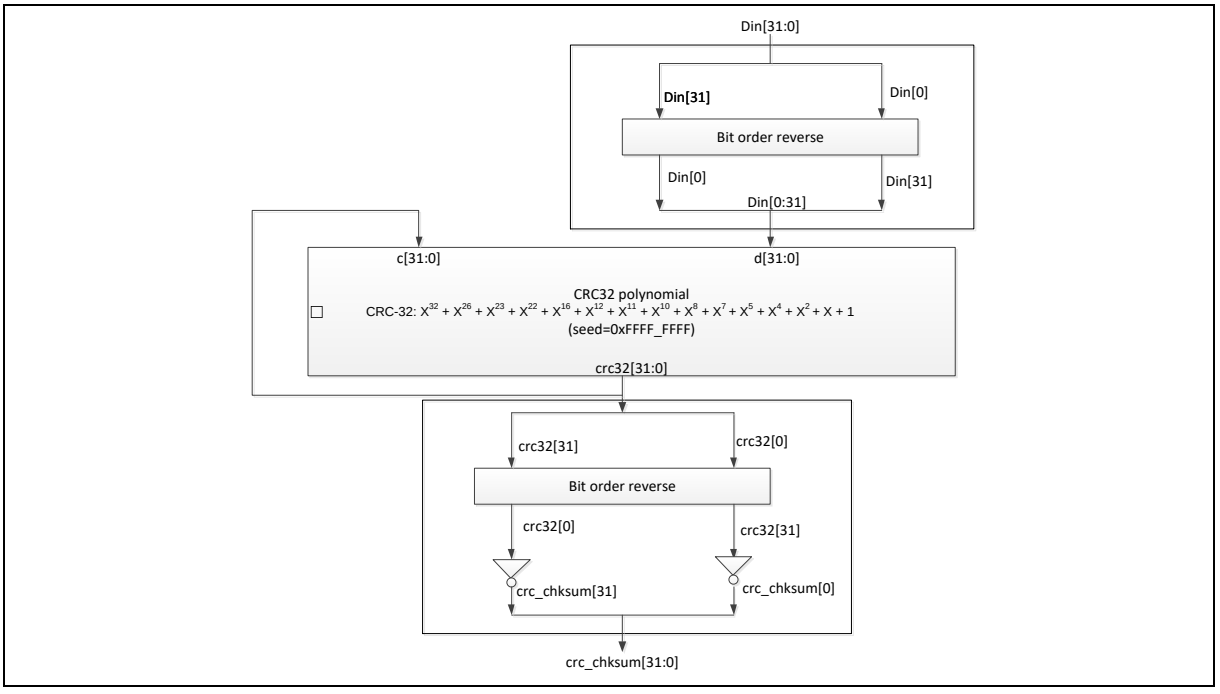


Figure 6.4-19 Flash CRC32 Checksum Calculation

Three steps complete this CRC32 checksum calculation.

1. Step 1.Perform ISP “Run Memory CRC32 Checksum” operation
2. Step 2.Perform ISP “Read Memory CRC32 Checksum” operation
3. 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 2 Kbytes alignment, the size should be ≥ 2 Kbytes and the starting address includes APROM, LDROM and Boot Loader.

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

Note that the range of CRC32 checksum cannot cross bank in one operation.

6.4.4.14 Flash All One Verification

The M471V/M471K/M471C series supports the Flash all one verification function to help user quickly check a memory block content blanking for APROM and LDROM after Flash erase operation.

Two or Three steps complete this Flash all one verification.

Two-step flow:

1. Step 1.Perform ISP “Run Flash All One Verification” operation
2. 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:

1. Step 1.Perform ISP “Run Flash All One Verification” operation
2. Step 2.Perform ISP “Read Flash All One Result” operation
3. 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

In Step 1, user has to set the memory starting address (FMC_ISPADDR) and size (FMC_ISPDAT) to verify. Both address and size have to be 2 Kbytes alignment, the size should be ≥ 2 Kbytes and the starting address includes APROM and LDROM.

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

Note that the range of “Run All One Verification” cannot cross bank in one operation.

6.4.4.15 Flash Access Cycle

The HCLK detected frequency and optimized CYCLE number is shown in Table 6.4-5.

Because HIRC has inaccuracy (12 MHz ± 2%) and the relationship of HCLK and HIRC is asynchronous, FMC may generate two possible optimized cycle numbers in some HCLK Clock Frequency.

HCLK Clock Frequency	Optimized CYCLE Number
0 MHz ~20 MHz	1
21 MHz~40 MHz	2
41 MHz~60 MHz	3
61 MHz~80 MHz	4
81 MHz~100 MHz	5
151 MHz~194 MHz	6
>195 MHz	7

Table 6.4-5 Flash Access Optimized Cycle under Auto-tuning Function

6.4.4.16 Cache Memory Controller

The cache memory controller will store each recent Flash data access on APROM and LDROM region to boost performance for later read access on the same address. The address space adopted for the cache memory controller is the same as CPU. The cache memory controller has snooping mechanism to monitor events that will change the contents of Flash. The monitored ISP commands are program-related, erase-related and VECMAP. For program-related ISP commands, if the data of the target address is already in the cache memory, the corresponding entry will be invalidated, while other entries data will remain unchanged. For erase-related and VECMAP ISP commands, all cache contents will be invalidated. Since the address space for CPU is dependent on boot mode and VECMAP setting, when VECMAP is active or the boot mode is other than APROM, the address space for CPU and ISP command (ISPADDR) will be different. Thus, if there's demand for using program-related ISP commands when either VECMAP is active or boot mode is other than APROM, the snooping mechanism can not detect successfully. In such scenarios, it's necessary to manually send invalidation command (by writing CACHEINV in FMC_FTCTL) afterwards.

6.4.4.17 Lock Effect Tables

The M471V/M471K/M471C series supports four kinds of protections include the Security Lock Control (i.e., LOCK in CONFIG0[1] and ALOCK in CONFIG2[7:0]). In this section, the M471V/M471K/M471C prepares some lock effect tables for user to understand the lock effects on APROM, LDROM, CONFIG and others with above four protections for CPU, ICE and ICP/Writer.

LOCK/ALOCK	OFF		ON	
APROM	R	W	R	W
Data Flash	R	W	R	W
LDROM	R	W	R	W
CFG	R	W	R	W
DID/UID/Checksum	R	-	R	-

Table 6.4-6 Lock Effect Table with Four Protections for CPU

Note:

1. The symbol “R” means readable and the symbol “W” means writable.
2. The symbol “X” means no matter what value is.

6.4.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
FMC Base Address **FMC_BA = 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 Control Register	0x0000_0000
FMC_ISPSTS	FMC_BA+0x40	R/W	ISP Status Register	0x0000_0000
FMC_CYCCTL	FMC_BA+0x4C	R/W	Flash Access Cycle Control Register	0x0000_0000
FMC_MPDAT0	FMC_BA+0x80	R/W	ISP Data0 Register	0x0000_0000
FMC_MPDAT1	FMC_BA+0x84	R/W	ISP Data1 Register	0x0000_0000
FMC_MPDAT2	FMC_BA+0x88	R/W	ISP Data2 Register	0x0000_0000
FMC_MPDAT3	FMC_BA+0x8C	R/W	ISP Data3 Register	0x0000_0000
FMC_MPSTS	FMC_BA+0xC0	R	ISP Multi-program Status Register	0x0000_0000
FMC_MPADDR	FMC_BA+0xC4	R	ISP Multi-program Address Register	0x0000_0000
FMC_XOMR0STS	FMC_BA+0xD0	R	XOM Region 0 Status Register	0x0FF8_00FF
FMC_XOMR1STS	FMC_BA+0xD4	R	XOM Region 1 Status Register	0x0FF8_00FF
FMC_XOMR2STS	FMC_BA+0xD8	R	XOM Region 2 Status Register	0x0FF8_00FF
FMC_XOMR3STS	FMC_BA+0xDC	R	XOM Region 3 Status Register	0x0FF8_00FF
FMC_XOMSTS	FMC_BA+0xE0	R	XOM Status Register	0x0000_0000

6.4.6 Register Description

ISP Control Register (FMC_ISPCTL)

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

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

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**INTEN**	**Secure ISP INT Enable Bit (Write Protect)** 0= ISP INT Disabled. 1= ISP INT Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.Before using INT,user needs to clear the INTFLAG(FMC_ISPSTS[24]) make sure INT happen at correct time.
[23:8]	**Reserved**	Reserved.
[7]	**Reserved**	Reserved.
[6]	**ISPFF**	**ISP Fail Flag (Write Protect)** This bit is set by hardware when a triggered ISP meets any of the following conditions: This bit needs to be cleared by writing 1 to it. ● APROM writes to itself if APUEN is set to 0. ● LDROM writes to itself if LDUEN is set to 0. ● CONFIG is erased/programmed if CFGUEN is set to 0. ● Erase or Program command at brown-out detected ● Destination address is illegal, such as over an available range. ● Invalid ISP commands ● The base and size of new XOM regions is wrong, overlap or writed twice ● The input setting of XOM page erase function is wrong ● The active XOM regions is accessed (except for chip erase, page erase, checksum and read CID/DID) ● The XOM setting page is accessed (except for chip erase, word program and read) ● Violate the load code read protection ● Checksum or Flash All One Verification is not executed in their valid range ● Bank erase is not executed in APROM **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

[5]	**LDUEN**	**LDROM Update Enable Bit (Write Protect)** 0 = LDROM cannot be updated. 1 = LDROM can be updated. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[4]	**CFGUEN**	**CONFIG Update Enable Bit (Write Protect)** 0 = CONFIG cannot be updated. 1 = CONFIG can be updated. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[3]	**APUEN**	**APROM Update Enable Bit (Write Protect)** 0 = APROM cannot be updated when the chip runs in APROM. 1 = APROM can be updated when the chip runs in APROM. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[2]	**Reserved**	Reserved.
[1]	**BS**	**Boot Select (Write Protect)** When MBS in CONFIG0 is 1, set/clear this bit to select next booting from LDROM/APROM, respectively. This bit also functions as chip booting status flag, which can be used to check where chip booted from. This bit is initiated with the inversed value of CBS[1] (CONFIG0[7]) after any reset is happened except CPU reset (CPU is 1) or system reset (SYS) is happened 0 = Boot from APROM when MBS (CONFIG0[5]) is 1. 1 = Boot from LDROM when MBS (CONFIG0[5]) is 1. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[0]	**ISPEN**	**ISP Enable Bit (Write Protect)** ISP function enable bit. Set this bit to enable ISP function. 0 = ISP function Disabled. 1 = ISP function Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

ISP Address (FMC_ISPADDR)

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

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

Bits	Description	
[31:0]	**ISPADDR**	**ISP Address** The M471V/M471K/M471C series is equipped with embedded Flash. ISPADDR[1:0] must be kept 00 for ISP 32-bit operation. ISPADDR[2:0] must be kept 000 for ISP 64-bit operation. For CRC32 Checksum Calculation command, this field is the Flash starting address for checksum calculation, 2 Kbytes alignment is necessary for CRC32 checksum calculation. For Flash32-bit Program, ISP address needs word alignment (4-byte). For Flash 64-bit Program, ISP address needs double word alignment (8-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 2 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.
[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 any page in two banks. 0x23= FLASH Bank Erase. Erase all pages of APROM in BANK0 or BANK1. 0x26= FLASH Mass Erase. Erase all pages in two banks. 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 Control Register	0x0000_0000

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

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

ISP Status Register (FMC_ISPSTS)

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

31	30	29	28	27	26	25	24
Reserved							INTFLAG
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	
[31:25]	**Reserved**	Reserved.
[24]	**INTFLAG**	**ISP Interuppt Flag** 0 = ISP Not Finished. 1 = ISP done or ISPFF set. **Note:** This function needs to be enabled by FMC_ISPCTRL[24].
[23:9]	**VECMAP**	**Vector Page Mapping Address (Read Only)** All access to 0x0000_0000~0x0000_01FF is remapped to the Flash memory address {VECMAP[14:0], 9'h000} ~ {VECMAP[14:0], 9'h1FF}
[8]	**Reserved**	Reserved.
[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 = Flash bits are not all 1 after "Run Flash All-One Verification" complete. 1 = All of Flash bits are 1 after "Run Flash All-One Verification" complete.

[6]	**ISPFF**	**ISP Fail Flag (Write Protect)** This bit is the mirror of ISPFF (FMC_ISPCTL[6]), it needs to be cleared by writing 1 to FMC_ISPCTL[6] or FMC_ISPSTS[6]. This bit is set by hardware when a triggered ISP meets any of the following conditions: ● APROM writes to itself if APUEN is set to 0. ● LDROM writes to itself if LDUEN is set to 0. ● CONFIG is erased/programmed if CFGUEN is set to 0. ● Erase or Program command at brown-out detected ● Destination address is illegal, such as over an available range. ● Invalid ISP commands ● The base and size of new XOM regions is wrong, overlap or writed twice ● The input setting of XOM page erase function is wrong ● The active XOM regions is accessed (except for chip erase, page erase, checksum and read CID/DID) ● The XOM setting page is accessed (except for chip erase, word program and read) ● Violate the load code read protection ● Checksum or Flash All One Verification is not executed in their valid range ● Bank erase is not executed in APROM **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[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 clear by performing ISP Flash erase or ISP read CID operation 0 = Flash Program is success. 1 = Flash Program is fail. Program data is different with data in the Flash memory
[4]	**FCYCDIS**	**Flash Access Cycle Auto-tuning Disable 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 Enabled. 1 = Flash access cyle auto-tuning Disabled.
[3]	**Reserved**	Reserved.
[2:1]	**CBS**	**Boot Selection of CONFIG (Read Only)** This bit is initiated with the CBS (CONFIG0[7:6]) after any reset is happened except CPU reset (CPU is 1) or system reset (SYS) is happened. The following function is valid when MBS (FMC_ISPSTS[3])= 1. 00 = LDROM with IAP mode. 01 = LDROM without IAP mode. 10 = APROM with IAP mode. 11 = APROM without IAP mode.
[0]	**ISPBUSY**	**ISP Busy Flag (Read Only)** Write 1 to start ISP operation and this bit will be cleared to 0 by hardware automatically when ISP operation is finished. This bit is the mirror of ISPGO(FMC_ISPTRG[0]). 0 = ISP operation is finished. 1 = ISP is progressed.

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	0x0000_0000

31	30	29	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.
[8]	**FADIS**	**Flash Access Cycle Auto-tuning Disable Bit (Write Protect)** Set this bit to disable Flash access cycle auto-tuning function 0 = Flash access cycle auto-tuning Enabled. 1 = Flash access cycle auto-tuning Disabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register. When FMC is doing auto-tuning, we considered as an ISP operation need to monitor busy flag.
[7:3]	**Reserved**	Reserved.
[3:0]	**CYCLE**	**Flash Access Cycle Control (Write Protect)** This register is updated by software.User needs to check the speed of HCLK and set the cycle >0 0001 = CPU access with one wait cycle if cache miss; Flash access cycle is 1;. The HCLK working frequency range range is <= 20 MHz 0010 = CPU access with two wait cycles if cache miss; Flash access cycle is 2;. The optimized HCLK working frequency range is (20 MHz +1)~40 MHz 0011 = CPU access with three wait cycles if cache miss; Flash access cycle is 3;. The optimized HCLK working frequency range is (40 MHz +1)~60 MHz 0100 = CPU access with four wait cycles if cache miss; Flash access cycle is 4;. The optimized HCLK working frequency range is (60 MHz +1)~80 MHz 0101 = CPU access with five wait cycles if cache miss; Flash access cycle is 5;. The optimized HCLK working frequency range is (80 MHz +1)~100 MHz 0110 = CPU access with six wait cycles if cache miss; Flash access cycle is 6;. The optimized HCLK working frequency range is (100 MHz +1)~120 MHz 0111 = CPU access with seven wait cycles if cache miss; Flash access cycle is 7;. The optimized HCLK working frequency range is (120 MHz +1)~192 MHz 1000 = CPU access with eight wait cycles if cache miss; Flash access cycle 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 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 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 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 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-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.
[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.

[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: ● APROM writes to itself if APUEN is set to 0. ● LDROM writes to itself if LDUEN is set to 0. ● CONFIG is erased/programmed if CFGUEN is set to 0. ● Erase or Program command at brown-out detected ● Destination address is illegal, such as over an available range. ● Invalid ISP commands ● The base and size of new XOM regions is wrong, overlap or writed twice ● The input setting of XOM page erase function is wrong ● The active XOM regions is accessed (except for chip erase, page erase, checksum and read CID/DID) ● The XOM setting page is accessed (except for chip erase, word program and read) ● Violate the load code read protection ● Checksum or Flash All One Verification is not executed in their valid range ● Bank erase is not executed in APROM
[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-program Address 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.

XOM Region0 Status Register (FMC_XOMR0STS)

Register	Offset	R/W	Description	Reset Value
FMC_XOMR0STS	FMC_BA+0xD0	R	XOM Region 0 Status Register	0x0FF8_00FF

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

Bits	Description	
[31:8]	**BASE**	**XOM Region 0 Base Address (Page-aligned)** BASE is the base address of XOM Region 0.
[7:0]	**SIZE**	**XOM Region 0 Size (Page-aligned)** SIZE is the page number of XOM Region 0.

XOM Region1 Status Register (FMC_XOMR1STS)

Register	Offset	R/W	Description	Reset Value
FMC_XOMR1STS	FMC_BA+0xD4	R	XOM Region 1 Status Register	0x0FF8_00FF

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

Bits	Description	
[31:8]	**BASE**	**XOM Region 1 Base Address (Page-aligned)** BASE is the base address of XOM Region 1.
[7:0]	**SIZE**	**XOM Region 1 Size (Page-aligned)** SIZE is the page number of XOM Region 1.

XOM Region2 Status Register (FMC_XOMR2STS)

Register	Offset	R/W	Description	Reset Value
FMC_XOMR2STS	FMC_BA+0xD8	R	XOM Region 2 Status Register	0x0FF8_00FF

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

Bits	Description	
[31:8]	**BASE**	**XOM Region 2 Base Address (Page-aligned)** BASE is the base address of XOM Region 2.
[7:0]	**SIZE**	**XOM Region 2 Size (Page-aligned)** SIZE is the page number of XOM Region 2.

XOM Region3 Status Register (FMC_XOMR3STS)

Register	Offset	R/W	Description	Reset Value
FMC_XOMR3STS	FMC_BA+0xDC	R	XOM Region 3 Status Register	0x0FF8_00FF

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

Bits	Description	
[31:8]	BASE	**XOM Region 3 Base Address (Page-aligned)** BASE is the base address of XOM Region 3.
[7:0]	SIZE	**XOM Region 3 Size (Page-aligned)** SIZE is the page number of XOM Region 3.

XOM Status Register (FMC_XOMSTS)

Register	Offset	R/W	Description	Reset Value
FMC_XOMSTS	FMC_BA+0xE0	R	XOM 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			XOMPEF	XOMR3ON	XOMR2ON	XOMR1ON	XOMR0ON

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**XOMPEF**	**XOM Page Erase Function Fail** XOM page erase function status. If XOMPEF is set to 1, user needs to erase XOM region again. 0 = Sucess. 1 = Fail.
[3]	**XOMR3ON**	**XOM Region 3 On** XOM Region 3 active status. 0 = No active. 1 = XOM region 3 is active.
[2]	**XOMR2ON**	**XOM Region 2 On** XOM Region 2 active status. 0 = No active. 1 = XOM region 2 is active.
[1]	**XOMR1ON**	**XOM Region 1 On** XOM Region 1 active status. 0 = No active. 1 = XOM region 1 is active.
[0]	**XOMR0ON**	**XOM Region 0 On** XOM Region 0 active status. 0 = No active. 1 = XOM region 0 is active.

6.5 Data Flash Memory Controller (DFMC)

6.5.1 Overview

The DFMC is equipped with 32 Kbytes on-chip Data Flash to store some application dependent data.

6.5.2 Features

- Supports 32 Kbytes application Data Flash
- Supports 256 bytes page erase for all embedded Flash
- Supports CRC32 checksum calculation function
- Supports Data Flash all one verification function
- Supports In-System-Programming (ISP) to update embedded Data Flash memory

FMC Features	M471
32 KB Data ROM	●
256 B page erase	●
Mass erase for Data Flash.	●
CRC32 checksum calculation function	●
hardware Writer mode and In-Circuit-Programming (ICP)	●
smart entry detection on Writer mode and ICP	●

Table 6.5-1 FMC Features Comparison Table at Different Chip

6.5.3 Block Diagram

The Data Flash memory controller (DFMC) consists of AHB slave interface, Data Flash control registers, Data Flash initialization controller, Data Flash operation control. The block diagram of Data Flash memory controller is shown as follows.

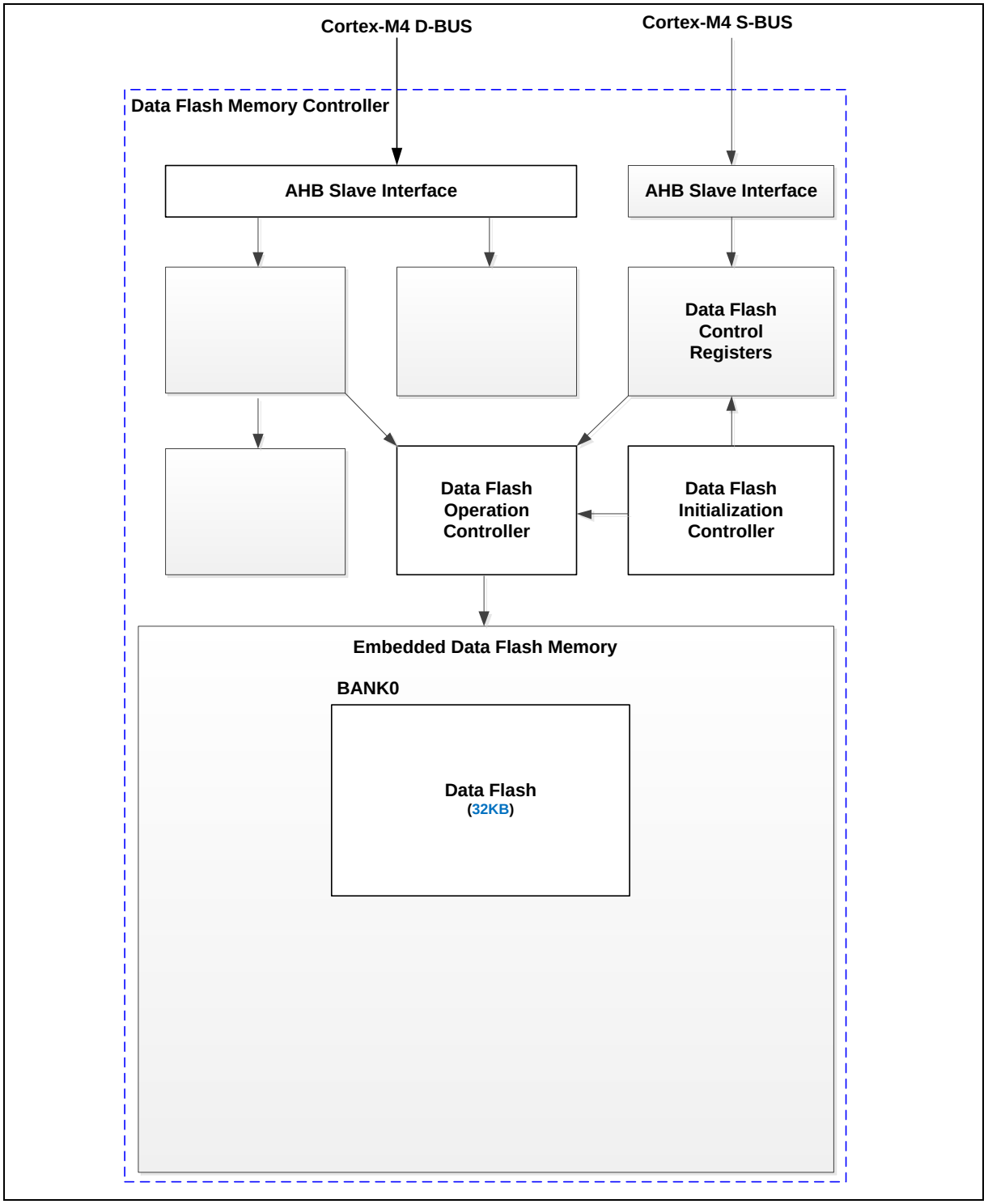


Figure 6.5-1 Data Flash Memory Controller Block Diagram

AHB Slave Interface

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

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

Data Flash Initialization Controller

When chip is powered on or active from reset, the Data Flash initialization controller will start to access Data Flash automatically and check the Data Flash stability.

Data Flash Operation Controller

The Data Flash operations, such as Data Flash erase, Data Flash program, and Data Flash read operation, have specific control timing for embedded Data Flash memory. The Data Flash operation controller generates those control timing by requested, the Data Flash control registers and the Data Flash initialization controller.

Embedded Data Flash Memory

The embedded Data Flash memory is the data memory for user access data. It is consists of the 32 Kbytes Data Flash. The page erase Data Flash size is 256 Bytes, and minimum program bit size is 32 bits.

6.5.4 Functional Description

DFMC functions include the memory organization, ISP, the embedded Data Flash programming, and checksum calculation. The Data Flash memory map and system memory map are also introduced in the memory organization.

6.5.4.1 Memory Organization

The DFMC memory consists of the embedded Data Flash memory. The embedded Data Flash memory is programmable. The address map includes Data Flash memory map.

BANK	Data Flash Memory Block	Address Range
0	Data Flash with 32 Kbytes	0x40_0000 ~ 0x40_7fff

Table 6.5-2 Block Address Range

6.5.4.2 Data Flash Memory Map

The Data Flash memory map is different from system memory map. The system memory map is used by CPU fetch data from DFMC memory. The Data Flash memory map is used for ISP function to read, program or erase DFMC memory. The Data Flash memory map is as Figure 6.5-2.

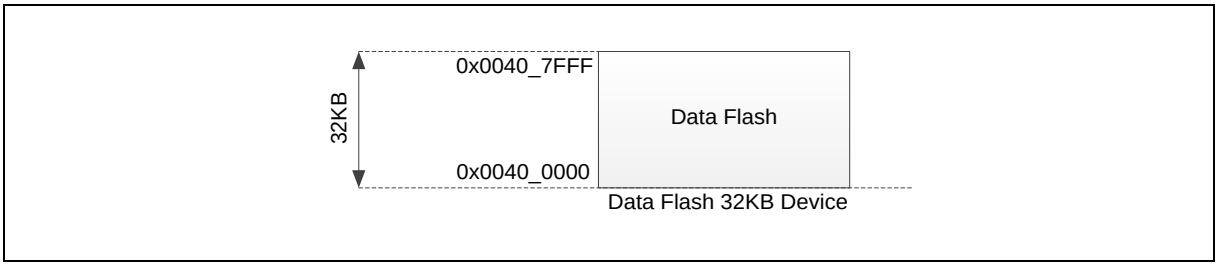


Figure 6.5-2 Data Flash Memory Map

6.5.4.3 In-System-Programming (ISP)

The M471V/M471K/M471C supports In-System-Programming (ISP) function allowing the embedded Data Flash memory to be reprogrammed under software control. ISP is performed without removing the microcontroller from the system through the firmware and on-chip connectivity interface, such as UART, USB, I^2C, and SPI(depended on chip feature). The target Data Flash memory space that ISP function operates cannot be across banks. Table 6.5-3 lists all ISP commands.

The ISP provides the following functions for embedded Data Flash memory.

- Supports Data Flash page erase function
- Supports Data Flash data program function
- Supports Data Flash data read function
- Supports memory CRC32 checksum calculation function
- Supports Data Flash all one verification function

ISP Commands

ISP Command	DFMC_ISPCMD	DFMC_ISPADDR	DFMC_ISPDAT DFMC_MPDAT0~DFMC_MPDAT3
Data Flash Page Erase	0x22	Valid address of Data Flash memory organization. It must be page (4 Kbytes) alignment. Note that DFMC_ISPADDR[11:0] will be ignored.	N/A

Data Flash Mass Erase	0x26	0x0040_0000	N/A
Data Flash 32-bit Program	0x21	Valid address of Flash memory organization It must be 4 bytes word alignment	DFMC_ISPDAT :Programming Data
Data Flash Read	0x00	Valid address of Flash memory organization It must be 4 bytes word alignment	DFMC_ISPDAT: Return Data
Read Company ID	0x0B	0x0000_0000	DFMC_ISPDAT: 0x0000_00DA
Read Device ID	0x0C	0x0000_0000	DFMC_ISPDAT: Return Device ID
Read CRC32 Checksum	0x0D	0x0000_0000	DFMC_ISPDAT: Return Checksum
Run CRC32 Checksum Calculation	0x2D	Valid start address of memory organization It must be 256 bytes page alignment	DFMC_ISPDAT: Size It must be 256 bytes alignment
Read Data Flash All One Result	0x08	Keep address of “Run Data Flash All One Verification”	DFMC_ISPDAT: Return Result 0xA110_0000 : Data Flash is not all one 0xA11F_FFFF: Data Flash is all one.
Run Data Flash All One Verification	0x28	Valid start address of memory organization It must be 256 bytes page alignment	DFMC_ISPDAT: Size It must be 256 bytes alignment

Table 6.5-3 ISP Command List

ISP Procedure

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

After unlocking the protected register bits, user needs to set the DFMC_ISPCTL control register to decide to update Data Flash, and then set ISPEN (DFMC_ISPCTL[0]) to enable ISP function.

Once the DFMC_ISPCTL register is set properly, user can set DFMC_ISPCMD (refer to Table 6.5-3 ISP command list) for specify operation. Set DFMC_ISPADDR for target Data Flash memory based on Data Flash memory organization. DFMC_ISPDAT can be used to set the data to program or used to return the read data according to DFMC_ISPCMD. The ISP procedure flow is shown in Figure 6.5-3 ISP Procedure Example

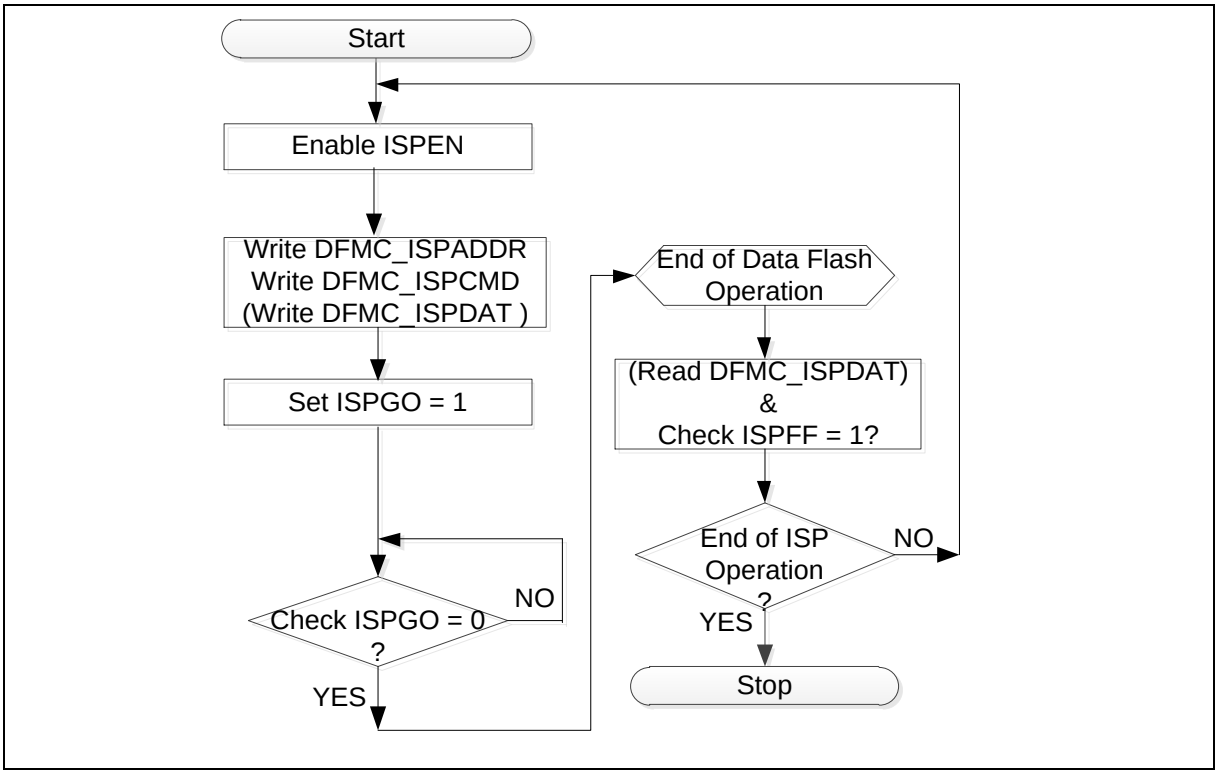


Figure 6.5-3 ISP Procedure Example

Finally, set the ISPGO (DFMC_ISPTRG[0]) register to perform the relative ISP function. When ISP function is active, the ISPBUSY(DFMC_ISPSTS[0]) be set to 1. The ISPGO(DFMC_ISPTRG[0]) and ISPBUSY(DFMC_ISPSTS[0]) are self-cleared when ISP function has been done.

Several error conditions will be checked after ISP is completed. If an error condition occurs, ISP operation is not started and the ISP fail flag will be set instead. ISPFF(DFMC_ISPSTS[6]) flag can only be cleared by software. The next ISP procedure can be started even ISPFF(DFMC_ISPSTS[6]) bit is kept as 1. Therefore, it is recommended to check the ISPFF(DFMC_ISPSTS[6]) bit and clear it after each ISP operation if it is set to 1.

When the ISPGO(DFMC_ISPTRG[0]) bit is set and then CPU access the same bank, 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. User can check whether ISP operation is finished or not by the ISPGO(DFMC_ISPTRG[0]) bit.

When CPU access operation and ISP command are executed in different bank, CPU and ISP command can operate in parallel.

Note: CPU access operation only can read word alignment data in Data Flash memory.

6.5.4.4 Embedded Data Flash Memory Programming

The M471V/M471K/M471C provides 32-bit Data Flash memory programming function to speed up Data Flash updated procedure. Table 6.5-4 lists required DFMC control registers in each embedded Data Flash programming function.

Register	Description	32-Bit Programming
DFMC_ISPCTL	ISP Control Register	√

DFMC_ISPADDR	ISP Address Register	√
DFMC_ISPDAT	ISP Data Register	√
DFMC_ISPCMD	ISP Command Register	0x21
DFMC_ISPTRG	ISP Trigger Register	√
DFMC_ISPSTS	ISP Status Register	√

Table 6.5-4 DFMC Control Registers for Data Flash Programming

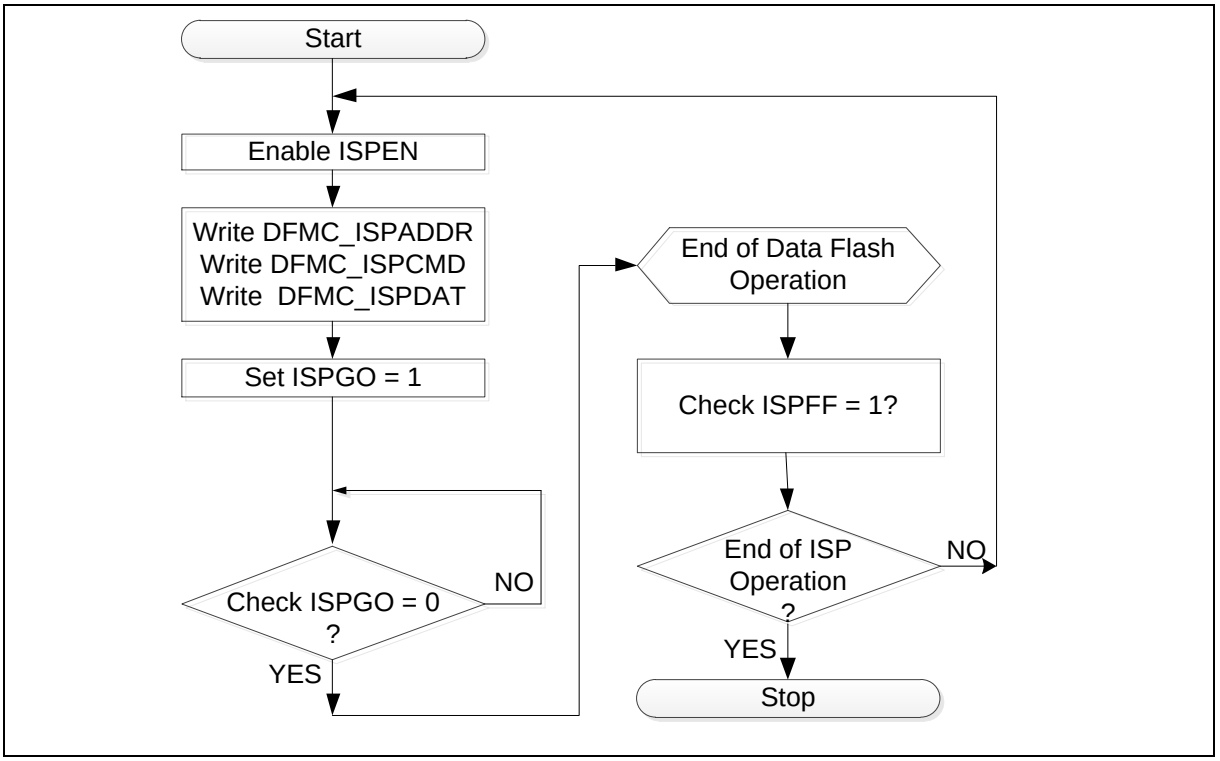


Figure 6.5-4 ISP 32-bit Programming Procedure

6.5.4.5 CRC32 Checksum Calculation

The M471V/M471K/M471C supports the CRC32 checksum calculation function to help user quickly check the Data Flash memory content. 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 6.5-5.

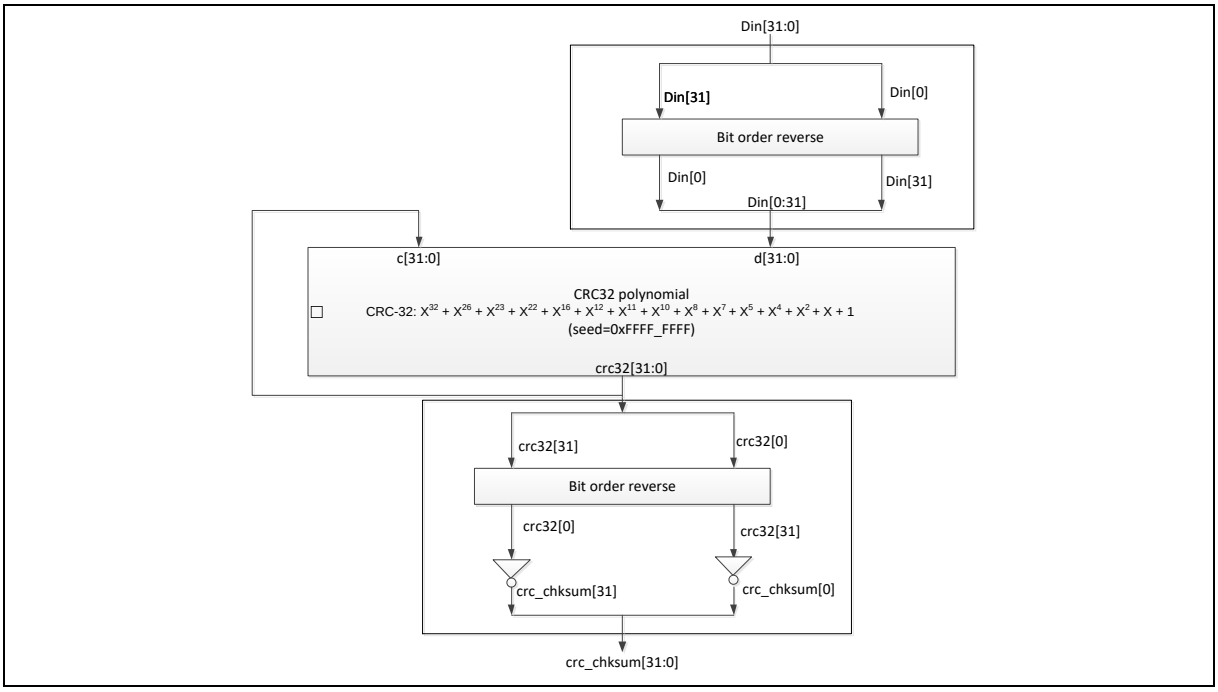


Figure 6.5-5 Data Flash CRC32 Checksum Calculation

Three steps complete this CRC32 checksum calculation.

1. Perform ISP “Run Memory CRC32 Checksum” operation
2. Perform ISP “Read Memory CRC32 Checksum” operation
3. Read DFMC_ISPDAT to get checksum.

In Step 1, user has to set the memory starting address (DFMC_ISPADDR) and size (DFMC_ISPDAT) to calculate. Both address and size have to be 256 bytes alignment, the size should be ≥ 256 bytes and the starting address includes Data Flash.

In Step 2, the DFMC_ISPADDR should be kept as the same as Step 1.

In Step 3, the checksum is read from DFMC_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

6.5.4.6 Data Flash All One Verification

The M471V/M471K/M471C supports the Data Flash all one verification function to help user quickly check a memory block content blanking for Data Flash after Data Flash erase operation.

Two or Three steps complete this Data Flash all one verification.

Two-step flow:

1. Perform ISP “Run Data Flash All One Verification” operation
2. Read ALLONE(DFMC_ISPSTA[7])bit to get the verification result

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

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

Three-step flow:

3. Perform ISP “Run Data Flash All One Verification” operation

4. Perform ISP “Read Data Flash All One Result” operation
5. Read DFMC_ISPDAT to get the verification result.

DFMC_ISPDATA : 0xA11F_FFFF, all of Data Flash bits are 1 in verification block memory.

DFMC_ISPDATA : 0xA110_0000, Data Flash bits are not all 1 in verification block memory.

In Step 1, user has to set the memory starting address (DFMC_ISPADDR) and size (DFMC_ISPDAT) to verify. Both address and size have to be 256 bytes alignment, the size should be ≥ 256 bytes and the starting address includes Data Flash.

In Step 2, the DFMC_ISPADDR should be kept as the same as Step 1.

6.5.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
DFMC Base Address **DFMC_BA = 0x4000_F000**				
DFMC_ISPCTL	DFMC_BA+0x00	R/W	ISP Control Register	0x0000_0000
DFMC_ISPADDR	DFMC_BA+0x04	R/W	ISP Address Register	0x0000_0000
DFMC_ISPDAT	DFMC_BA+0x08	R/W	ISP Data Register	0x0000_0000
DFMC_ISPCMD	DFMC_BA+0x0C	R/W	ISP Command Register	0x0000_0000
DFMC_ISPTRG	DFMC_BA+0x10	R/W	ISP Trigger Control Register	0x0000_0000
DFMC_ISPSTS	DFMC_BA+0x40	R/W	ISP Status Register	0x0000_0000
DFMC_CYCCTL	DFMC_BA+0x4C	R/W	Data Flash Access Cycle Control Register	0x0000_0001

6.5.6 Register Description

ISP Control Register (DFMC_ISPCTL)

Register	Offset	R/W	Description	Reset Value
DFMC_ISPCTL	DFMC_BA+0x00	R/W	ISP Control Register	0x0000_0000

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

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**ISPIFEN**	**ISP Interrupt Enable bit (Write Protect)** 0 = ISP Interrupt Disabled. 1 = ISP Interrupt Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[23:7]	**Reserved**	Reserved.
[6]	**ISPFF**	**ISP Fail Flag (Write Protect)** This bit is set by hardware when a triggered ISP meets any of the following conditions: This bit needs to be cleared by writing 1 to it. ● Data Flash writes to itself if DATAEN is set to 0. ● Erase or Program command at brown-out detected ● Destination address is illegal, such as over an available range. ● Invalid ISP commands ● Violate the load code read protection ● Checksum or Flash All One Verification is not executed in their valid range ● Mass erase is not executed in Data Flash **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[5:4]	**Reserved**	Reserved.
[3]	**DATAEN**	**Data Flash Update Enable Bit (Write Protect)** 0 = Data Flash cannot be updated. 1 = Data Flash can be updated. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[2:1]	**Reserved**	Reserved.

[0]	**ISPEN**	**ISP Enable Bit (Write Protect)** ISP function enable bit. Set this bit to enable ISP function. 0 = ISP function Disabled. 1 = ISP function Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

ISP Address (DFMC_ISPADDR)

Register	Offset	R/W	Description	Reset Value
DFMC_ISPADDR	DFMC_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 M471V/M471K/M471C is equipped with embedded Data Flash. ISPADDR[1:0] must be kept 00 for ISP 32-bit operation. For CRC32 Checksum Calculation command, this field is the Data Flash starting address for checksum calculation, 256 bytes alignment is necessary for CRC32 checksum calculation. For Data Flash32-bit Program, ISP address needs word alignment (4-byte).

ISP Data Register (DFMC_ISPDAT)

Register	Offset	R/W	Description	Reset Value
DFMC_ISPDAT	DFMC_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 (DFMC_ISPCTL[6]) is 1, ISPDAT = 0xffff_ffff. For Run CRC32 Checksum Calculation command, ISPDAT is the memory size (byte) and 256 bytes 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 (DFMC_ISPCMD)

Register	Offset	R/W	Description	Reset Value
DFMC_ISPCMD	DFMC_BA+0x0C	R/W	ISP Command Register	0x0000_0000

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

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6:0]	**CMD**	**ISP Command** ISP command table is shown below: 0x00= Data FLASH Read. 0x08= Read Data Flash All-One Result. 0x0B= Read Company ID. 0x0C= Read Device ID. 0x0D= Read Checksum. 0x21= Data FLASH 32-bit Program. 0x22= Data FLASH Page Erase. Erase any page in Data Flash. 0x26= Data FLASH Mass Erase. Erase all pages in Data Flash. 0x28= Run Data Flash All-One Verification. 0x2D= Run Checksum Calculation. The other commands are invalid.

ISP Trigger Control Register (DFMC_ISPTRG)

Register	Offset	R/W	Description	Reset Value
DFMC_ISPTRG	DFMC_BA+0x10	R/W	ISP Trigger Control Register	0x0000_0000

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

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

ISP Status Register (DFMC_ISPSTS)

Register	Offset	R/W	Description	Reset Value
DFMC_ISPSTS	DFMC_BA+0x40	R/W	ISP Status Register	0x0000_0000

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

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**ISPIF**	**ISP Interrupt Flag** 0 = ISP command not finish or ISP fail flag is 0. 1 = ISP command finish or ISP fail is 1. **Note:** Write 1 to clear this bit.
[23:8]	**Reserved**	Reserved.
[7]	**ALLONE**	**Data 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 Data Flash All-One Verification” complete; this bit also can be clear by writing 1 0 = Data Flash bits are not all 1 after “Run Data Flash All-One Verification” is complete. 1 = All of Data Flash bits are 1 after “Run Data Flash All-One Verification” is complete.
[6]	**ISPFF**	**ISP Fail Flag (Write Protect)** This bit is the mirror of ISPFF (DFMC_ISPCTL[6]), it needs to be cleared by writing 1 to DFMC_ISPCTL[6] or DFMC_ISPSTS[6]. This bit is set by hardware when a triggered ISP meets any of the following conditions: ● Data Flash writes to itself if DATAEN is set to 0. ● Erase or Program command at brown-out detected ● Destination address is illegal, such as over an available range. ● Invalid ISP commands ● Violate the load code read protection ● Checksum or Flash All One Verification is not executed in their valid range ● Mass erase is not executed in Data Flash **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[5:1]	**Reserved**	Reserved.

[0]	**ISPBUSY**	**ISP Busy Flag (Read Only)** Write 1 to start ISP operation and this bit will be cleared to 0 by hardware automatically when ISP operation is finished. This bit is the mirror of ISPGO(DFMC_ISPTRG[0]). 0 = ISP operation is finished. 1 = ISP is progressed.

Data Flash Access Cyce Control Register (DFMC_CYCCTL)

Register	Offset	R/W	Description	Reset Value
DFMC_CYCCTL	DFMC_BA+0x4C	R/W	Data Flash Access Cycle 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				**CYCLE**			

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3:0]	**CYCLE**	**Data Flash Access Cycle Control (Write Protect)** This register is updated by software. 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 cycle is 1;. The HCLK working frequency range range is<(27 MHz-1) 0010 = CPU access with two wait cycles if cache miss; Flash access cycle is 2;. The optimized HCLK working frequency range is 27~(54 MHz-1) 0011 = CPU access with three wait cycles if cache miss; Flash access cycle is 3;. The optimized HCLK working frequency range is 54~(81 MHz-1) 0100 = CPU access with four wait cycles if cache miss; Flash access cycle is 4;. The optimized HCLK working frequency range is81~(108 MHz-1) 0101 = CPU access with five wait cycles if cache miss; Flash access cycle is 5;. The optimized HCLK working frequency range is 108~(135 MHz-1) 0110 = CPU access with six wait cycles if cache miss; Flash access cycle is 6;. The optimized HCLK working frequency range is 135~(162 MHz-1) 0111 = CPU access with seven wait cycles if cache miss; Flash access cycle is 7;. The optimized HCLK working frequency range is 162~(192 MHz-1) 1000 = CPU access with eight wait cycles if cache miss; Flash access cycle is 8;. The optimized HCLK working frequency range is >192 MHz **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

6.6 General Purpose I/O (GPIO)

6.6.1 Overview

This chip has up to 119 General Purpose I/O pins to be shared with other function pins depending on the chip configuration. These 119 pins are arranged in 9 ports named as PA, PB, PC, PD, PE, PF, PG, PH and PI. PA, PB, PD, PE and PF has 16 pins on port. PC has 15 pins on port. PG has 10 pins on port. PH has 8 pins on port. PI has 6 pins on port. Each of the 119 pins is independent and has the corresponding register bits to control the pin mode function and data.

The I/O type of each of I/O pins can be configured by software individually as Input, Push-pull output, Open-drain output or Quasi-bidirectional mode. After the chip is reset, the I/O mode of all pins are depending on CIOINI (CONFIG0[10]).

6.6.2 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 selectable
- I/O pin can be configured as interrupt source with edge/level setting
- Supports High Drive and High Slew Rate I/O mode
- Configurable default I/O mode of all pins after reset by CIOINI (CONFIG0[10]) setting
  - CIOINI = 0, all GPIO pins in Quasi-bidirectional mode after chip reset
  - CIOINI = 1, all GPIO pins in input mode after chip reset
- Support independent pull-up control
- Enabling the pin interrupt function will also enable the wake-up function

6.6.3 Block Diagram

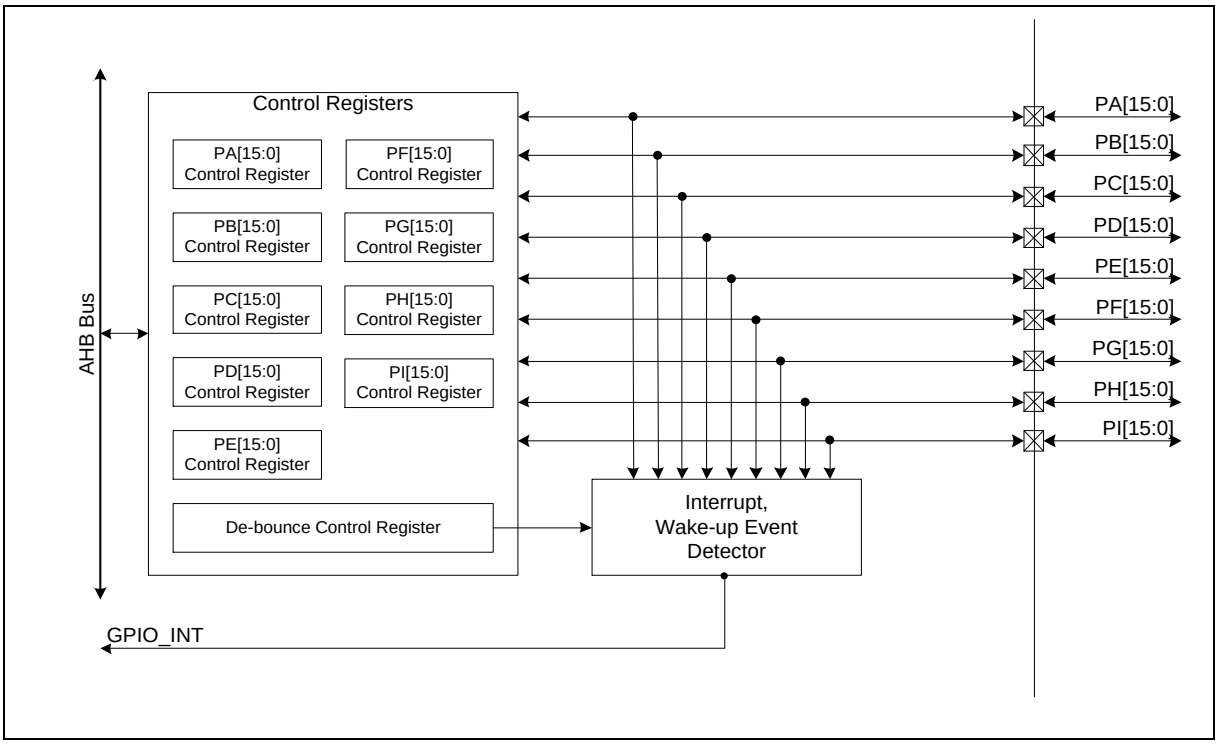


Figure 6.6-1 GPIO Controller Block Diagram

Note: The PC.15/PG.0/PG.1/PG.5~PG.8/PH.0~PH.3/PH.12~PH.15/PI.6~PI.15 pins are not available.

6.6.4 Basic Configuration

- Reset configuration
  - Reset GPIO in GPIORST SYS_IPRST1[1]
- Pin configuration

Group	Pin Name	GPIO	MFP
INT0	INT0	PA.6, PB.5	MFP15
INT1	INT1	PA.7, PB.4, PD.15	MFP15
INT2	INT2	PB.3, PC.6	MFP15
INT3	INT3	PB.2, PC.7	MFP15
INT4	INT4	PB.6	MFP13
		PA.8, PF.15	MFP15
INT5	INT5	PB.7	MFP13
		PD.12, PF14	MFP15
INT6	INT6	PB.8	MFP13
		PD.11	MFP15
INT7	INT7	PB.9	MFP13
		PD.10	MFP15

6.6.5 Functional Description

6.6.5.1 Input Mode

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

6.6.5.2 Push-pull Output Mode

Figure 6.6-2 shows the diagram of Push-pull Output Mode. Set MODEn (Px_MODE[2n+1:2n]) to 01 as Px.n pin is in Push-pull Output mode and the I/O pin supports digital output function with source/sink current capability. The bit value in the corresponding DOUT (Px_DOUT[n]) is driven on the pin.

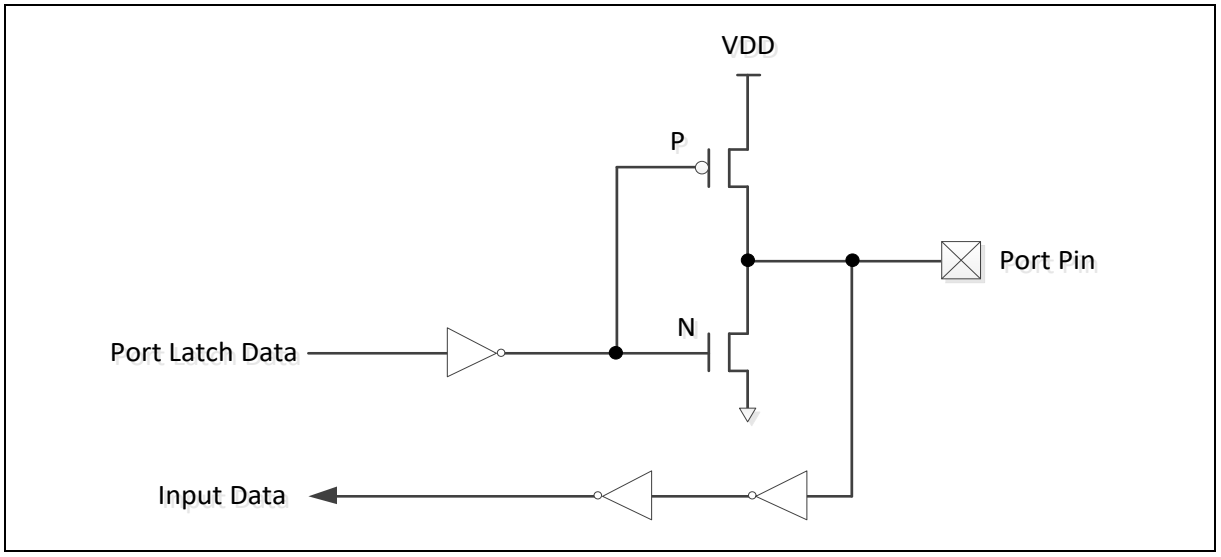


Figure 6.6-2 Push-Pull Output

6.6.5.3 Open-drain Mode

Figure 6.6-3 shows the diagram of Open-drain Mode. Set MODEn (Px_MODE[2n+1:2n]) to 10 as the Px.n pin is in Open-drain mode and the digital output function of I/O pin supports only sink current capability, an external pull-up register is needed for driving high state. If the bit value in the corresponding DOUT (Px_DOUT[n]) bit is 0, the pin drive a low output on the pin. If the bit value in the corresponding DOUT (Px_DOUT[n]) bit is 1, the pin output drives high that is controlled by external pull high resistor.

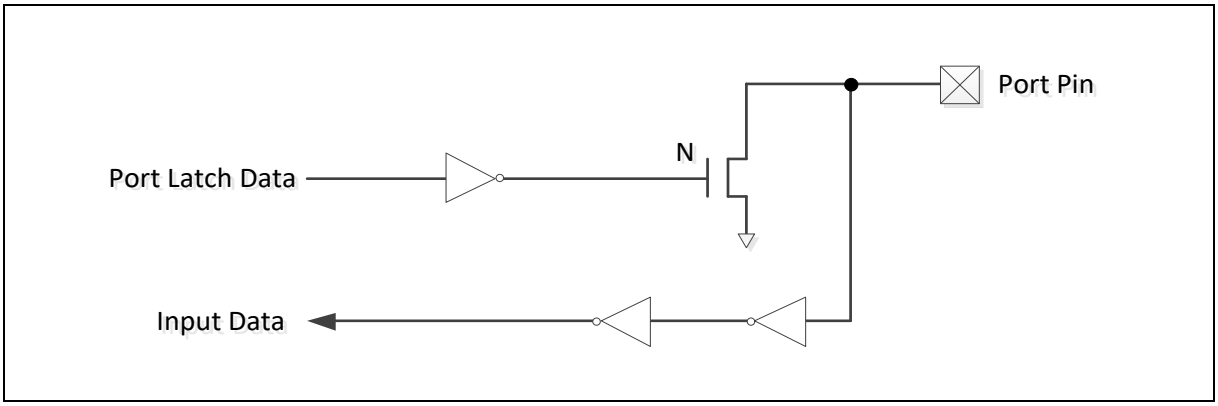


Figure 6.6-3 Open-Drain Output

6.6.5.4 Quasi-bidirectional Mode

Figure 6.6-4 shows the diagram of Quasi-bidirectional Mode. Set MODEn (Px_MODE[2n+1:2n]) to 11 as the Px.n pin is in Quasi-bidirectional mode and the I/O pin supports digital output and input function at the same time but the source current is only up to hundreds uA. Before the digital input function is performed the corresponding DOUT (Px_DOUT[n]) bit must be set to 1. The quasi-bidirectional output is common on the 80C51 and most of its derivatives. If the bit value in the corresponding DOUT (Px_DOUT[n]) bit is 0, the pin drive a low output on the pin. If the bit value in the corresponding DOUT (Px_DOUT[n]) bit is 1, the pin will check the pin value. If pin value is high, no action takes. If pin state is low, the pin will drive strong high with 2 clock cycles on the pin and then disable the strong output drive. Meanwhile, the pin status is controlled by internal pull-up resistor. Note that the source current capability in quasi-bidirectional mode is only about 200 uA to 30 uA for V_{DD} is form 5.0 V to 2.5 V.

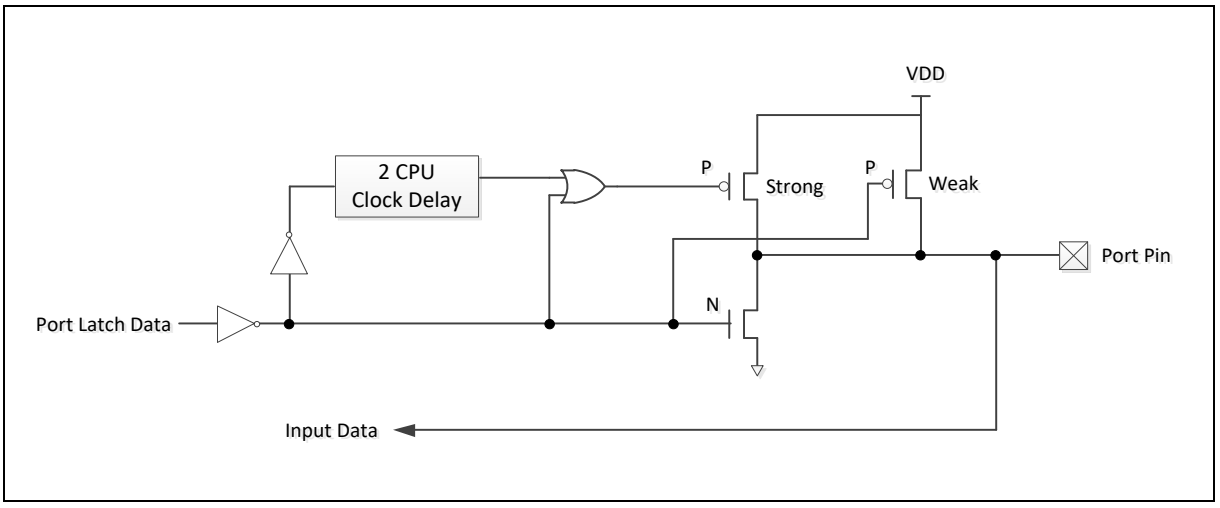


Figure 6.6-4 Quasi-Bidirectional I/O Mode

6.6.5.5 GPIO Interrupt and Wake-up Function

Each GPIO pin can be set as chip interrupt source by setting correlative RHIEN (Px_INTEN[n+16])/ FLIEN (Px_INTEN[n]) bit and TYPE (Px_INTTYPE[n]). 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 can also be the chip wake-up source when chip enters Idle/Power-down mode. The setting of wake-up trigger condition is the same as GPIO interrupt trigger.

6.6.5.6 GPIO De-bounce Function

GPIO de-bounce function can be used to sample interrupt input for GPIO pin and prevent unexpected interrupt happened which caused by noise. GPIO de-bounce function only support edge detection trigger type and are not supported in Power-down mode. For edge trigger condition, there are three types of interrupt condition can be selected for de-bounce function: falling edge trigger, rising edge trigger and both rising and falling edge trigger by setting correlative RHIEN (Px_INTEN[n+16])/ FLIEN (Px_INTEN[n]) bit and TYPE (Px_INTTYPE[n]). If user wants to use de-bounce function, de-bounce enable control register Px_DBEN must be set for corresponding GPIO pin. The de-bounce clock source can be HCLK or LIRC by setting DBCLKSRC (GPIO_DBCTL[4]) register. And DBCLKSEL (GPIO_DBCTL[3:0]) register can control sampling cycle period.

Figure 6.6-5 shows GPIO rising edge trigger interrupt. The interval of time between the two valid sample signal is determined by DBCLKSRC (GPIO_DBCTL[4]) and DBCLKSEL (GPIO_DBCTL[3:0]). Each valid data from GPIO pin need to be sample twice. For rising edge setting, if pin status is low before setting DBEN (Px_DBEN[n]), interrupt will happen when generating a pin high valid data. But, if pin status is high before setting DBEN (Px_DBEN[n]), interrupt will happen when generating a pin low valid data first, and then generating a pin high valid data. For falling edge trigger, Figure 6.6-6 shows the situation is opposite to rising edge trigger.

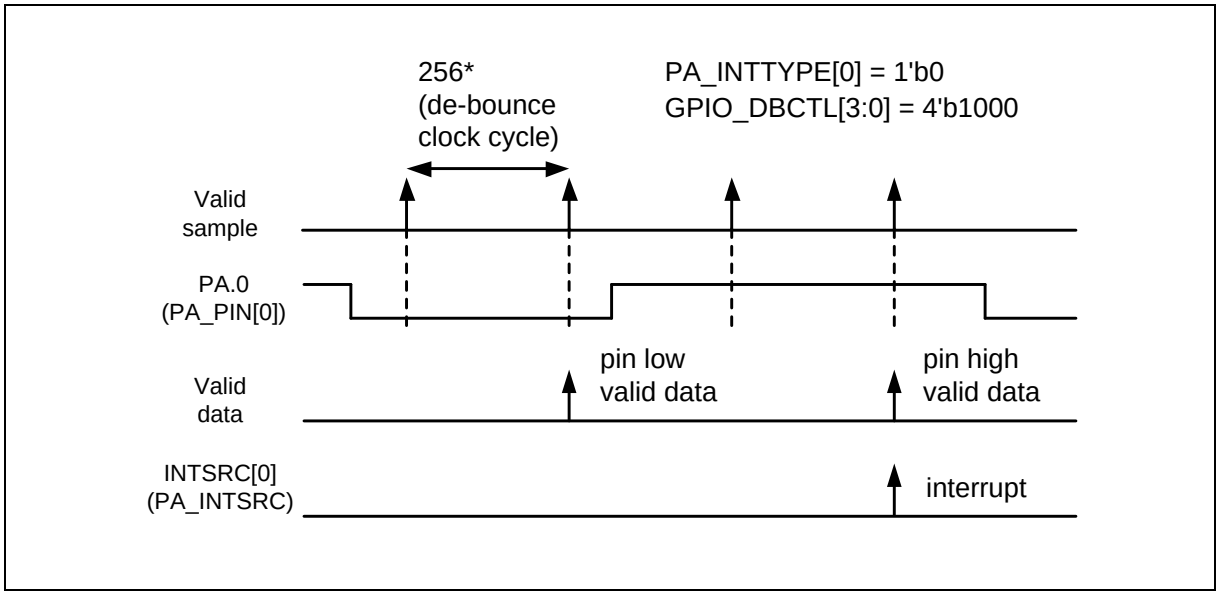


Figure 6.6-5 GPIO Rising Edge Trigger Interrupt

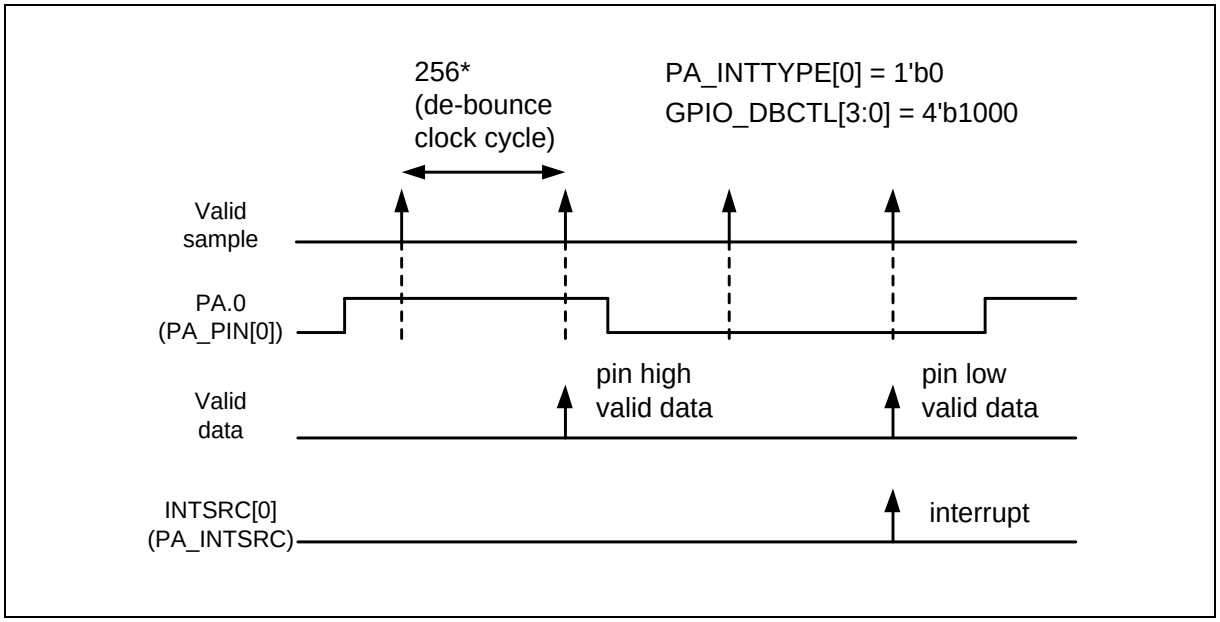


Figure 6.6-6 GPIO Falling Edge Trigger Interrupt

6.6.5.7 INT Edge Detect and Trigger Function

There are many INT0~7 GPIO pins and can be selected by multi-function. Each INT pin will be filtered by a 3-bit noise filter. User can enable the noise filter function by NFEN (INTn_INNF[0], n=0~7) bit, and noise filter sampling clock can be selected by setting NFSEL (INTn_INNF[6:4], n=0~7) bits to fit different noise properties. Moreover, by setting the NFCNT (INTn_INNF[10:8], n=0~7) bits, user can define how many sampling clock cycles a filter will recognize the effective edge of the INT pin signal.

Each INT pin can be detected by setting EDETCTLn (INT_EDETCTL[2n+1:2n]) and EDIFn (INT_EDSTS[n], n=0~7) will be set if INT pin matches a condition. Interrupt will happen if user enables

EDIENn (INT_EDINTEN[n], n=0~7) and INT pin matches the condition.

In addition to detect function, it can also trigger EPWM if the INT pin matches the condition. More detailed setting is in EPWM “External Pin Event Trigger” section.

6.6.6 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_FFFF
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_XXXX
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 Selection Register	0x0000_0000
PB_MODE	GPIO_BA+0x040	R/W	PB I/O Mode Control	0xXXXX_XXXX
PB_DINOFF	GPIO_BA+0x044	R/W	PB Digital Input Path Disable Control	0x0000_0000
PB_DOUT	GPIO_BA+0x048	R/W	PB Data Output Value	0x0000_FFFF
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_XXXX
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 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_7FFF

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
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_XXXX
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 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_XXXX
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 Selection Register	0x0000_0000
PE_MODE	GPIO_BA+0x100	R/W	PE I/O Mode Control	0xXXXX_XXXX
PE_DINOFF	GPIO_BA+0x104	R/W	PE Digital Input Path Disable Control	0x0000_0000
PE_DOUT	GPIO_BA+0x108	R/W	PE Data Output Value	0x0000_FFFF
PE_DATMSK	GPIO_BA+0x10C	R/W	PE Data Output Write Mask	0x0000_0000
PE_PIN	GPIO_BA+0x110	R	PE Pin Value	0x0000_XXXX
PE_DBEN	GPIO_BA+0x114	R/W	PE De-Bounce Enable Control Register	0x0000_0000
PE_INTTYPE	GPIO_BA+0x118	R/W	PE Interrupt Trigger Type Control	0x0000_0000
PE_INTEN	GPIO_BA+0x11C	R/W	PE Interrupt Enable Control Register	0x0000_0000
PE_INTSRC	GPIO_BA+0x120	R/W	PE Interrupt Source Flag	0x0000_XXXX
PE_SMTEN	GPIO_BA+0x124	R/W	PE Input Schmitt Trigger Enable Register	0x0000_0000

PE_SLEWCTL	GPIO_BA+0x128	R/W	PE High Slew Rate Control Register	0x0000_0000
PE_PUSEL	GPIO_BA+0x130	R/W	PE Pull-up Selection Register	0x0000_0000
PF_MODE	GPIO_BA+0x140	R/W	PF I/O Mode Control	0xXXXX_XXXX
PF_DINOFF	GPIO_BA+0x144	R/W	PF Digital Input Path Disable Control	0x0000_0000
PF_DOUT	GPIO_BA+0x148	R/W	PF Data Output Value	0x0000_FFFF
PF_DATMSK	GPIO_BA+0x14C	R/W	PF Data Output Write Mask	0x0000_0000
PF_PIN	GPIO_BA+0x150	R	PF Pin Value	0x0000_XXXX
PF_DBEN	GPIO_BA+0x154	R/W	PF De-Bounce Enable Control Register	0x0000_0000
PF_INTTYPE	GPIO_BA+0x158	R/W	PF Interrupt Trigger Type Control	0x0000_0000
PF_INTEN	GPIO_BA+0x15C	R/W	PF Interrupt Enable Control Register	0x0000_0000
PF_INTSRC	GPIO_BA+0x160	R/W	PF Interrupt Source Flag	0x0000_XXXX
PF_SMTEN	GPIO_BA+0x164	R/W	PF Input Schmitt Trigger Enable Register	0x0000_0000
PF_SLEWCTL	GPIO_BA+0x168	R/W	PF High Slew Rate Control Register	0x0000_0000
PF_PUSEL	GPIO_BA+0x170	R/W	PF Pull-up Selection Register	0x0000_0000
PG_MODE	GPIO_BA+0x180	R/W	PG I/O Mode Control	0xXXXX_XXXX
PG_DINOFF	GPIO_BA+0x184	R/W	PG Digital Input Path Disable Control	0x0000_0000
PG_DOUT	GPIO_BA+0x188	R/W	PG Data Output Value	0x0000_FE1C
PG_DATMSK	GPIO_BA+0x18C	R/W	PG Data Output Write Mask	0x0000_0000
PG_PIN	GPIO_BA+0x190	R	PG Pin Value	0x0000_XXXX
PG_DBEN	GPIO_BA+0x194	R/W	PG De-Bounce Enable Control Register	0x0000_0000
PG_INTTYPE	GPIO_BA+0x198	R/W	PG Interrupt Trigger Type Control	0x0000_0000
PG_INTEN	GPIO_BA+0x19C	R/W	PG Interrupt Enable Control Register	0x0000_0000
PG_INTSRC	GPIO_BA+0x1A0	R/W	PG Interrupt Source Flag	0x0000_XXXX
PG_SMTEN	GPIO_BA+0x1A4	R/W	PG Input Schmitt Trigger Enable Register	0x0000_0000
PG_SLEWCTL	GPIO_BA+0x1A8	R/W	PG High Slew Rate Control Register	0x0000_0000
PG_PUSEL	GPIO_BA+0x1B0	R/W	PG Pull-up Selection Register	0x0000_0000
PH_MODE	GPIO_BA+0x1C0	R/W	PH I/O Mode Control	0xXXXX_XXXX
PH_DINOFF	GPIO_BA+0x1C4	R/W	PH Digital Input Path Disable Control	0x0000_0000
PH_DOUT	GPIO_BA+0x1C8	R/W	PH Data Output Value	0x0000_0FF0
PH_DATMSK	GPIO_BA+0x1CC	R/W	PH Data Output Write Mask	0x0000_0000
PH_PIN	GPIO_BA+0x1D0	R	PH Pin Value	0x0000_XXXX

PH_DBEN	GPIO_BA+0x1D4	R/W	PH De-Bounce Enable Control Register	0x0000_0000
PH_INTTYPE	GPIO_BA+0x1D8	R/W	PH Interrupt Trigger Type Control	0x0000_0000
PH_INTEN	GPIO_BA+0x1DC	R/W	PH Interrupt Enable Control Register	0x0000_0000
PH_INTSRC	GPIO_BA+0x1E0	R/W	PH Interrupt Source Flag	0x0000_XXXX
PH_SMTEN	GPIO_BA+0x1E4	R/W	PH Input Schmitt Trigger Enable Register	0x0000_0000
PH_SLEWCTL	GPIO_BA+0x1E8	R/W	PH High Slew Rate Control Register	0x0000_0000
PH_PUSEL	GPIO_BA+0x1F0	R/W	PH Pull-up Selection Register	0x0000_0000
PI_MODE	GPIO_BA+0x200	R/W	PI I/O Mode Control	0xXXXX_XXXX
PI_DINOFF	GPIO_BA+0x204	R/W	PI Digital Input Path Disable Control	0x0000_0000
PI_DOUT	GPIO_BA+0x208	R/W	PI Data Output Value	0x0000_003F
PI_DATMSK	GPIO_BA+0x20C	R/W	PI Data Output Write Mask	0x0000_0000
PI_PIN	GPIO_BA+0x210	R	PI Pin Value	0x0000_XXXX
PI_DBEN	GPIO_BA+0x214	R/W	PI De-Bounce Enable Control Register	0x0000_0000
PI_INTTYPE	GPIO_BA+0x218	R/W	PI Interrupt Trigger Type Control	0x0000_0000
PI_INTEN	GPIO_BA+0x21C	R/W	PI Interrupt Enable Control Register	0x0000_0000
PI_INTSRC	GPIO_BA+0x220	R/W	PI Interrupt Source Flag	0x0000_XXXX
PI_SMTEN	GPIO_BA+0x224	R/W	PI Input Schmitt Trigger Enable Register	0x0000_0000
PI_SLEWCTL	GPIO_BA+0x228	R/W	PI High Slew Rate Control Register	0x0000_0000
PI_PUSEL	GPIO_BA+0x230	R/W	PI Pull-up Selection Register	0x0000_0000
GPIO_DBCTL	GPIO_BA+0x440	R/W	Interrupt De-bounce Control Register	0x0000_0000
INTn_INNF n=0,1..7	GPIO_BA+0x450+(0x04 * n)	R/W	INTn Input Noise Filter Register	0x0000_0000
INT_EDETCTL	GPIO_BA+0x490	R/W	INT Edge Detect Control Register	0x0000_0000
INT_EDINTEN	GPIO_BA+0x498	R/W	INT Edge Detect Interrupt Enable Control Register	0x0000_0000
INT_EDSTS	GPIO_BA+0x49C	R/W	INT Edge Detect Interrupt Flag Register	0x0000_0000
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..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

PEn_PDIO n=0,1..15	GPIO_BA+0x900+(0x04 * n)	R/W	GPIO PE.n Pin Data Input/Output Register	0x0000_000X
PFn_PDIO n=0,1..15	GPIO_BA+0x940+(0x04 * n)	R/W	GPIO PF.n Pin Data Input/Output Register	0x0000_000X
PGn_PDIO n=0,1..12	GPIO_BA+0x980+(0x04 * n)	R/W	GPIO PG.n Pin Data Input/Output Register	0x0000_000X
PHn_PDIO n=0,1..12	GPIO_BA+0x9C0+(0x04 * n)	R/W	GPIO PH.n Pin Data Input/Output Register	0x0000_000X
PIn_PDIO n=0,1..7	GPIO_BA+0xA00+(0x04 * n)	R/W	GPIO PI.n Pin Data Input/Output Register	0x0000_000X

6.6.7 Register Description

Port A-I 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	0xXXXX_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
PE_MODE	GPIO_BA+0x100	R/W	PE I/O Mode Control	0xXXXX_XXXX
PF_MODE	GPIO_BA+0x140	R/W	PF I/O Mode Control	0xXXXX_XXXX
PG_MODE	GPIO_BA+0x180	R/W	PG I/O Mode Control	0xXXXX_XXXX
PH_MODE	GPIO_BA+0x1C0	R/W	PH I/O Mode Control	0xXXXX_XXXX
PI_MODE	GPIO_BA+0x200	R/W	PI 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-I 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 initial value of this field is defined by CIOINI (CONFIG0 [10]). If CIOINI is set to 0, the default value is 0xFFFF_FFFF and all pins will be quasi-bidirectional mode after chip powered on. If CIOINI is set to 1, the default value is 0x0000_0000 and all pins will be input mode after chip powered on. **Note 2:** The PC.15/PG.0/PG.1/PG.5~PG.8/PH.0~PH.3/PH.12~PH.15/PI.6~PI.15 pins are not available

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

Register	Offset	R/W	Description	Reset Value
PA_DINOFF	GPIO_BA+0x004	R/W	PA Digital Input Path Disable Control	0x0000_0000
PB_DINOFF	GPIO_BA+0x044	R/W	PB Digital Input Path Disable Control	0x0000_0000
PC_DINOFF	GPIO_BA+0x084	R/W	PC Digital Input Path Disable Control	0x0000_0000
PD_DINOFF	GPIO_BA+0x0C4	R/W	PD Digital Input Path Disable Control	0x0000_0000
PE_DINOFF	GPIO_BA+0x104	R/W	PE Digital Input Path Disable Control	0x0000_0000
PF_DINOFF	GPIO_BA+0x144	R/W	PF Digital Input Path Disable Control	0x0000_0000
PG_DINOFF	GPIO_BA+0x184	R/W	PG Digital Input Path Disable Control	0x0000_0000
PH_DINOFF	GPIO_BA+0x1C4	R/W	PH Digital Input Path Disable Control	0x0000_0000
PI_DINOFF	GPIO_BA+0x204	R/W	PI 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-I Pin[n] Digital Input Path Disable Bit** 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:** The PC.15/PG.0/PG.1/PG.5~PG.8/PH.0~PH.3/PH.12~PH.15/PI.6~PI.15 pins are not available.
[15:0]	**Reserved**	Reserved.

Port A-I Data Output Value (Px_DOUT)

Register	Offset	R/W	Description	Reset Value
PA_DOUT	GPIO_BA+0x008	R/W	PA Data Output Value	0x0000_FFFF
PB_DOUT	GPIO_BA+0x048	R/W	PB Data Output Value	0x0000_FFFF
PC_DOUT	GPIO_BA+0x088	R/W	PC Data Output Value	0x0000_7FFF
PD_DOUT	GPIO_BA+0x0C8	R/W	PD Data Output Value	0x0000_FFFF
PE_DOUT	GPIO_BA+0x108	R/W	PE Data Output Value	0x0000_FFFF
PF_DOUT	GPIO_BA+0x148	R/W	PF Data Output Value	0x0000_FFFF
PG_DOUT	GPIO_BA+0x188	R/W	PG Data Output Value	0x0000_FE1C
PH_DOUT	GPIO_BA+0x1C8	R/W	PH Data Output Value	0x0000_0FF0
PI_DOUT	GPIO_BA+0x208	R/W	PI Data Output Value	0x0000_003F

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**DOUT[n]**	**Port A-I 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:** The PC.15/PG.0/PG.1/PG.5~PG.8/PH.0~PH.3/PH.12~PH.15/PI.6~PI.15 pins are not available.

Port A-I Data Output Write Mask (Px_DATMSK)

Register	Offset	R/W	Description	Reset Value
PA_DATMSK	GPIO_BA+0x00C	R/W	PA Data Output Write Mask	0x0000_0000
PB_DATMSK	GPIO_BA+0x04C	R/W	PB Data Output Write Mask	0x0000_0000
PC_DATMSK	GPIO_BA+0x08C	R/W	PC Data Output Write Mask	0x0000_0000
PD_DATMSK	GPIO_BA+0x0CC	R/W	PD Data Output Write Mask	0x0000_0000
PE_DATMSK	GPIO_BA+0x10C	R/W	PE Data Output Write Mask	0x0000_0000
PF_DATMSK	GPIO_BA+0x14C	R/W	PF Data Output Write Mask	0x0000_0000
PG_DATMSK	GPIO_BA+0x18C	R/W	PG Data Output Write Mask	0x0000_0000
PH_DATMSK	GPIO_BA+0x1CC	R/W	PH Data Output Write Mask	0x0000_0000
PI_DATMSK	GPIO_BA+0x20C	R/W	PI Data Output Write Mask	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[n] n=0,1..15	**DATMSK[n]**	**Port A-I 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:** The PC.15/PG.0/PG.1/PG.5~PG.8/PH.0~PH.3/PH.12~PH.15/PI.6~PI.15 pins are not available.

Port A-I 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
PE_PIN	GPIO_BA+0x110	R	PE Pin Value	0x0000_XXXX
PF_PIN	GPIO_BA+0x150	R	PF Pin Value	0x0000_XXXX
PG_PIN	GPIO_BA+0x190	R	PG Pin Value	0x0000_XXXX
PH_PIN	GPIO_BA+0x1D0	R	PH Pin Value	0x0000_XXXX
PI_PIN	GPIO_BA+0x210	R	PI 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.
[n] n=0,1..15	**PIN[n]**	**Port A-I 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:** The PC.15/PG.0/PG.1/PG.5~PG.8/PH.0~PH.3/PH.12~PH.15/PI.6~PI.15 pins are not available.

Port A-I 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
PE_DBEN	GPIO_BA+0x114	R/W	PE De-Bounce Enable Control Register	0x0000_0000
PF_DBEN	GPIO_BA+0x154	R/W	PF De-Bounce Enable Control Register	0x0000_0000
PG_DBEN	GPIO_BA+0x194	R/W	PG De-Bounce Enable Control Register	0x0000_0000
PH_DBEN	GPIO_BA+0x1D4	R/W	PH De-Bounce Enable Control Register	0x0000_0000
PI_DBEN	GPIO_BA+0x214	R/W	PI De-Bounce Enable Control Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**DBEN[n]**	**Port A-I 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:** The PC.15/PG.0/PG.1/PG.5~PG.8/PH.0~PH.3/PH.12~PH.15/PI.6~PI.15 pins are not available.

Port A-I 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
PE_INTTYPE	GPIO_BA+0x118	R/W	PE Interrupt Trigger Type Control	0x0000_0000
PF_INTTYPE	GPIO_BA+0x158	R/W	PF Interrupt Trigger Type Control	0x0000_0000
PG_INTTYPE	GPIO_BA+0x198	R/W	PG Interrupt Trigger Type Control	0x0000_0000
PH_INTTYPE	GPIO_BA+0x1D8	R/W	PH Interrupt Trigger Type Control	0x0000_0000
PI_INTTYPE	GPIO_BA+0x218	R/W	PI 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.
[n] n=0,1..15	**TYPE[n]**	**Port A-I 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:** The PC.15/PG.0/PG.1/PG.5~PG.8/PH.0~PH.3/PH.12~PH.15/PI.6~PI.15 pins are not available.

Port A-I 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
PE_INTEN	GPIO_BA+0x11C	R/W	PE Interrupt Enable Control Register	0x0000_0000
PF_INTEN	GPIO_BA+0x15C	R/W	PF Interrupt Enable Control Register	0x0000_0000
PG_INTEN	GPIO_BA+0x19C	R/W	PG Interrupt Enable Control Register	0x0000_0000
PH_INTEN	GPIO_BA+0x1DC	R/W	PH Interrupt Enable Control Register	0x0000_0000
PI_INTEN	GPIO_BA+0x21C	R/W	PI 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-I 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:** The PC.15/PG.0/PG.1/PG.5~PG.8/PH.0~PH.3/PH.12~PH.15/PI.6~PI.15 pins are not available.
[n] n=0,1..15	**FLIEN[n]**	**Port A-I 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:** The PC.15/PG.0/PG.1/PG.5~PG.8/PH.0~PH.3/PH.12~PH.15/PI.6~PI.15 pins are not available.

Port A-I Interrupt Source Flag (Px_INTSRC)

Register	Offset	R/W	Description	Reset Value
PA_INTSRC	GPIO_BA+0x020	R/W	PA Interrupt Source Flag	0x0000_XXXX
PB_INTSRC	GPIO_BA+0x060	R/W	PB Interrupt Source Flag	0x0000_XXXX
PC_INTSRC	GPIO_BA+0x0A0	R/W	PC Interrupt Source Flag	0x0000_XXXX
PD_INTSRC	GPIO_BA+0x0E0	R/W	PD Interrupt Source Flag	0x0000_XXXX
PE_INTSRC	GPIO_BA+0x120	R/W	PE Interrupt Source Flag	0x0000_XXXX
PF_INTSRC	GPIO_BA+0x160	R/W	PF Interrupt Source Flag	0x0000_XXXX
PG_INTSRC	GPIO_BA+0x1A0	R/W	PG Interrupt Source Flag	0x0000_XXXX
PH_INTSRC	GPIO_BA+0x1E0	R/W	PH Interrupt Source Flag	0x0000_XXXX
PI_INTSRC	GPIO_BA+0x220	R/W	PI Interrupt Source Flag	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
INTSRC							
7	6	5	4	3	2	1	0
INTSRC							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**INTSRC[n]**	**Port A-I 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:** The PC.15/PG.0/PG.1/PG.5~PG.8/PH.0~PH.3/PH.12~PH.15/PI.6~PI.15 pins are not available.

Port A-I 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
PE_SMTEN	GPIO_BA+0x124	R/W	PE Input Schmitt Trigger Enable Register	0x0000_0000
PF_SMTEN	GPIO_BA+0x164	R/W	PF Input Schmitt Trigger Enable Register	0x0000_0000
PG_SMTEN	GPIO_BA+0x1A4	R/W	PG Input Schmitt Trigger Enable Register	0x0000_0000
PH_SMTEN	GPIO_BA+0x1E4	R/W	PH Input Schmitt Trigger Enable Register	0x0000_0000
PI_SMTEN	GPIO_BA+0x224	R/W	PI 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.
[n] n=0,1..15	**SMTEN[n]**	**Port A-I Pin[n] Input Schmitt Trigger Enable Bit** 0 = Px.n input schmitt trigger function Disabled. 1 = Px.n input schmitt trigger function Enabled. **Note:** The PC.15/PG.0/PG.1/PG.5~PG.8/PH.0~PH.3/PH.12~PH.15/PI.6~PI.15 pins are not available.

Port A-I 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
PE_SLEWCTL	GPIO_BA+0x128	R/W	PE High Slew Rate Control Register	0x0000_0000
PF_SLEWCTL	GPIO_BA+0x168	R/W	PF High Slew Rate Control Register	0x0000_0000
PG_SLEWCTL	GPIO_BA+0x1A8	R/W	PG High Slew Rate Control Register	0x0000_0000
PH_SLEWCTL	GPIO_BA+0x1E8	R/W	PH High Slew Rate Control Register	0x0000_0000
PI_SLEWCTL	GPIO_BA+0x228	R/W	PI 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
HSREN7		HSREN6		HSREN5		HSREN4	
7	6	5	4	3	2	1	0
HSREN3		HSREN2		HSREN1		HSREN0	

Bits	Description	
[2n+1:2n] n=0,1..15	**HSRENn**	**Port A-I 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 = Reserved. 11 = Reserved. **Note:** The PC.15/PG.0/PG.1/PG.5~PG.8/PH.0~PH.3/PH.12~PH.15/PI.6~PI.15 pins are not available.

Port A-I Pull-up Selection Register (Px_PUSEL)

Register	Offset	R/W	Description	Reset Value
PA_PUSEL	GPIO_BA+0x030	R/W	PA Pull-up Selection Register	0x0000_0000
PB_PUSEL	GPIO_BA+0x070	R/W	PB Pull-up Selection Register	0x0000_0000
PC_PUSEL	GPIO_BA+0x0B0	R/W	PC Pull-up Selection Register	0x0000_0000
PD_PUSEL	GPIO_BA+0x0F0	R/W	PD Pull-up Selection Register	0x0000_0000
PE_PUSEL	GPIO_BA+0x130	R/W	PE Pull-up Selection Register	0x0000_0000
PF_PUSEL	GPIO_BA+0x170	R/W	PF Pull-up Selection Register	0x0000_0000
PG_PUSEL	GPIO_BA+0x1B0	R/W	PG Pull-up Selection Register	0x0000_0000
PH_PUSEL	GPIO_BA+0x1F0	R/W	PH Pull-up Selection Register	0x0000_0000
PI_PUSEL	GPIO_BA+0x230	R/W	PI Pull-up Selection Register	0x0000_0000

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

Bits	Description	
[2n] n=0,1..15	**PUSELn**	**Port A-I Pin[n] Pull-up Enable Register** Determine each I/O Pull-up of Px.n pins. 0 = Px.n pull-up disable. 1 = Px.n pull-up enable. **Note 1:** Basically, the pull-up control has following behavior limitation The independent pull-up control register only valid when MODEn set as tri-state and open-drain mode **Note 2:** The PC.15/PG.0/PG.1/PG.5~PG.8/PH.0~PH.3/PH.12~PH.15/PI.6~PI.15 pins are not available.

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_0000

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

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5]	**ICLKON**	**Interrupt Clock on Mode** 0 = Edge detection circuit is active only if I/O pin corresponding RHIEN (Px_INTEN[n+16])/FLIEN (Px_INTEN[n]) bit is set to 1. If corresponding RHIEN or FLIEN not set to 1, the clock of IO detect circuit is stopped and INTSRC flag cannot be clear also. 1 = All I/O pins edge detection circuit is always active after reset. **Note 1:** It is recommended to disable this bit to save system power if no special application concern. **Note 2:**The PC.15/PG.0/PG.1/PG.5~PG.8/PH.0~PH.3/PH.12~PH.15/PI.6~PI.15 pins are not available.
[4]	**DBCLKSRC**	**De-bounce Counter Clock Source Selection** 0 = De-bounce counter clock source is the HCLK. 1 = De-bounce counter clock source is the internal low speed RC oscillator (LIRC).
[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.

INT0~7 Input Noise Filter Register (INTx_INNF)

Register	Offset	R/W	Description	Reset Value
INTn_INNF N=0,1..7	GPIO_BA+0x450+(0x04 * n)	R/W	INTn Input 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
Reserved					NFCNT		
7	6	5	4	3	2	1	0
Reserved	NFSEL			Reserved			NFEN

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10:8]	**NFCNT**	**Noise Filter Count** The register bits control the filter counter to count from 0 to NFCNT.
[7]	**Reserved**	Reserved.
[6:4]	**NFSEL**	**Noise 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.
[3:1]	**Reserved**	Reserved.
[0]	**NFEN**	**Noise Filter Enable** 0 = Noise Filter function Disabled. 1 = Noise Filter function Enabled.

INT Edge Detect Control Register (INT_EDETCTL)

Register	Offset	R/W	Description	Reset Value
INT_EDETCTL	GPIO_BA+0x490	R/W	INT Edge Detect Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reseved							
23	22	21	20	19	18	17	16
Reseved							
15	14	13	12	11	10	9	8
EDETCTL7		EDETCTL6		EDETCTL5		EDETCTL4	
7	6	5	4	3	2	1	0
EDETCTL3		EDETCTL2		EDETCTL1		EDETCTL0	

Bits	Description	
[31:16]	Reseved	Reseved
[2n+1:2n] n=0,1..7	EDETCTLn	INTn Edge Detect Control Bits 00 = Not detect. 01 = INTn low to high detection Enable. 10 = INTn high to low detection Enable. 11 = INTn both low to high and high to low detection Enable.

INT Egde Detect Interrupt Enable Control Register (INT_EDINTEN)

Register	Offset	R/W	Description	Reset Value
INT_EDINTEN	GPIO_BA+0x498	R/W	INT Edge Detect Interrupt Enable Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reseved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reseved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reseved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EDIEN7	**EDIEN6**	**EDIEN5**	**EDIEN4**	**EDIEN3**	**EDIEN2**	**EDIEN1**	**EDIEN0**

Bits	Description	
[31:8]	**Reseved**	**Reseved**
[n] n=0,1..7	**EDIENn**	**INTn Edge Detect Interrupt Enable Bit** 0 = INTx Edge Detect Interrupt Disable. 1 = INTx Edge Detect Interrupt Enable.

INT Edge Detect Interrupt Flag Register (INT_EDSTS)

Register	Offset	R/W	Description	Reset Value
INT_EDSTS	GPIO_BA+0x49C	R/W	INT Edge Detect Interrupt Flag Register	0x0000_0000

31	30	29	28	27	26	25	24
Reseved							
23	22	21	20	19	18	17	16
Reseved							
15	14	13	12	11	10	9	8
Reseved							
7	6	5	4	3	2	1	0
EDIF7	EDIF6	EDIF5	EDIF4	EDIF3	EDIF2	EDIF1	EDIF0

Bits	Description	
[31:8]	Reseved	Reseved
[n] n=0,1..7	EDIFn	**INTn Edge Detect Interrupt Flag** 0 = No Edge Detection happened. 1 = Rising Edge or Falling edge has been detected. **Note:** This bit is cleared by writing 1 to it.

GPIO Px.n Pin Data Input/Outut 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..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
PEn_PDIO n=0,1..15	GPIO_BA+0x900+(0x04 * n)	R/W	GPIO PE.n Pin Data Input/Output Register	0x0000_000X
PFn_PDIO n=0,1..15	GPIO_BA+0x940+(0x04 * n)	R/W	GPIO PF.n Pin Data Input/Output Register	0x0000_000X
PGn_PDIO n=0,1..12	GPIO_BA+0x980+(0x04 * n)	R/W	GPIO PG.n Pin Data Input/Output Register	0x0000_000X
PHn_PDIO n=0,1..12	GPIO_BA+0x9C0+(0x04 * n)	R/W	GPIO PH.n Pin Data Input/Output Register	0x0000_000X
PIn_PDIO n=0,1..5	GPIO_BA+0xA00+(0x04 * n)	R/W	GPIO PI.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.
[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 (PA_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 PC.15/PG.0/PG.1/PG.5~PG.8/PH.0~PH.3/PH.12~PH.15/PI.6~PI.15 pins are not available.

6.7 PDMA Controller (PDMA)

6.7.1 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 6 channels and each channel can perform transfer between memory and peripherals or between memory and memory.

6.7.2 Features

- Supports 6 independently configurable channels
- 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 UART, SPI, EPWM, TIMER, EADC, DAC, ACMP and I^2C 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

6.7.3 Block Diagram

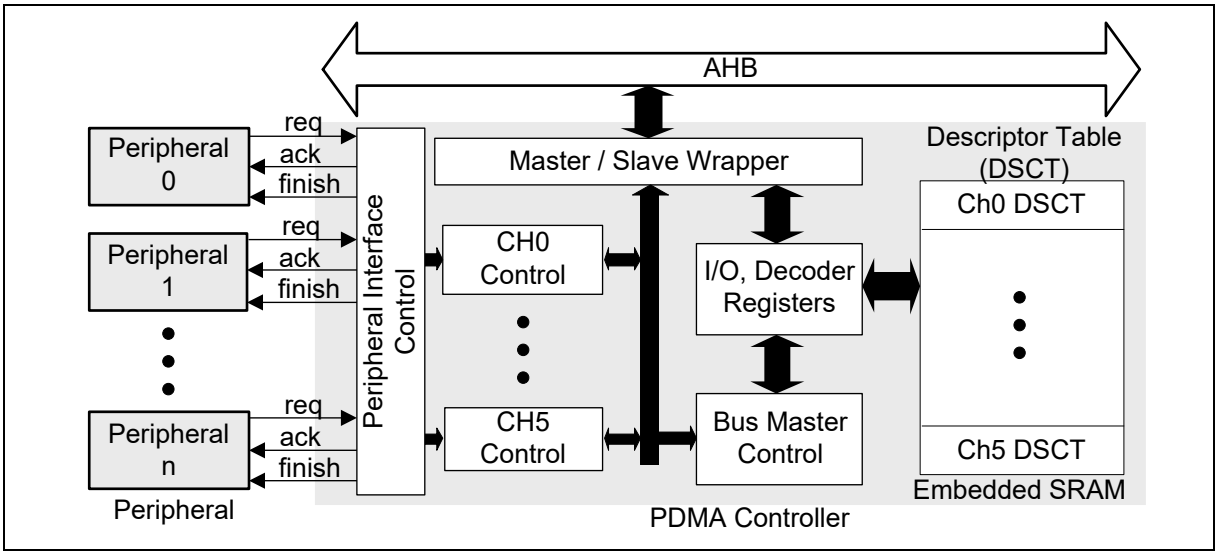


Figure 6.7-1 PDMA Controller Block Diagram

6.7.4 Basic Configuration

- Clock Source Configuration
  - Enable PDMA controller clock in PDMACKEN (CLK_AHBCLK [1]).
- Reset Configuration

- Reset PDMA controller in PDMARST (SYS_IPRST0[2]).

6.7.5 Functional Description

The PDMA controller transfers data from one address to another without CPU intervention. The PDMA controller supports 6 independent channels and serves only one channel at one time, as the result, PDMA controller supports two level channel priorities: fixed and round-robin priority, PDMA controller serves channel in order from highest to lowest priority channel. The PDMA controller supports two operation modes: Basic mode and Scatter-gather mode. Basic mode is used to perform one descriptor table transfer. Scatter-gather mode has more entries for each PDMA channel, and thus the PDMA controller supports sophisticated transfer through the entries. The descriptor table entry data structure contains many transfer information including the transfer source address, transfer destination address, transfer count, burst size, transfer type and operation mode. Figure 6.7-2 shows the diagram of descriptor table (DSCT) data structure.

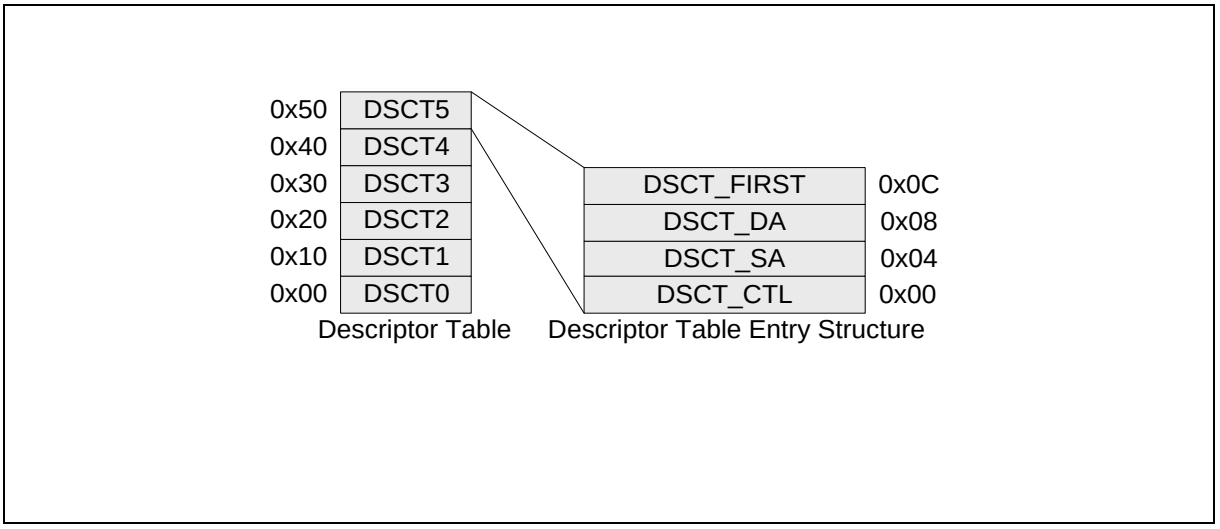


Figure 6.7-2 Descriptor Table Entry Structure

The 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).

6.7.5.1 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 6.7-1.

PDMA_PRISET	Channel Number	Priority Setting	Arbitration Priority In Descending Order
1	5	Channel5, Fixed Priority	Highest
1	4	Channel4, Fixed Priority	---
---	---	---	---
1	0	Channel0, Fixed Priority	---

0	5	Channel5, Round-Robin Priority	---
0	4	Channel4, Round-Robin Priority	---
---	---	---	---
0	0	Channel0, Round-Robin Priority	Lowest

Table 6.7-1 Channel Priority Table

6.7.5.2 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, peripherals and memory or peripherals and peripherals. However, if user wants to transfer data between peripherals and peripherals, the user must ensure that the request from peripherals knows that the data is ready for transfer or not. PDMA controller operation mode can be set from OPMODE (PDMA_DSCTn_CTL[1:0], where n denotes PDMA channel), the default setting is in idle state (OPMODE (PDMA_DSCTn_CTL[1:0]) = 0x0) and it is recommended that user configure the descriptor table in idle state. If operation mode is not in idle state, user re-configure channel setting may lead to 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 a request signal. Finishing this task will generate an interrupt to CPU if the corresponding PDMA interrupt bit INTENn (PDMA_INTEN[5:0]) is enabled and the operation mode will be updated to idle state as shown in Figure 6.7-3. 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.

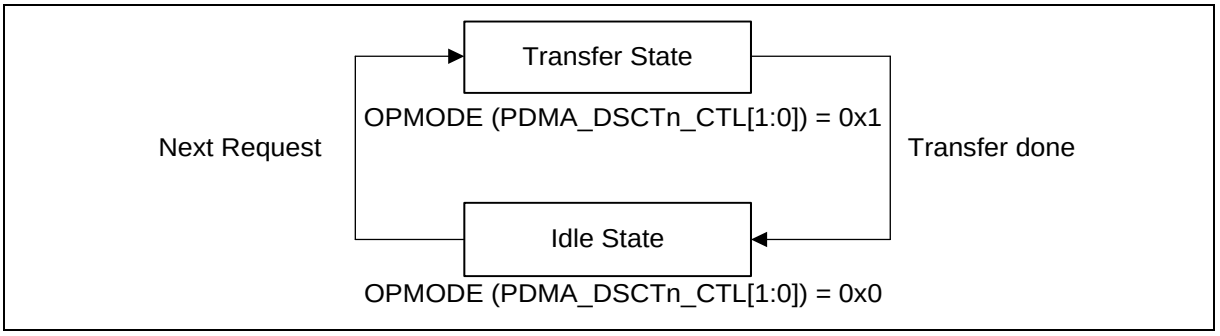


Figure 6.7-3 Basic Mode Finite State Machine

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 6.7-4. Through operation mode user can perform peripheral wrapper-around, and multiple PDMA task can be used for data transfer between varied locations in system memory instead of a set of contiguous locations. Scatter-gather mode only needs a request to finish all table entries task till the last task with OPMODE (PDMA_DSCTn_CTL[1:0]) is idle state without ack. It also means scatter-gather mode can be used 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[5:0]) flag will be asserted and if this bit is “1” TDIFn will not be active).

If channel 5 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 16 bits valid in PDMA_SCATBA), the current link address is 0x0000_0100 (only LSB 16-bits without last two bits [1:0] valid in PDMA_DSCTn_NEXT), and then the next DSCT entry start address is 0x2000_0100. Please note that, only LSB 16-bits without last two bits [1:0] will cause scatter-gather descriptor table only supports to SRAM address less than 64 Kbytes.

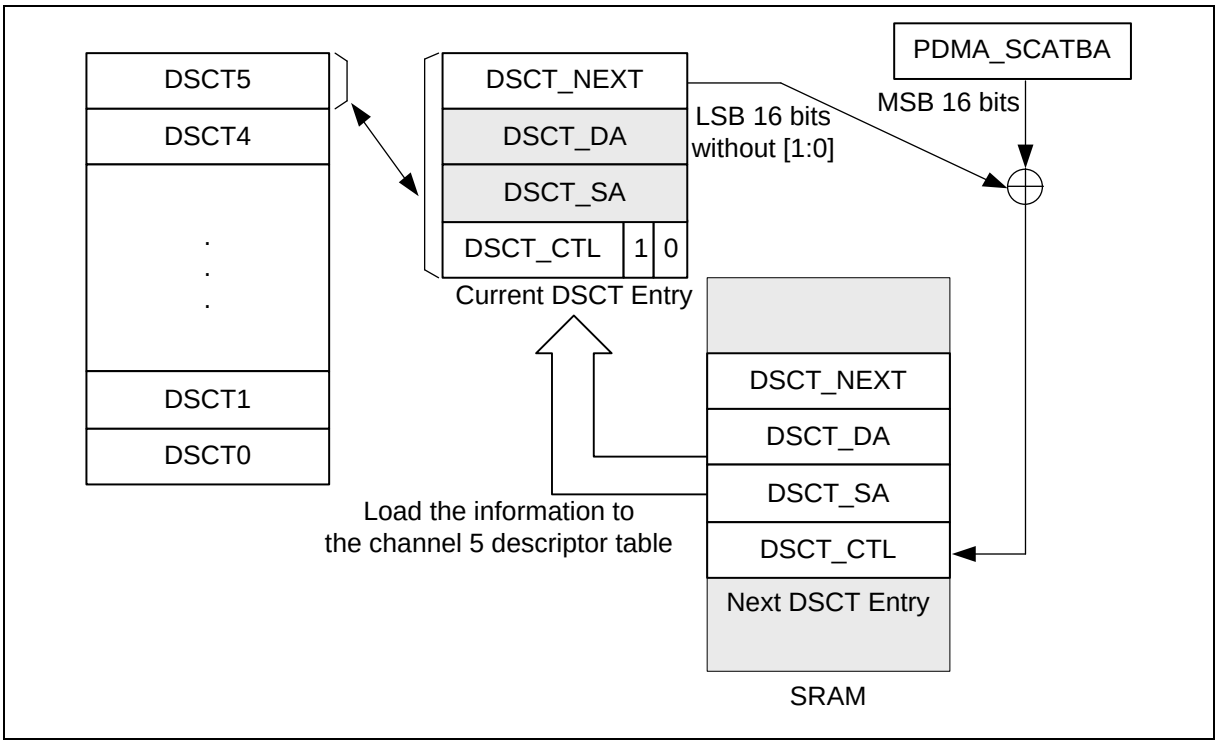


Figure 6.7-4 Descriptor Table Link List Structure

The above link list table operation is DSCT state in Scatter-Gather Mode as shown in Figure 6.7-5. 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 set to Scatter-Gather mode, the hardware will catch the next PDMA information block when the current task is finished. The Scatter-Gather mode switches to basic mode when doing the next task. Then, the basic mode switches to Idle state when the last task is finished.

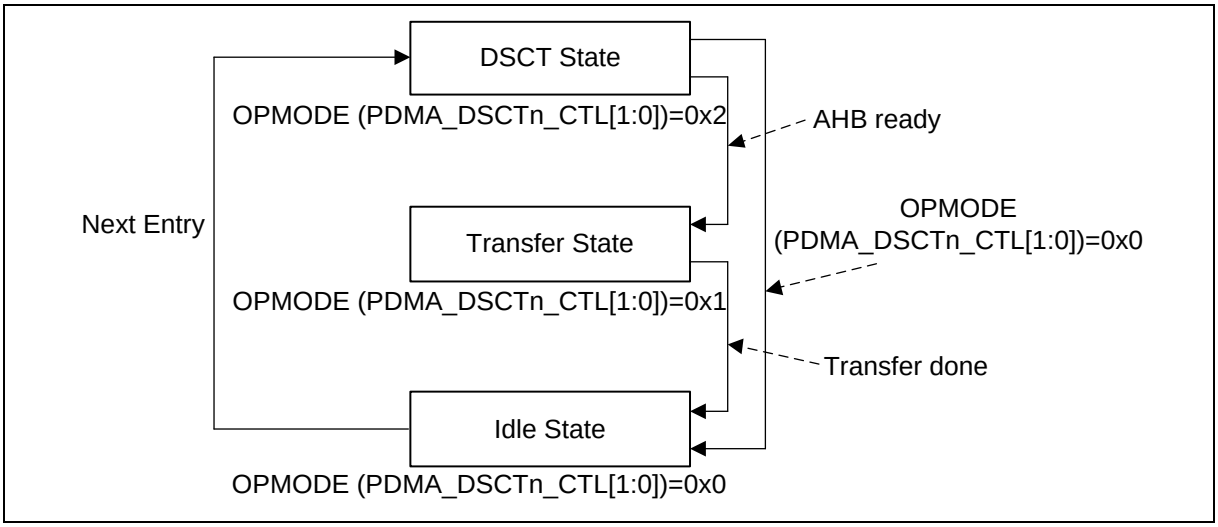


Figure 6.7-5 Scatter-Gather Mode Finite State Machine

6.7.5.3 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 the PDMA controller is 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 be finished after the TXCNT (PDMA_DSCTn_CTL[31:16]) decreases 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.

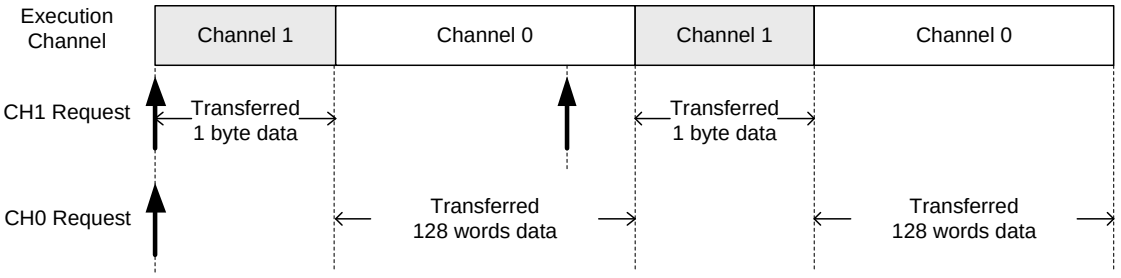
For the burst transfer type, the PDMA controller transfers TXCNT (PDMA_DSCTn_CTL[31:16]) of data and need only one request signal. After transferred BURSIZE (PDMA_DSCTn_CTL[6:4]) of data, TXCNT (PDMA_DSCTn_CTL[31:16]) will decrease BURSIZE number. Transfer will be done after the transfer count TXCNT (PDMA_DSCTn_CTL[31:16]) decreases 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. Please note that, PDMA transfer data between Flash and memory should finish before MCU enter idle mode or power done mode to prevent access wrong data

Figure 6.7-6 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]) = 128 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, and thus the PDMA controller will only transfer one transfer data.
3. Then, the 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, the 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, the 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. The 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 6.7-6 Example of Single Transfer Type and Burst Transfer Type in Basic Mode

6.7.5.4 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 will restart from 0 while counter count to TOCn (PDMA_TOC0_1 [16(n+1)-1:16n], n=0,1), received trigger signal, time-out function is disabled or chip enters Power-down mode. The time-out counter will keep counting until time-out function is disabled.

Figure 6.7-7 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 setting TOUTEN0(PDMA_TOUTEN[0]) bit to 1.
2. Time-out counter starts counting from 0 to the value of TOC0(PDMA_TOC0_1[15:0]) bits when receiving the 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.

5. Time-out counter is reset to 0 when time-out function is disabled.

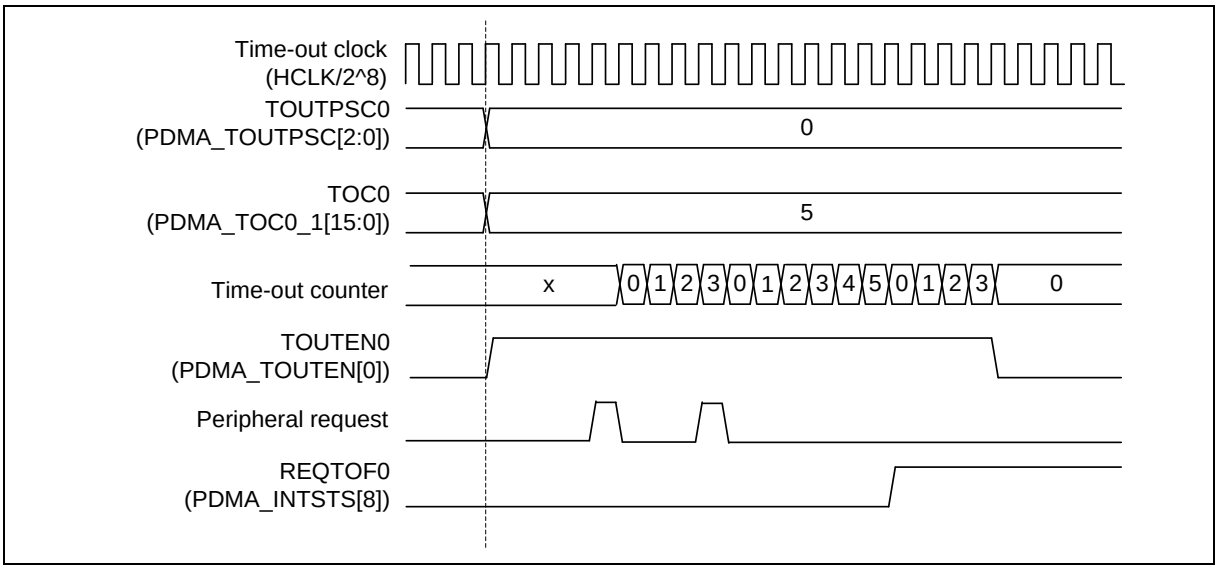


Figure 6.7-7 Example of PDMA Channel 0 Time-out Counter Operation

6.7.6 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_DSCTn_CTL **n = 0,1..5**	PDMA_BA+0x10*n	R/W	Descriptor Table Control Register of PDMA Channel n	0xXXXX_XXXX
PDMA_DSCTn_SA **n = 0,1..5**	PDMA_BA+0x0004+0x10*n	R/W	Source Address Register of PDMA Channel n	0xXXXX_XXXX
PDMA_DSCTn_DA **n = 0,1..5**	PDMA_BA+0x0008+0x10*n	R/W	Destination Address Register of PDMA Channel n	0xXXXX_XXXX
PDMA_DSCTn_NEXT **n = 0,1..5**	PDMA_BA+0x000c+0x10*n	R/W	Next Scatter-gather Descriptor Table Offset Address of PDMA Channel n	0xXXXX_XXXX
PDMA_CURSCATn **n = 0,1..5**	PDMA_BA+0x0100+0x004*n	R	Current Scatter-gather Descriptor Table Address of PDMA Channel n	0xXXXX_XXXX
PDMA_CHCTL	PDMA_BA + 0x400	R/W	PDMA Channel Control Register	0x0000_0000
PDMA_PAUSE	PDMA_BA + 0x404	W	PDMA Transfer Pause Control Register	0x0000_0000
PDMA_SWREQ	PDMA_BA + 0x408	W	PDMA Software Request Register	0x0000_0000
PDMA_TRGSTS	PDMA_BA + 0x40C	R	PDMA Channel Request Status Register	0x0000_0000
PDMA_PRISET	PDMA_BA + 0x410	R/W	PDMA Fixed Priority Setting Register	0x0000_0000
PDMA_PRICLR	PDMA_BA + 0x414	W	PDMA Fixed Priority Clear Register	0x0000_0000
PDMA_INTEN	PDMA_BA + 0x418	R/W	PDMA Interrupt Enable Register	0x0000_0000
PDMA_INTSTS	PDMA_BA + 0x41C	R/W	PDMA Interrupt Status Register	0x0000_0000
PDMA_ABTSTS	PDMA_BA + 0x420	R/W	PDMA Channel Read/Write Target Abort Flag Register	0x0000_0000
PDMA_TDSTS	PDMA_BA + 0x424	R/W	PDMA Channel Transfer Done Flag Register	0x0000_0000
PDMA_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
PDMA_REQSEL0_3	PDMA_BA + 0x480	R/W	PDMA Request Source Select Register 0	0x0000_0000
PDMA_REQSEL4_5	PDMA_BA + 0x484	R/W	PDMA Request Source Select Register 1	0x0000_0000

6.7.7 Register Description

Descriptor Table Control Register (PDMA_DSCTn_CTL)

Register	Offset	R/W	Description	Reset Value
PDMA_DSCTn_CTL	PDMA_BA+0x10*n	R/W	Descriptor Table Control Register of PDMA Channel n	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
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 finishes each transfer data, this field will be decrease immediately.
[15:14]	**Reserved**	Reserved.
[13:12]	**TXWIDTH**	**Transfer Width Selection** This field is used for transfer width. 00 = One byte (8 bit) is transferred for every operation. 01= One half-word (16 bit) is transferred for every operation. 10 = One word (32-bit) is transferred for every operation. 11 = Reserved. **Note:** The PDMA transfer source address (PDMA_DSCT_SA) and PDMA transfer destination address (PDMA_DSCT_DA) should be alignment under the TXWIDTH selection
[11:10]	**DAINC**	**Destination Address Increment** This field is used to set the destination address increment size. 11 = No increment (fixed address). Others = Increment and size is depended on TXWIDTH selection. **Note:** The fixed address function is not supported in memory to memory transfer type.
[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. **Note:** The fixed address function is not supported in memory to memory transfer type.
[7]	**TBINTDIS**	**Table Interrupt Disable Bit** This field can be used to decide whether to enable table interrupt or not. If the TBINTDIS bit is enabled it will

Bits	Description	
		not generates TDIFn(PDMA_TDSTS[15:0]) when PDMA controller finishes transfer task. 0 = Table interrupt Enabled. 1 = Table interrupt Disabled. **Note:** This function only for scatter-gather mode.
[6:4]	**BURSIZE**	**Burst Size** 000 = 128 Transfers. 001 = 64 Transfers. 010 = 32 Transfers. 011 = 16 Transfers. 100 = 8 Transfers. 101 = 4 Transfers. 110 = 2 Transfers. 111 = 1 Transfers. **Note:** This field is only useful in burst transfer type.
[3]	**Reserved**	Reserved.
[2]	**TXTYPE**	**Transfer Type** 0 = Burst transfer type. 1 = Single transfer type.
[1:0]	**OPMODE**	**PDMA Operation Mode Selection** 00 = Idle state: Channel is stopped or this table is complete, when PDMA finish channel table task, OPMODE will be cleared to idle state automatically. 01 = Basic mode: The descriptor table only has one task. When this task is finished, the PDMA_INTSTS[1] 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. **Note:** Before filling new transfer task in the Descriptor Table, user must check the PDMA_INTSTS[1] to make sure the current task is complete.

Start Source Address Register (PDMA_DSCTn_SA)

Register	Offset	R/W	Description	Reset Value
PDMA_DSCTn_SA	PDMA_BA+0x0004+0x10*n	R/W	Source Address Register of PDMA Channel n	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** This field indicates a 32-bit source address of PDMA controller.

Destination Address Register (PDMA_DSCTn_DA)

Register	Offset	R/W	Description	Reset Value
PDMA_DSCTn_DA	PDMA_BA+0x0008+0x10*n	R/W	Destination Address Register of PDMA Channel n	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** This field indicates a 32-bit destination address of PDMA controller.

Next Scatter-gather Descriptor Table Offset Address (PDMA_DSCTn_NEXT)

Register	Offset	R/W	Description	Reset Value
PDMA_DSCTn_NEXT	PDMA_BA+0x000c+0x10*n	R/W	Next Scatter-gather Descriptor Table Offset Address of PDMA Channel n	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. When operating in scatter-gather mode, the last two bits NEXT[1:0] will become scatter-gather mode control indicator as below. 0 = Idle mode. 1 = operating in the basic mode (final scatter-gather table). 2 = loading scatter-gather table from SRAM. 3 = operating in the scatter-gather mode. **Note 1:** The 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.

Current Scatter-gather Descriptor Table Address (PDMA_CURSCATn)

Register	Offset	R/W	Description	Reset Value
PDMA_CURSCATn	PDMA_BA+0x0100+0x004*n	R	Current Scatter-gather Descriptor Table Address of PDMA Channel n	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 (Read Only)** This field indicates a 32-bit current external description address of PDMA controller. **Note:** This field is read only and used for Scatter-Gather mode only to indicate the current external description address.

Channel Control Register (PDMA_CHCTL)

Register	Offset	R/W	Description	Reset Value
PDMA_CHCTL	PDMA_BA + 0x400	R/W	PDMA Channel Control Register	0x0000_0000

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

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**CHENn**	**PDMA Channel Enable Bits** Set this bit to 1 to enable PDMAn operation. Channel cannot be active if it is not set as enabled. 0 = PDMA channel [n] Disabled. 1 = PDMA channel [n] Enabled. **Note:** Setting the corresponding bit of PDMA_PAUSE or PDMA_CHRST register will also clear this bit.

PDMA Transfer Pause Control Register (PDMA_PAUSE)

Register	Offset	R/W	Description	Reset Value
PDMA_PAUSE	PDMA_BA + 0x404	W	PDMA Transfer Pause Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		PAUSE5	PAUSE4	PAUSE3	PAUSE2	PAUSE1	PAUSE0

Bits	Description	
[31:6]	Reserved	Reserved.
[n] n=0,1..5	PAUSEn	**PDMA Channel n Transfer Pause Control (Write Only)** User can set PAUSEn bit field to pause the PDMA transfer. When user sets PAUSEn bit, the PDMA controller will pause the on-going transfer, then clear the channel enable bit CHEN(PDMA_CHCTL [n], n=0,1..5) and clear request active flag(PDMA_TRGSTS[n:0], n=0,1..5). If the paused channel is re-enabled again, the remaining transfers will be processed. 0 = No effect. 1 = Pause PDMA channel n 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
Reserved							
7	6	5	4	3	2	1	0
Reserved		SWREQ5	SWREQ4	SWREQ3	SWREQ2	SWREQ1	SWREQ0

Bits	Description	
[31:6]	Reserved	Reserved.
[n] n=0,1..5	SWREQn	**PDMA Software Request (Write Only)** Set this bit to 1 to generate a software request to PDMA [n]. 0 = No effect. 1 = Generate a software request. **Note 1:** User can read PDMA_TRGSTS register to know which channel is on active. Active flag may be triggered by software request or peripheral request. **Note 2:** If user does not enable corresponding PDMA channel, the software request will be ignored.

PDMA Channel Request Status Register (PDMA_TRGSTS)

Register	Offset	R/W	Description	Reset Value
PDMA_TRGSTS	PDMA_BA + 0x40C	R	PDMA Channel Request Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		REQSTS5	REQSTS4	REQSTS3	REQSTS2	REQSTS1	REQSTS0

Bits	Description	
[31:6]	Reserved	Reserved.
[n] n=0,1..5	REQSTSn	**PDMA Channel Request Status (Read Only)** This flag indicates whether channel[n] have a request or not, no matter request from software or peripheral. When PDMA controller finishes channel transfer, this bit will be cleared automatically. 0 = PDMA Channel n has no request. 1 = PDMA Channel n has a request. **Note:** If user pauses or resets each PDMA transfer by setting PDMA_PAUSE or PDMA_CHRST register respectively, this bit will be cleared automatically after finishing the current transfer.

PDMA Fixed Priority Setting Register (PDMA_PRISET)

Register	Offset	R/W	Description	Reset Value
PDMA_PRISET	PDMA_BA + 0x410	R/W	PDMA Fixed Priority Setting Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**FPRISET5**	**FPRISET4**	**FPRISET3**	**FPRISET2**	**FPRISET1**	**FPRISET0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**FPRISETn**	**PDMA Fixed Priority Setting** Set this bit to 1 to enable fixed priority level. Write Operation: 0 = No effect. 1 = Set PDMA channel [n] to fixed priority channel. Read Operation: 0 = Corresponding PDMA channel is round-robin priority. 1 = Corresponding PDMA channel is fixed priority. **Note:** This field is only set to fixed priority; to 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
Reserved							
7	6	5	4	3	2	1	0
Reserved		FPRICLR5	FPRICLR4	FPRICLR3	FPRICLR2	FPRICLR1	FPRICLR0

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**FPRICLRn**	**PDMA Fixed Priority Clear Bits (Write Only)** Set this bit to 1 to clear fixed priority level. 0 = No effect. 1 = Clear PDMA channel [n] fixed priority setting. **Note:** User can read PDMA_PRISET register to know the channel priority.

PDMA Interrupt Enable Register (PDMA_INTEN)

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

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**INTEN5**	**INTEN4**	**INTEN3**	**INTEN2**	**INTEN1**	**INTEN0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**INTENn**	**PDMA Interrupt Enable Bits** This field is used to enable PDMA channel[n] interrupt. 0 = PDMA channel n interrupt Disabled. 1 = PDMA channel n interrupt Enabled. **Note:** The interrupt flag is time-out, abort, transfer done and align.

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.
[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. **Note:** Please disable time-out function before clearing this bit.
[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. **Note:** Please disable time-out function before clearing this bit.
[7:3]	**Reserved**	Reserved.
[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. 1 = AHB bus ERROR response received.

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

Register	Offset	R/W	Description	Reset Value
PDMA_ABTSTS	PDMA_BA + 0x420	R/W	PDMA Channel Read/Write Target Abort Flag Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		ABTIF5	ABTIF4	ABTIF3	ABTIF2	ABTIF1	ABTIF0

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**ABTIFn**	**PDMA Read/Write Target Abort Interrupt Status Flag** This bit indicates which PDMA controller has target abort error; User can write 1 to clear these bits. 0 = No AHB bus ERROR response received when channel n transfer. 1 = AHB bus ERROR response received when channel n transfer. **Note:** If channel n target abort, REQSRCn should set 0 to disable peripheral request.

PDMA Channel Transfer Done Flag Register (PDMA_TDSTS)

Register	Offset	R/W	Description	Reset Value
PDMA_TDSTS	PDMA_BA + 0x424	R/W	PDMA Channel Transfer Done Flag Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		TDIF5	TDIF4	TDIF3	TDIF2	TDIF1	TDIF0

Bits	Description	
[31:6]	Reserved	Reserved.
[n] n=0,1..5	TDIFn	**Transfer Done Flag** This bit indicates whether PDMA controller channel transfer has been finished or not, user can write 1 to clear these bits. 0 = PDMA channel transfer has not finished. 1 = PDMA channel has finished transmission.

PDMA 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**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**ALIGN5**	**ALIGN4**	**ALIGN3**	**ALIGN2**	**ALIGN1**	**ALIGN0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**ALIGNn**	**Transfer Alignment Flag** 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. **Note:** Source address and destination address should be alignment.

PDMA Transfer Active Flag Register (PDMA_TACTSTS)

Register	Offset	R/W	Description	Reset Value
PDMA_TACTSTS	PDMA_BA + 0x42C	R	PDMA Transfer Active Flag Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		TXACTF5	TXACTF4	TXACTF3	TXACTF2	TXACTF1	TXACTF0

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**TXACTFn**	**Transfer on Active Flag (Read Only)** This bit indicates which PDMA channel is in active. 0 = PDMA channel is finished. 1 = PDMA channel is active.

PDMA Time-out Prescaler Register (PDMA_TOUTPSC)

Register	Offset	R/W	Description	Reset Value
PDMA_TOUTPSC	PDMA_BA + 0x430	R/W	PDMA Time-out Prescaler Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	TOUTPSC1			Reserved	TOUTPSC0		

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

PDMA Time-out Enable Register (PDMA_TOUTEN)

Register	Offset	R/W	Description	Reset Value
PDMA_TOUTEN	PDMA_BA + 0x434	R/W	PDMA Time-out Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**TOUTEN1**	**TOUTEN0**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[n] n=0,1	**TOUTENn**	**PDMA Time-out Enable Bits** 0 = PDMA Channel n time-out function Disabled. 1 = PDMA Channel n time-out function Enabled.

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.
[n] n=0,1	**TOUTIENn**	**PDMA Time-out Interrupt Enable Bits** 0 = PDMA Channel n time-out interrupt Disabled. 1 = PDMA Channel n time-out interrupt Enabled.

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** 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.

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

PDMA Channel Reset 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
Reserved							
7	6	5	4	3	2	1	0
Reserved		CH5RST	CH4RST	CH3RST	CH2RST	CH1RST	CH0RST

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**CHnRST**	**Channel n Reset** 0 = corresponding channel n is not reset. 1 = corresponding channel n is reset.

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.
[29:24]	**REQSRC3**	**Channel 3 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 3. User can configure the peripheral setting by REQSRC3. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[23:22]	**Reserved**	Reserved.
[21:16]	**REQSRC2**	**Channel 2 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 2. User can configure the peripheral setting by REQSRC2. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[15:14]	**Reserved**	Reserved.
[13:8]	**REQSRC1**	**Channel 1 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 1. User can configure the peripheral setting by REQSRC1. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[7:6]	**Reserved**	Reserved.
[5:0]	**REQSRC0**	**Channel 0 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 0. User can configure the peripheral by setting REQSRC0. 0 = Disable PDMA peripheral request. 1 = Channel connects to UART0_TX. 2 = Channel connects to UART0_RX. 3 = Channel connects to UART1_TX. 4 = Channel connects to UART1_RX. 5 = Channel connects to UART2_TX.

Bits	Description	
		6 = Channel connects to UART2_RX. 7=Channel connects to UART3_TX. 8 = Channel connects to UART3_RX. 9 = Channel connects to UART4_TX. 10 = Channel connects to UART4_RX. 11 = Channel connects to UART5_TX. 12 = Channel connects to UART5_RX. 13 = Channel connects to SPI0_TX. 14 = Channel connects to SPI0_RX. 15 = Channel connects to SPI1_TX. 16 = Channel connects to SPI1_RX. 17 = Channel connects to EPWM0_P1_RX. 18 = Channel connects to EPWM0_P2_RX. 19 = Channel connects to EPWM0_P3_RX. 20 = Channel connects to EPWM1_P1_RX. 21 = Channel connects to EPWM1_P2_RX. 22 = Channel connects to EPWM1_P3_RX. 23 = Channel connects to I2C0_TX. 24 = Channel connects to I2C0_RX. 25 = Channel connects to I2C1_TX. 26 = Channel connects to I2C1_RX. 27 = Channel connects to TMR0. 28 = Channel connects to TMR1. 29 = Channel connects to TMR2. 30 = Channel connects to TMR3. 31 = Channel connects to EADC0_RX. 32 = Channel connects to DAC0_TX. 33 = Channel connects to EPWM0_CH0_TX. 34 = Channel connects to EPWM0_CH1_TX. 35 = Channel connects to EPWM0_CH2_TX. 36 = Channel connects to EPWM0_CH3_TX. 37 = Channel connects to EPWM0_CH4_TX. 38 = Channel connects to EPWM0_CH5_TX. 39 = Channel connects to EPWM1_CH0_TX. 40 = Channel connects to EPWM1_CH1_TX. 41 = Channel connects to EPWM1_CH2_TX. 42 = Channel connects to EPWM1_CH3_TX. 43 = Channel connects to EPWM1_CH4_TX. 44 = Channel connects to EPWM1_CH5_TX. 45 = Channel connects to ACMP0. 46 = Channel connects to ACMP1. 47 = Reserved. 48 = Reserved. 49 = Reserved. 50 = Reserved. Others = Reserved. **Note 1:** A peripheral cannot be assigned 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_5)

Register	Offset	R/W	Description	Reset Value
PDMA_REQSEL4_5	PDMA_BA + 0x484	R/W	PDMA Request 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**
Reserved		**REQSRC5**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**REQSRC4**					

Bits	Description	
[31:13]	**Reserved**	Reserved.
[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.
[5:0]	**REQSRC4**	**Channel 4 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 4. User can configure the peripheral setting by REQSRC4. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.

6.8 Timer Controller (TMR)

6.8.1 Overview

The timer controller includes four 32-bit timers, Timer0 ~ Timer3, allowing user to easily implement a timer control for applications. The timer can perform functions, such as frequency measurement, delay timing, clock generation, and event counting by external input pins, and interval measurement by external capture pins.

The timer controller also provides four PWM generators. Each PWM generator supports one PWM output and two selectable PWM output channels (TMx or TMx_EXT). The output state of PWM output pin can be control by polarity control, output enable control and output channel select.

6.8.2 Features

6.8.2.1 Timer Function Features

- Four sets of 32-bit timers, each timer having one 24-bit up counter and one 8-bit prescale counter
- Independent clock source for each timer
- Provides one-shot, periodic, toggle-output and continuous counting operation modes
- 24-bit up counter value is readable through CNT (TIMERx_CNT[23:0])
- Supports event counting function
- 24-bit capture value is readable through CAPDAT (TIMERx_CAP[23:0])
- Supports external capture pin (TMx_EXT) event for interval measurement
- Supports external capture pin (TMx_EXT) event to reset 24-bit up counter
- Supports internal clock (HIRC, LIRC) and external clock (HXT, LXT) for capture event
- Supports chip wake-up from Idle/Power-down mode if a timer interrupt signal is generated
- Support Timer0 ~ Timer3 time-out interrupt signal or capture interrupt signal to trigger BPWM, EPWM, EADC, DAC and PDMA function
- Supports internal capture triggered while internal ACMP output signal transition
- Supports Inter-Timer trigger mode

6.8.2.2 PWM Function Features

- Supports PWM generator with two selectable output channels
- Supports 16-bit PWM counter
  - Up count operation type
  - One-shot or auto-reload counter operation mode
- Supports 8-bit prescale from 1 to 256
- Supports 16-bit compare register and period register and double buffer for period register and compare register
- Supports tri-state enable and polarity control for each PWM selectable output channels
- Supports interrupt on the following events:
  - PWM period point, up-count compared point events
- Supports wake-up when interrupt occurs when clock source is LXT or LIRC

- PWM can generator output in Power-down mode
- Supports trigger EADC, PDMA, and DAC on the following events:
  - PWM period point and up-count compared point events

6.8.4 Block Diagram

The timer controller block diagram and clock control are shown as follows.

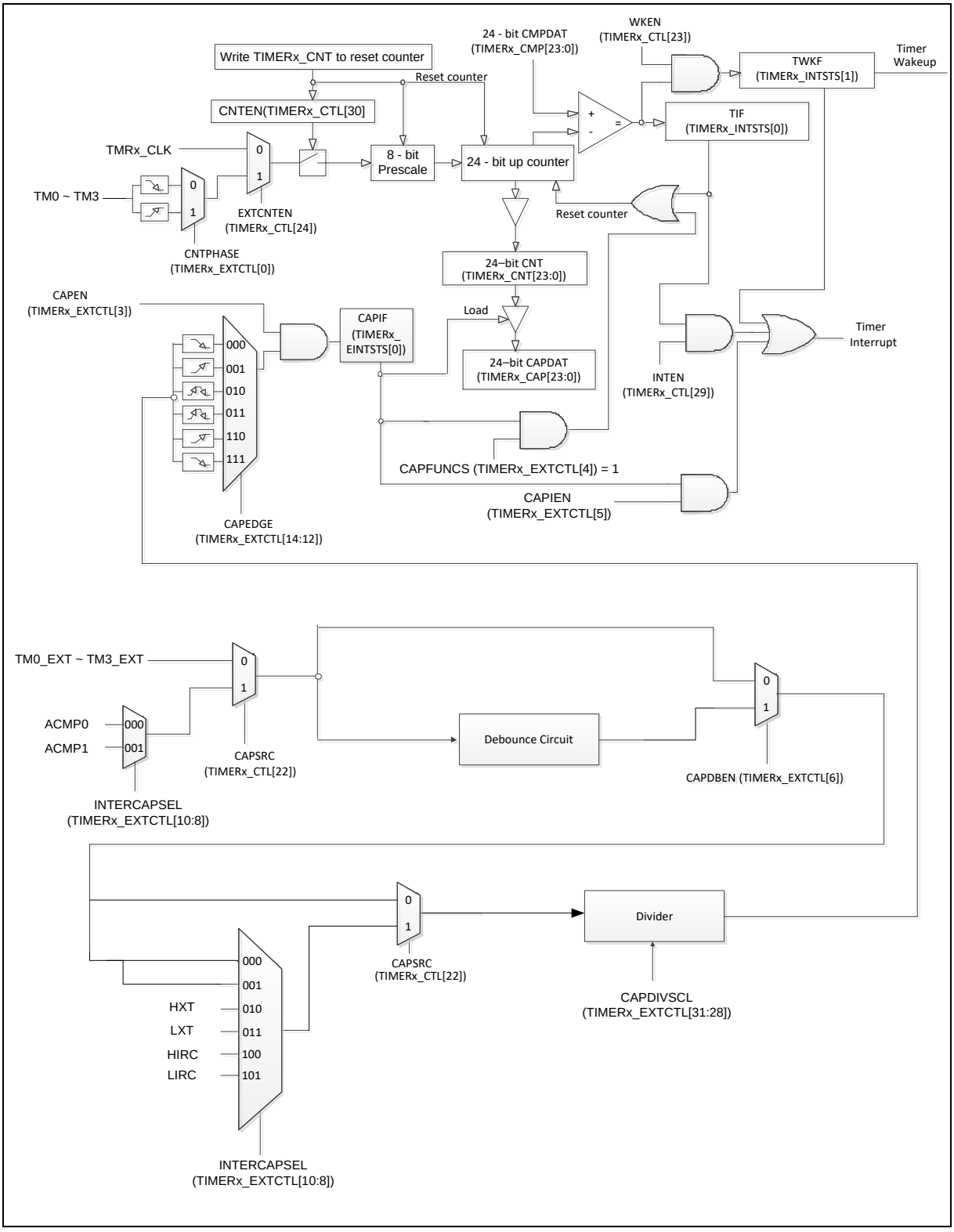


Figure 6.8-1 Timer Controller Block Diagram

Set FUNCSEL (TIMERx_CTL[15]) 0 to enable timer mode. The clock source of Timer0 ~ Timer3 in timer mode can be enabled in TMRxCKEN (CLK_APBCLK0[5:2]) and selected as different frequency in TMR0SEL (CLK_CLKSEL1[10:8]) for Timer0, TMR1SEL (CLK_CLKSEL1[14:12]) for Timer1, TMR2SEL (CLK_CLKSEL1[18:16]) for Timer2 and TMR3SEL (CLK_CLKSEL1[22:20]) for Timer3 as Figure 6.8-2.

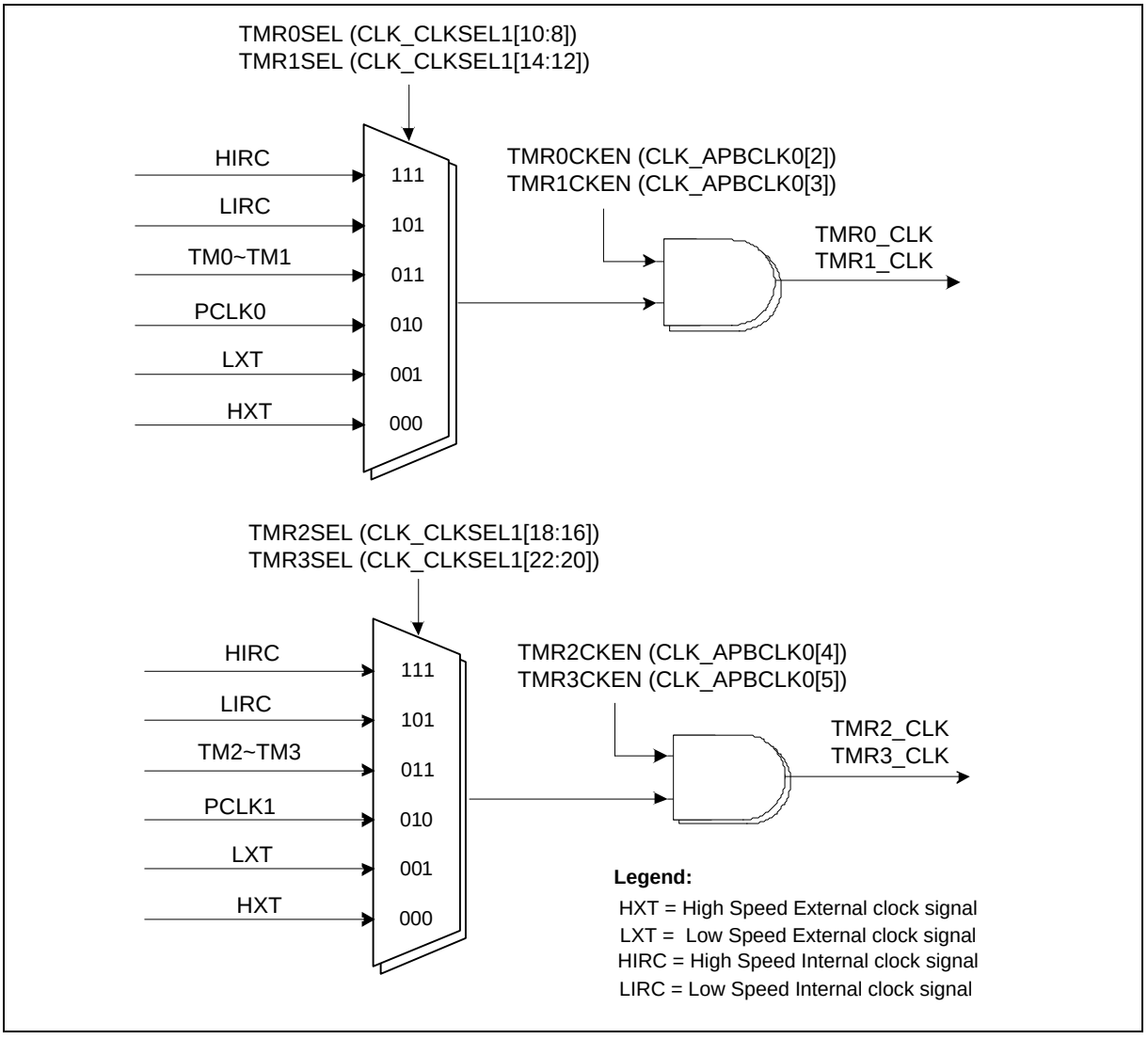


Figure 6.8-2 Clock Source of Timer Controller

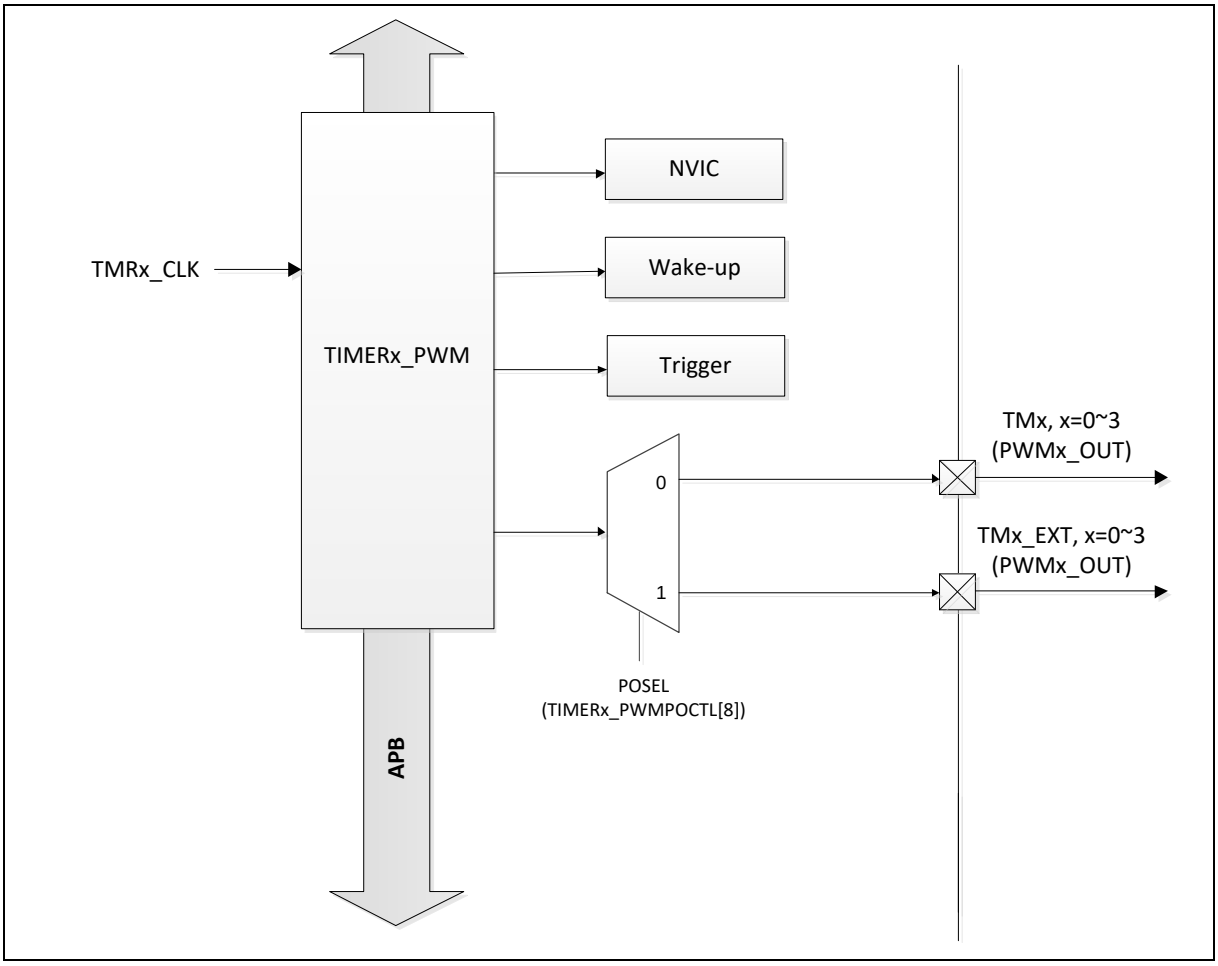


Figure 6.8-3 PWM Generator Overview Block Diagram

Set FUNCSEL (TIMERx_CTL[15]) 1 to enable PWM mode. The clock source of Timer0 ~ Timer3 in PWM mode can be enabled in TMRxCKEN (CLK_APBCLK0[5:2]). PWM system clock and counter clock source are from TMRx_CLK.

Figure 6.8-4 illustrates the PWM architecture that supports one PWM output and two seletable TMx or TMx_EXT output channels in each PWM generator.

When PWM counter (TIMERx_PWMCNT[15:0]) is equal to CMP (TIMERx_PWMCMPDAT[15:0]), relative events will be generated. These events are passed to corresponding generators to generate PWM pulse (Pulse Generator), interrupt signal (Interrupt Generator), wake-up (Wake-up Generator) and trigger signal (Trigger Generator) for EADC, PDMA and DAC to start conversion. Output Control block is used to decide PWM pulse output.

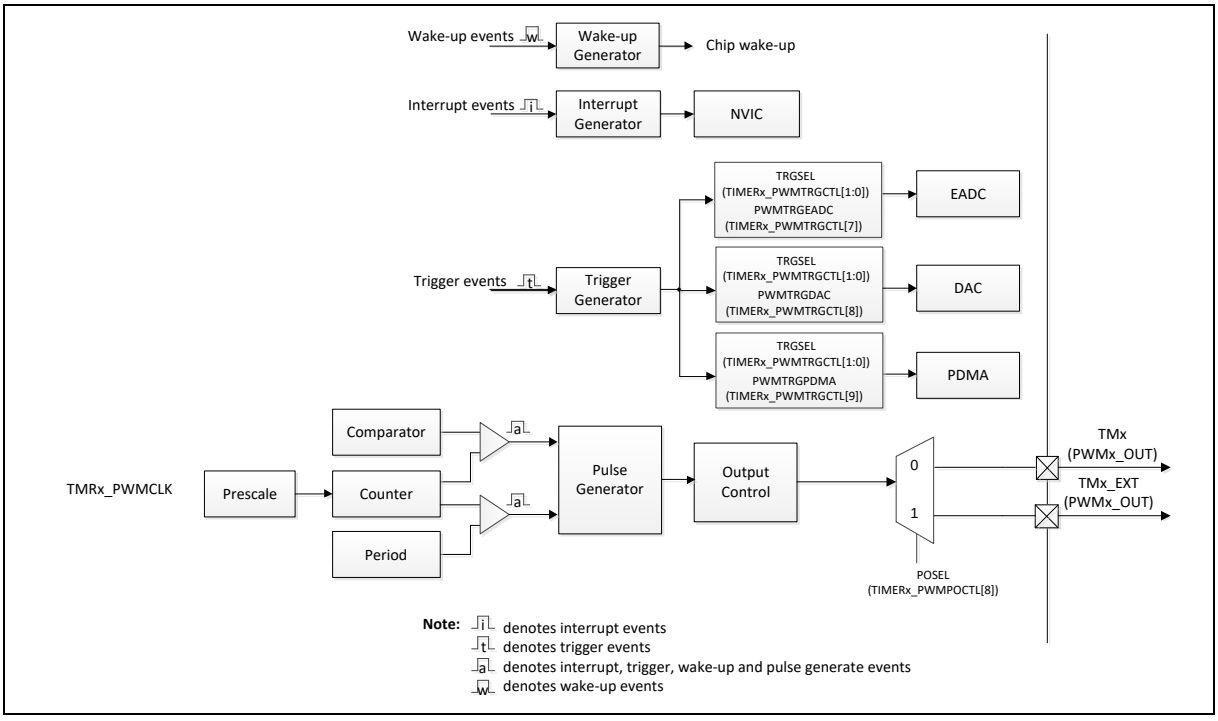


Figure 6.8-4 PWM Architecture Diagram

6.8.5 Basic Configuration

Set FUNCSEL (TIMERx_CTL[15]) 0 to enable timer mode. The clock source of Timer0 ~ Timer3 in timer mode can be enabled in TMRxCKEN (CLK_APBCLK0[5:2]) and selected as different frequency in TMR0SEL (CLK_CLKSEL1[10:8]) for Timer0, TMR1SEL (CLK_CLKSEL1[14:12]) for Timer1, TMR2SEL (CLK_CLKSEL1[18:16]) for Timer2 and TMR3SEL (CLK_CLKSEL1[22:20]) for Timer3.

Set FUNCSEL (TIMERx_CTL[15]) 1 to enable PWM mode. The clock source of Timer0 ~ Timer3 in PWM mode can be enabled in TMRxCKEN (CLK_APBCLK0[5:2]). PWM system clock and counter clock source are from TMRx_CLK.

6.8.5.1 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]).

6.8.5.2 TIMER23 basic configurations

- Clock source configuration
  - Enable TIMER2 peripheral clock in TMR2CKEN (CLK_APBCLK0[4]).
  - Enable TIMER3 peripheral clock in TMR3CKEN (CLK_APBCLK0[5]).
- Reset configuration
  - Reset TIMER2 controller in TMR2RST (SYS_IPRST1[4]).
  - Reset TIMER3 controller in TMR3RST (SYS_IPRST1[5]).

6.8.6 Timer Functional Description

6.8.6.1 Timer Interrupt Flag

The timer controller supports the followong interrupt flags; one is TIF (TIMERx_INTSTS[0]) and its set while timer counter value CNT (TIMERx_CNT[23:0]) matches the timer compared value CMPDAT (TIMERx_CMP[23:0]), and CAPIF (TIMERx_EINTSTS[0]) is set means when the transition on the TMx_EXT pin ,ACMP, internal clock (HIRC, LIRC) or external clock (HXT, LXT) associated CAPEDGE (TIMERx_EXTCTL[14:12]) setting. The TWKF (TIMERx_INTSTS[1]) bit indicates the interrupt wake-up flag status of timer. User can set CAPSRC (TIMERx_CTL[22]) and INTERCAPSEL (TIMERx_EXTCTL[10:8]) to select capture source. Set WKEN (TIMERx_CTL[23] to 1 can use wake-up function.

6.8.6.2 Timer Counting Mode

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

One–shot Mode

If the timer controller is configured at one-shot mode (TIMERx_CTL[28:27] is 00) and CNTEN (TIMERx_CTL[30]) is set, the timer counter starts up counting. Once the CNT (TIMERx_CNT[23:0]) value reaches CMPDAT (TIMERx_CMP[23:0]) value, the TIF (TIMERx_INTSTS[0]) will be set to 1, CNT value and CNTEN bit is cleared automatically by timer controller then timer counting operation stops. In the meantime, if the INTEN (TIMERx_CTL[29]) is enabled, the timer interrupt signal is generated and sent to NVIC to inform CPU also.

User can monitor the RSTACT (TIMERx_CNT[31]) to ensure counter reset operation active and disable ICE debug mode acknowledgement effects TIMER counting by setting ICEDEBUG (TIMERx_CTL[31]) to 1.

Periodic Mode

If the timer controller is configured at periodic mode (TIMERx_CTL[28:27] is 01) and CNTEN (TIMERx_CTL[30]) is set, the timer counter starts up counting. Once the CNT (TIMERx_CNT[23:0]) value reaches CMPDAT (TIMERx_CMP[23:0]) value, the TIF (TIMERx_INTSTS[0]) will be set to 1, CNT value will be cleared automatically by timer controller and timer counter operates counting again. In the meantime, if the INTEN (TIMERx_CTL[29]) bit is enabled, the timer interrupt signal is generated and sent to NVIC to inform CPU also. In this mode, the timer controller operates counting and compares with CMPDAT value periodically until the CNTEN bit is cleared by user.

User can set PERIOSEL (TIMERx_CTL[20]) to select Timer behavior.

Toggle-Output Mode

If the timer controller is configured at toggle-output mode (TIMERx_CTL[28:27] is 10) and CNTEN (TIMERx_CTL[30]) is set, the timer counter starts up counting. The counting operation of toggle-output mode is almost the same as periodic mode, except toggle-output mode has associated TM0 ~ TM3 or TM0_EXT ~ TM3_EXT pin to output signal while specify TIF (TIMERx_INTSTS[0]) is set. User can set TGLPINSEL (TIMERx_CTL[21]) to choose TMx or TMx_EXT as toggle-output pin. Thus, the toggle-output signal on TM0 ~ TM3 pin is high and changing back and forth with 50% duty cycle.

Continuous Counting Mode

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

For example, CMPDAT value is set as 80, first. The TIF will set to 1 when CNT value is equal to 80,

timer counter is kept counting and CNT value will not goes back to 0, it continues to count 81, 82, 83, … to (2^{24} -1), 0, 1, 2, 3, … to (2^{24} -1) again and again. Next, if user programs CMPDAT value as 200 and clears TIF, the TIF will set to 1 again when CNT value reaches to 200. At last, user programs CMPDAT as 500 and clears TIF, the TIF will set to 1 again when CNT value reaches to 500.

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

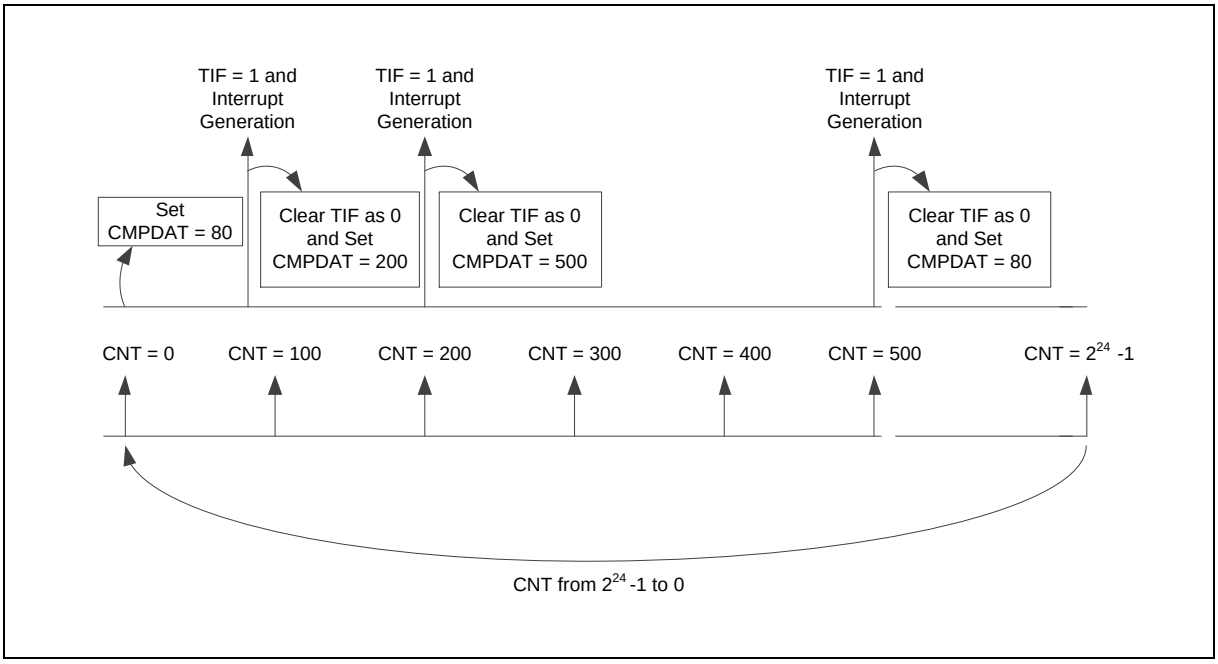


Figure 6.8-5 Continuous Counting Mode

6.8.6.3 Event Counting Mode

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

If ECNTSSEL (TIMERx_EXTCTL[16]) is 0, the event counter source is from external TMx pin. User can enable or disable TMx pin de-bounce circuit by setting CNTDBEN (TIMERx_EXTCTL[7]). The input event frequency should be less than 1/3 PCLK if TMx pin de-bounce disabled or less than 1/8 PCLK if TMx pin de-bounce enabled to assure the returned CNT value is correct, and user can also select edge detection phase of TMx pin by setting CNTPHASE (TIMERx_EXTCTL[0]) bit.

In event counting mode, the timer counting operation mode can be selected as one-shot, periodic and continuous counting mode to counts the counter value CNT (TIMERx_CNT[23:0]) for TMx. .

6.8.6.4 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, ACMP, internal clock and external clock. 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.

If CAPSRC (TIMERx_CTL[22]) is 0, the capture event is triggered by TMx_EXT pin transition. 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.

User can enable CAPIEN (TIMERx_EXTCTL[5]) to use capture interrupt fuction. When the TMx_EXT edge transition meets setting, CAPIF is high.

User 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.

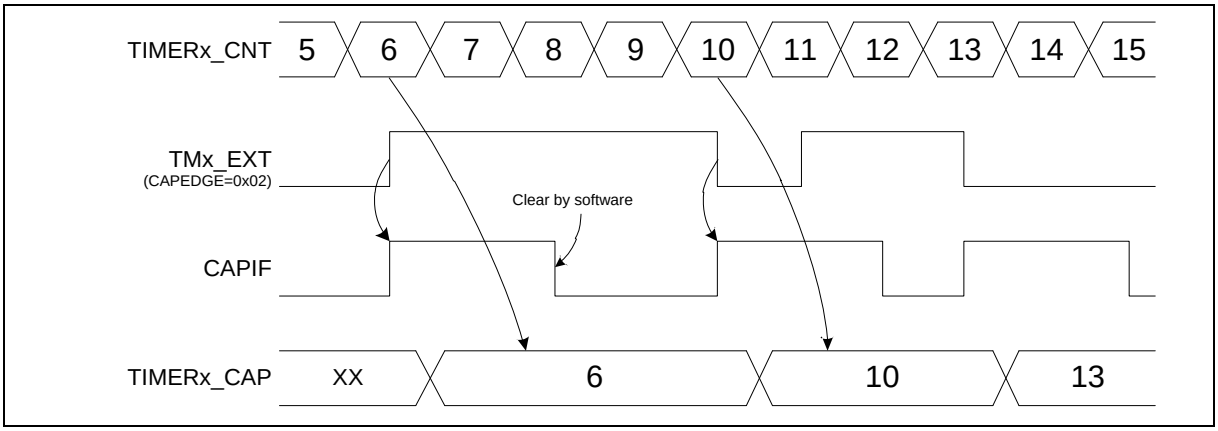


Figure 6.8-6 External Capture Mode

If CAPSRC (TIMERx_CTL[22]) is 1, set INTERCAPSEL (TIMERx_EXTCTL[10:8]) to choose different capture source. The capture event can be triggered by internal output signal transition on ACMP0 if INTERCAPSEL (TIMERx_EXTCTL[10:8]) is 000, or ACMP1 if INTERCAPSEL is 001 ; Other capture sources are HXT, LXT, LIRC, HIRC if INTERCAPSEL is 010, 011, 110, 101, 100 respectively. For example, if user wants to switch LXT or LIRC, both of capture source should be set off first.

The capture source can be divided by capture divider. Use can set CAPDIVSCL (TIMERx_EXTCTL[31:28]) to select different divider number. Be aware that timer clock frequency must be four times than capture source at least.

6.8.6.5 Reset Counter Mode

The 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 1.User must set CAPSRC and INTERCAPSEL to select TMx_EXT transition, internal ACMPx output signal and internal clock or external clock to trigger reset counter value.

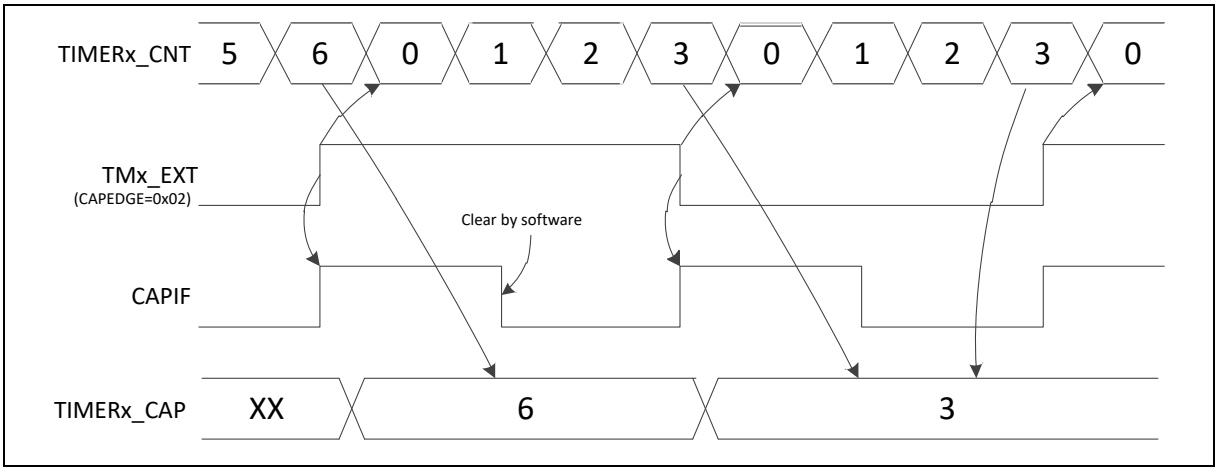


Figure 6.8-7 Reset Counter Mode

6.8.6.6 Timer Trigger Function

The timer controller provides timer time-out interrupt or capture interrupt to trigger EPWM, EADC, DAC and PDMA. If TRGSSEL (TIMERx_TRGCTL[0]) is 0, time-out interrupt signal is used to trigger EPWM, EADC, DAC and PDMA. If TRGSSEL (TIMERx_TRGCTL[0]) is 1, capture interrupt signal is used to trigger EPWM, EADC, DAC 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 EPWM external clock source.

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

When the TRGDAC (TIMERx_TRGCTL[3]) is set, if the timer interrupt signal is generated, the timer controller will trigger DAC to start converter.

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

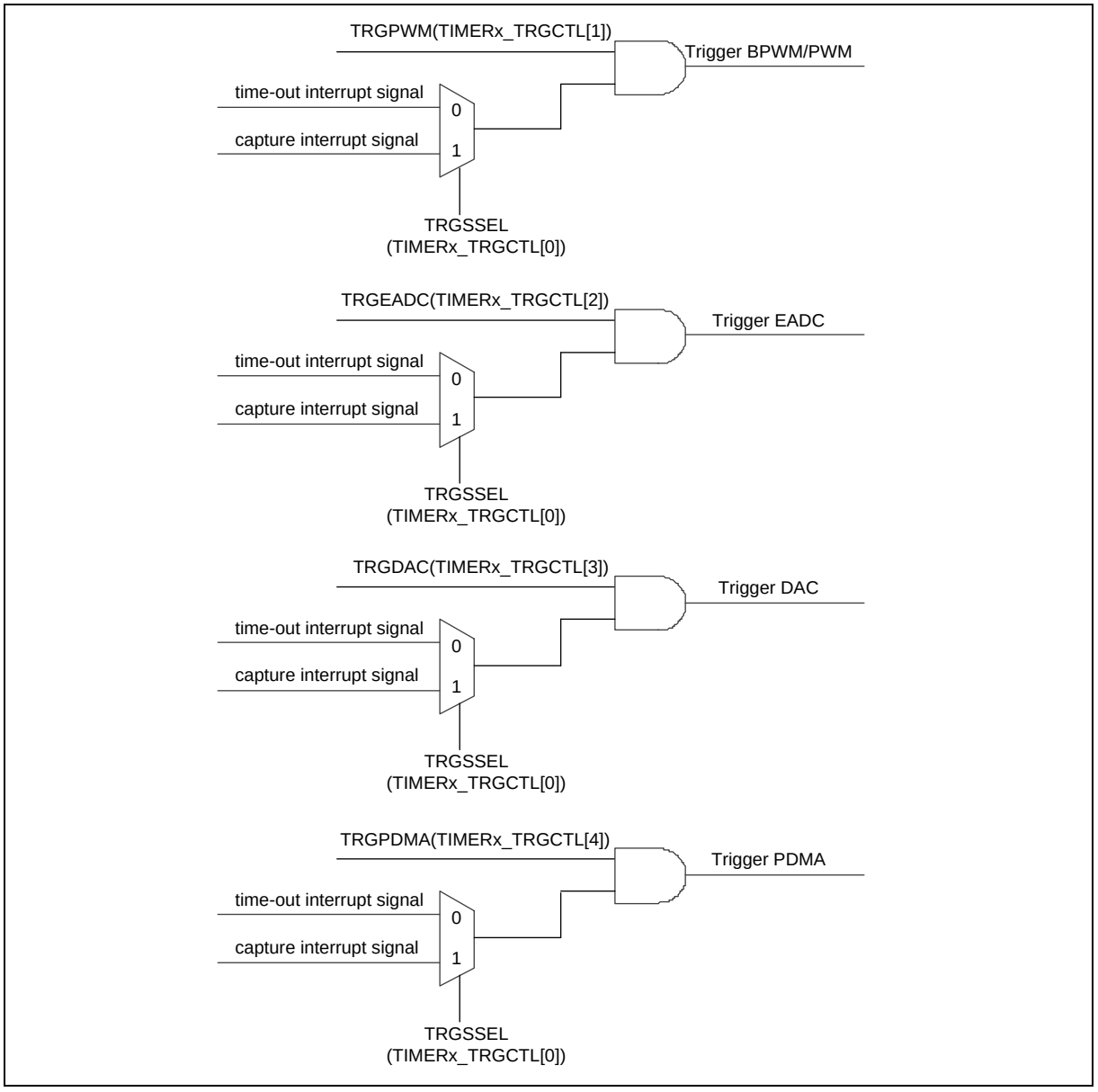


Figure 6.8-8 Internal Timer Trigger

When both the TRGPDMA (TIMERx_TRGCTL[4]) and TRGSSEL (TIMERx_TRGCTL[0]) are 1, TIMERx_CAP will change when encountering timer capture edge no matter whether CPU clear CAPIF status or not. In Trigger EADC DAC or EPWM function, user must consider that the Timer will keep register TIMERx_CAP unchanged if the CPU does not clear the CAPIF status.

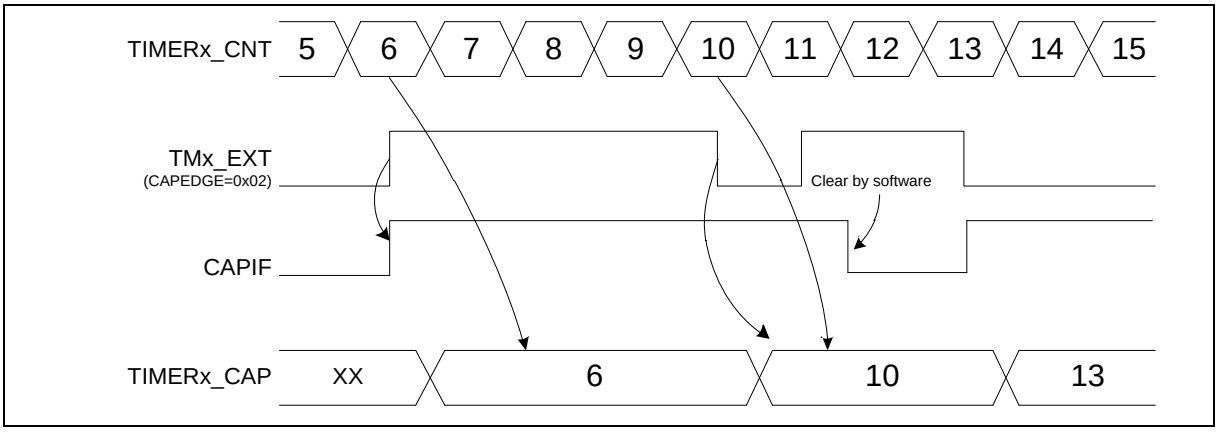


Figure 6.8-9 Capture interrupt trigger PDMA

6.8.6.7 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 Capture 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(TIMERx_CMP[23:0]) 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 6.8-10 shows the sample flow of Inter-Timer Trigger Capture Mode for Timer0 as event counting mode and Timer1 as trigger-counting capture mode.

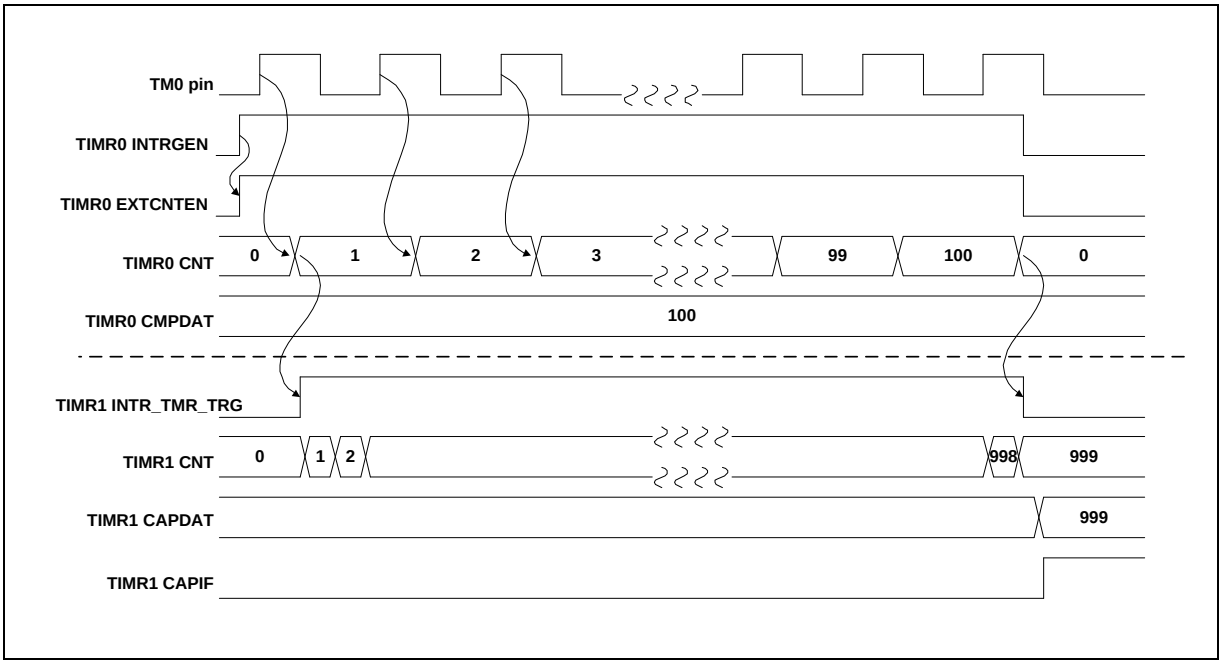


Figure 6.8-10 Inter-Timer Trigger Capture Timing

User must clear Timer1/3 CAPIF if user wants to use inter-timer trigger function again.

6.8.7 PWM Functional Description

6.8.7.1 PWM Block Diagram

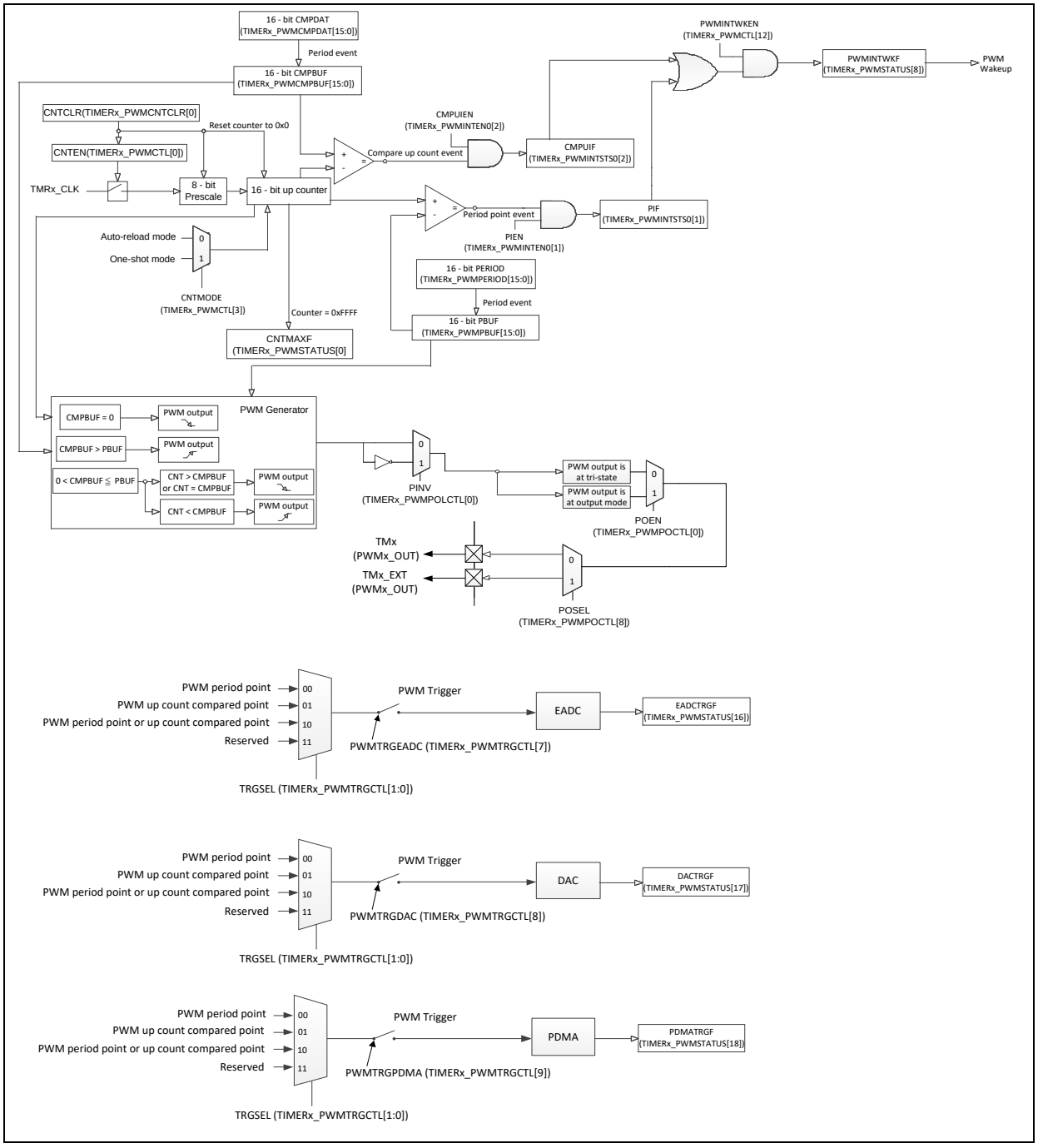


Figure 6.8-11 Timer PWM Block Diagram

6.8.7.2 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[7:0]). Figure 6.8-12 shows an example of PWM prescale waveform in up count type.

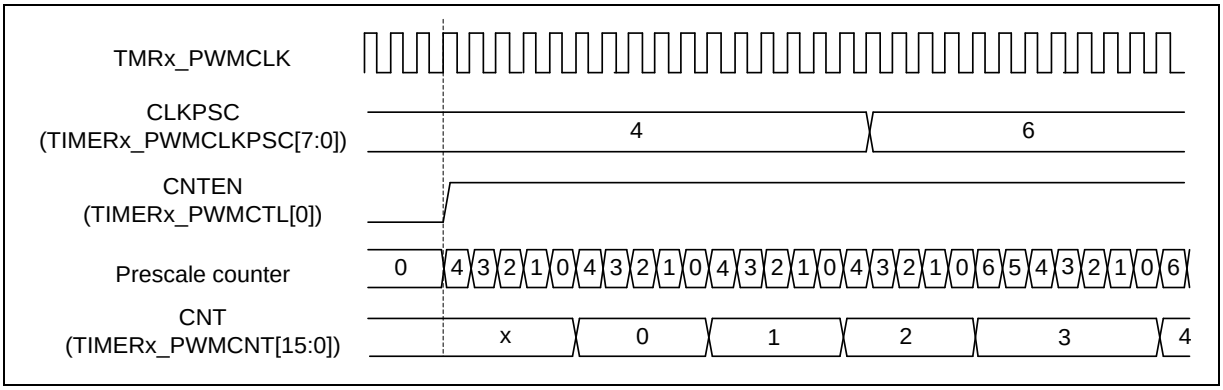


Figure 6.8-12 PWM Prescale Waveform in Up Count Type

6.8.7.3 PWM Counter

The PWM supports up count type

It starts up-counting from 0 to PERIOD (TIMERx_PWMPERIOD[15:0]). The current counter value can be read from the CNT (TIMERx_PWMCNT[15:0]). PWM generates a period point event when the counter counts to PERIOD and prescale counts to 0. Figure 6.8-13 shows an example of PWM up count type, where PWM period time is (PERIOD+1) * (CLKPSC+1) * TMRx_PWMCLK.

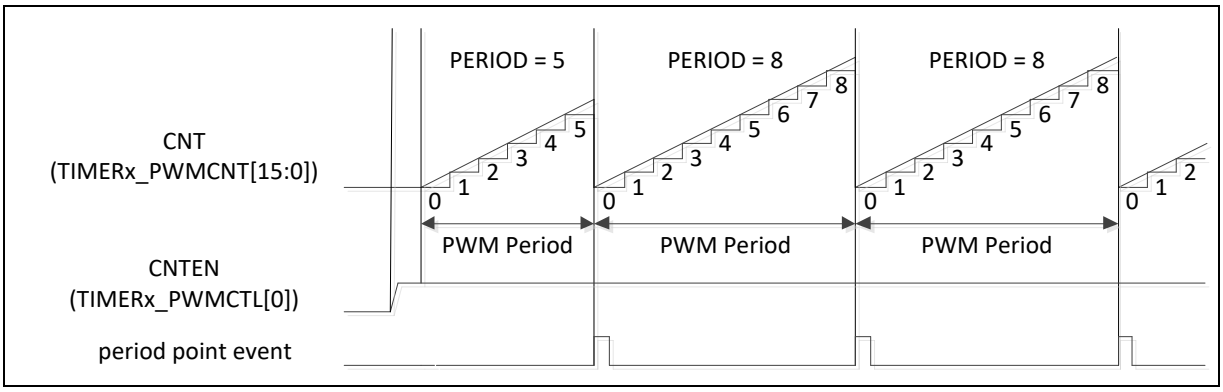


Figure 6.8-13 PWM Up Count Type

6.8.7.4 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 0 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.

For example, user writes 0x10 to CMP, writes 0x3F to PERIOD and set CMPUIEN (TIMERx_PWMINTEN0[2]) and PIEN (TIMERx_PWMINTEN0[1]) to 1. When counter (CNT) counts to 0x10, CMPUIF (TIMERx_PWMINTSTS0[2]) is set to 1. Counter keeps counting until 0x3F, PIF (TIMERx_PWMINTSTS0[1]) is set to 1 then counter reload to 0, and counter stops counting.

Another example is user writes 0x10 to CMP, writes 0x3F to PERIOD and set CMPUIEN and PIEN to 1. If user writes 0x25 to CMP when counter counting, this value will load to CMPBUF (TIMERx_PWMCMPBUF[15:0]) when counter counts to 0x3F and PIF is set to 1, then counter counts

from 0. If user clears CMPUIF and PIF before counter counts to 0x25, CMPUIF and PIF will be set to 1 when counter counts to 0x25 and 0x3F, respectively.

In auto-reload mode, PWM counter is continuous running with current active PERIOD and CMP. If user sets PERIOD to zero in auto-reload mode, PWM counter value will reload to 0 after one PWM period is completed.

For example, user writes 0x10 to CMP, writes 0x3F to PERIOD. CMPUIEN and PIEN are set to 1. When counter counts to 0x10, CMPUIF is set to 1. Counter keeps counting until 0x3F, PIF is set to 1, and counter will restart counting from 0. If user clear CMPUIF and PIF before counter counts to 0x10 (2nd counts), CMPUIF will equal to 1 and PIF will equal to 1 when counter counts to 0x10 and 0x3F, respectively.

When user writes 0 to PERIOD and this value will be loaded to PBUF (TIMERx_PWMPBUF[15:0]). This action will make counter be set to 0 and stops counting until user rewrites another value (except 0) to PERIOD and this value loads to PBUF, counter counts from 0.

The one-shot and auto-reload mode comparsion is shown as Table 6.8-1.

Counter Mode Functions	One-Shot	Auto-Reload
CNTMODE (TIMERx_PWMCTL[3])	1	0
CNT = PERIOD (PERIOD is not 0)	Stops counting	Keeps counting from 0 to PERIOD (Periodic)
PERIOD = 0 and loads to PBUF, then write another value (Except 0) to PERIOD	CNT stops counting even though user writes new PERIOD	CNT = 0 (reset value), CNT stops counting until PERIOD and PBUF is not equal to 0

Table 6.8-1 One-shot and Auto-reload Functions

Note: PERIOD (TIMERx_PWMPERIOD[15:0]) and CMP (TIMERx_PWMCMPDAT[15:0]) will load to PBUF (TIMERx_PWMPBUF[15:0]) and CMPBUF (TIMERx_PWMCMPBUF[15:0]), respectively, when one PWM period completes. The first CMP and PERIOD will load to CMPBUF and PBUF, respectively, after CNTEN (TIMERx_PWMCTL[0]) is 1.

6.8.7.5 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 wake-up or trigger EADC, PDMA, DAC to start conversion.

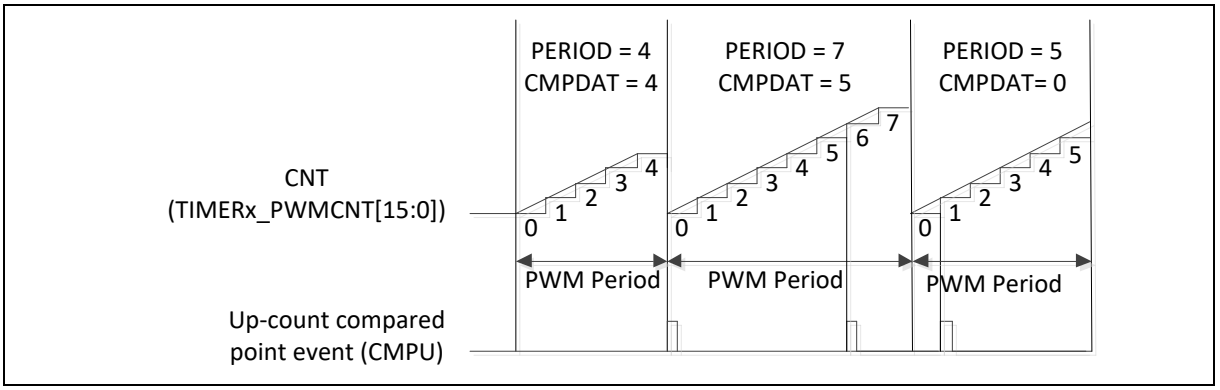


Figure 6.8-14 PWM Comparator Events in Up Count Type

6.8.7.6 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 6.8-15 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 6.8-15.

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.

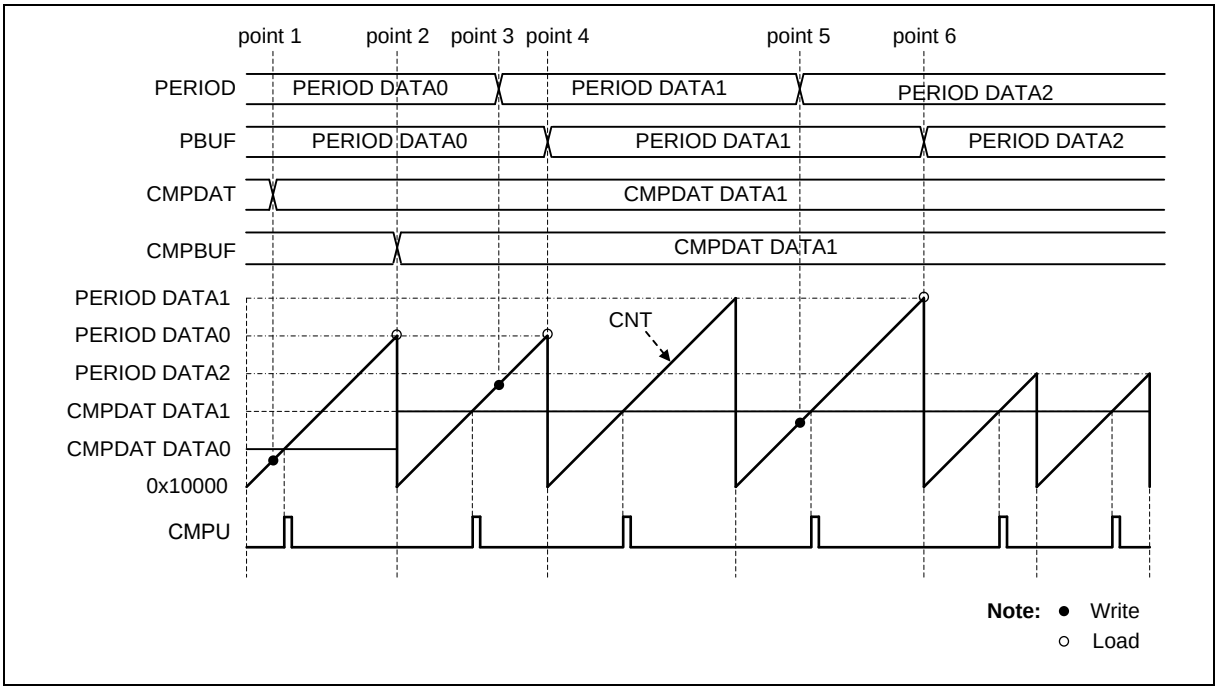


Figure 6.8-15 Period Loading Mode with Up Count Type

6.8.7.7 PWM Pulse Generator

PWM pulse generator uses counter and comparator events to generate PWM output pulse. The events are zero point in up count type and counter equal to comparator point in up count type.

The event point can generate PWM output waveform in up count type as shown in Figure 6.8-16

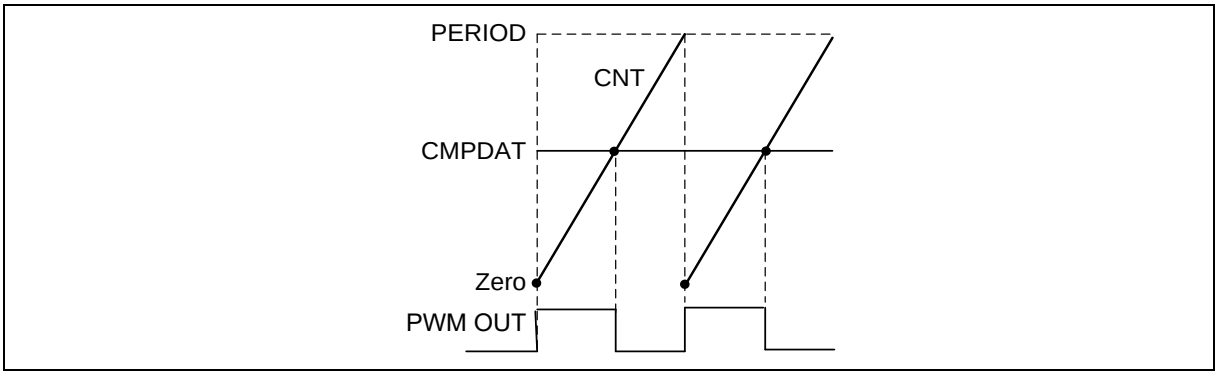


Figure 6.8-16 PWM Pulse Generation in Up Count Type

The PWM output pulse is associated with CMPBUF (TIMERx_PWMCMPBUF[15:0]) and PBUF (TIMERx_PWMPBUF[15:0]). The rules are listed as below.

1. If CMPBUF is zero, PWM output is 100% low.
2. If CMPBUF > PBUF, PWM output is 100% high.
3. If 0 < CMPBUF ≦ PBUF, PWM output high/low duty is according to counter value CNT(TIMERx_PWMCNT[15:0]).
   1) If PWM counter is higher than CMPBUF or equal to CMPBUF, PWM output is low.
   2) If PWM counter is lower than CMPBUF, PWM output is high.

The PWM output level table is shown asTable 6.8-2.

Conditions	**CMPBUF = 0**	**CMPBUF > PBUF**	**0 < CMPBUF ≦ PBUF**	
			CNT ≧ CMPBUF	CNT < CMPBUF
PWM Output	Low	High	Low	High

Table 6.8-2 PWM Output Level

Figure 6.8-17 is an example about PWM duty cycle from 0% to 100% in up count type where PERIOD is 4 with different CMP value.

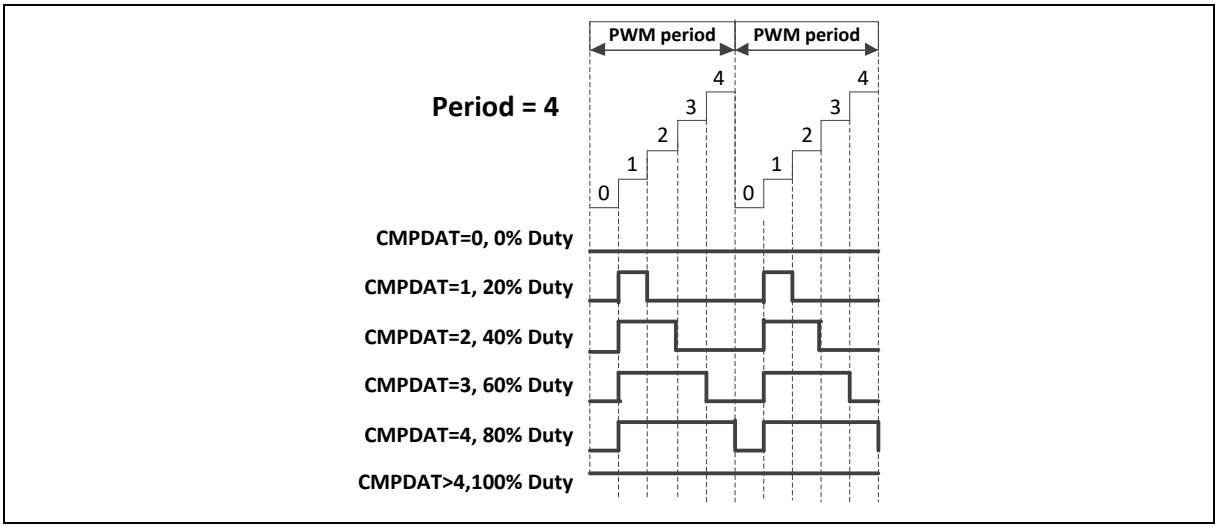


Figure 6.8-17 PWM 0% to 100% Duty Cycle in Up Count Type

6.8.7.8 PWM Output Control

After PWM pulse generator, there are three steps to control output waveform. User can set POEN (TIMERx_PWMPOCTL[0]) 1 to enable PWMx_OUT output waveform and set POSEL (TIMERx_PWMPOCTL[8]) to select TMx or TMx_EXT as PWM output channel.

There are polarity control, output enable control and output channel select to control output waveform as shown in Figure 6.8-18.

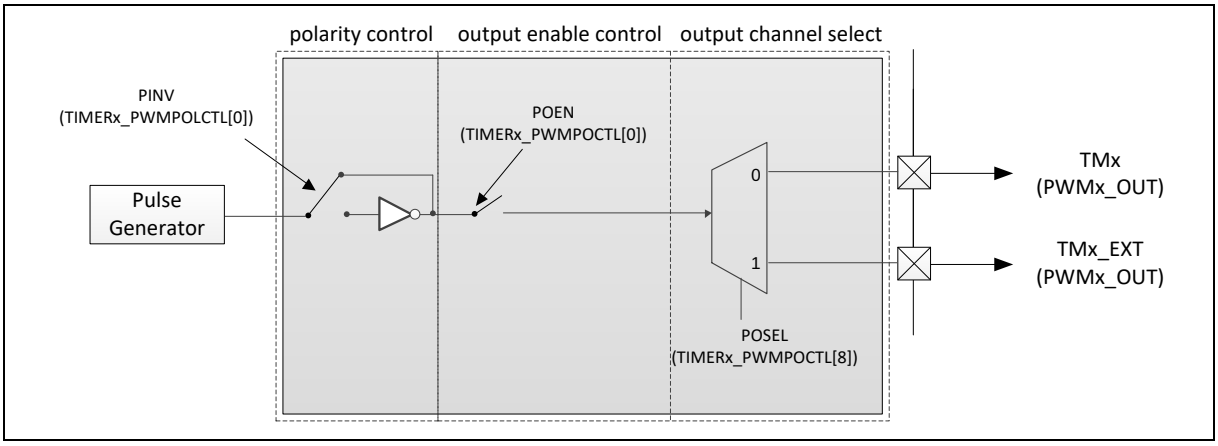


Figure 6.8-18 PWM Output PWMx_OUT Control

6.8.7.9 Polarity Control

Each PWMx_OUT (TMx or TMx_EXT) has an independent polarity control to configure the polarity of the active state of PWM output. User can control polarity state on PINV (TIMERx_PWMPOLCTL[0]). Figure 6.8-19 shows the PWMx_OUT with polarity control.

When PWM output is selected as TMx pin, PINV is 0, TMx_EXT pin is kept as 0. Similarly, TMx_EXT pin is kept as 1 when PINV is 1.

When PWM output is selected as TMx_EXT pin, PINV is 0, TMx pin is kept as 0. Similarly, TMx pin is kept as 1 when PINV is 1.

When PWM output is selected as TMx pin, PINV is 0, but POEN (TIMERx_PWMPOCTL[0]) is set as 0, TMx pin is kept as 0. Similarly, TMx pin is kept as 1 when PINV is 1.

When PWM output is selected as TMx_EXT pin, PINV is 0, but POEN is set as 0, TMx_EXT pin is kept as 0. Similarly, TMx_EXT pin is kept as 1 when PINV is 1.

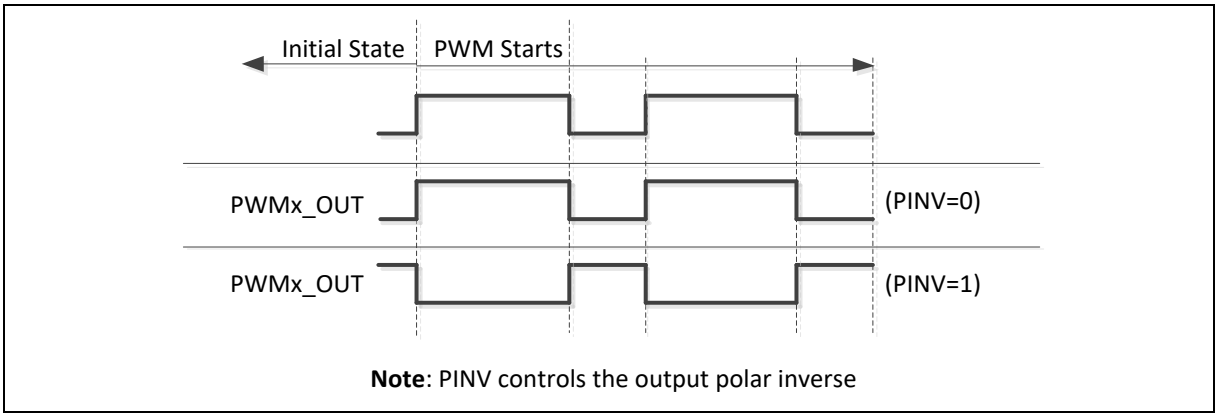


Figure 6.8-19 PWMx_OUT Polarity Control

6.8.7.10 PWM Interrupt Generator

There are independent interrupts for each PWM as shown in Figure 6.8-20.

The PWM interrupt (PWMx_INT) comes from PWM complementary pair events. The counter can generate the period point interrupt flag PIF (TIMERx_PWMINTSTS0[1]). When counter equals to the comparator value stored in CMP (TIMERx_PWMCMPDAT[15:0]) at up-count direction, the comparator up interrupt flag CMPUIF (TIMERx_PWMINTSTS0[2]) is set. If the corresponding interrupt enable bits are set, the interrupt trigger events will also generates interrupt signals.

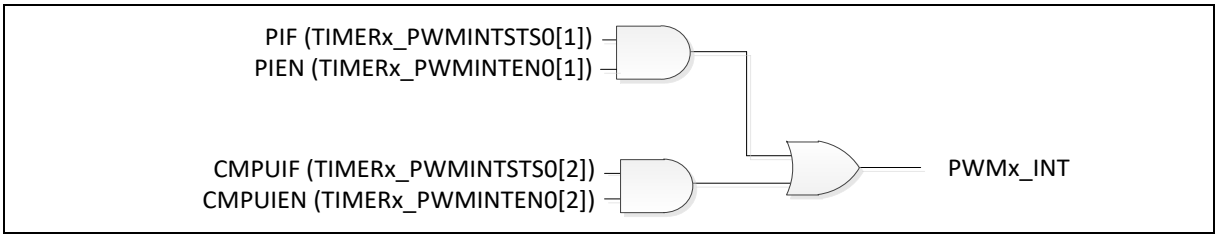


Figure 6.8-20 PWM Interrupt Architecture Diagram

6.8.7.11 PWM Wake-up Generator

User can sets PWMINTWKEN (TIMERx_PWMCTL[12]) high to wake up when PWM interrupt occurs. Before enter power-down mode, user must select LXT or LIRC as PWM clock source.

When PWM wake-up occurs, PWMINTWKF (TIMERx_PWMSTATUS[8]) is set to 1. User can write 1 to this bit to clear this flag.

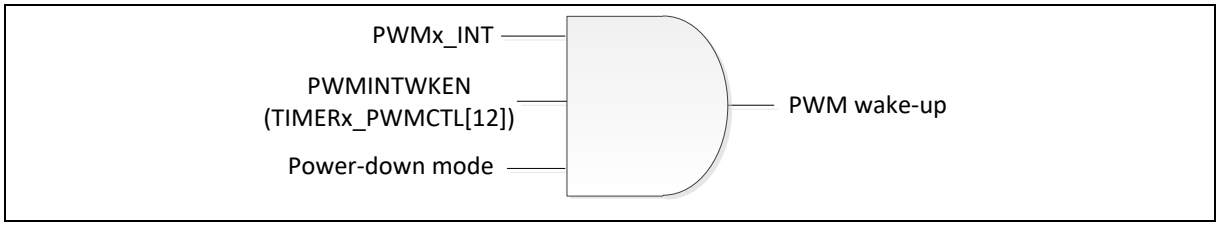


Figure 6.8-21 PWM Wake-up Architecture Diagram

6.8.7.12 PWM Trigger EADC, PDMA, DAC Generator

PWM counter event can be one of the EADC, PDMA, DAC conversion trigger source. User sets TRGSEL (TIMERx_PWMTRGCTL[1:0]) to select which PWM counter event can trigger conversion.

When the PWMTRGEADC (TIMERx_PWMTRGCTL[7]) is set to 1, the PWM can trigger EADC. When the PWMTRGDAC (TIMERx_PWMTRGCTL[8]) is set to 1, the PWM can trigger DAC. When the PWMTRGPDMA (TIMERx_PWMTRGCTL[9]) is set to 1, the PWM can trigger PDMA.

There are three PWM counter events can be selected as the trigger source to start conversion as shown in Figure 6.8-22.

The PWM interrupt, wake-up and trigger comparison table is shown as Table 6.8-3.

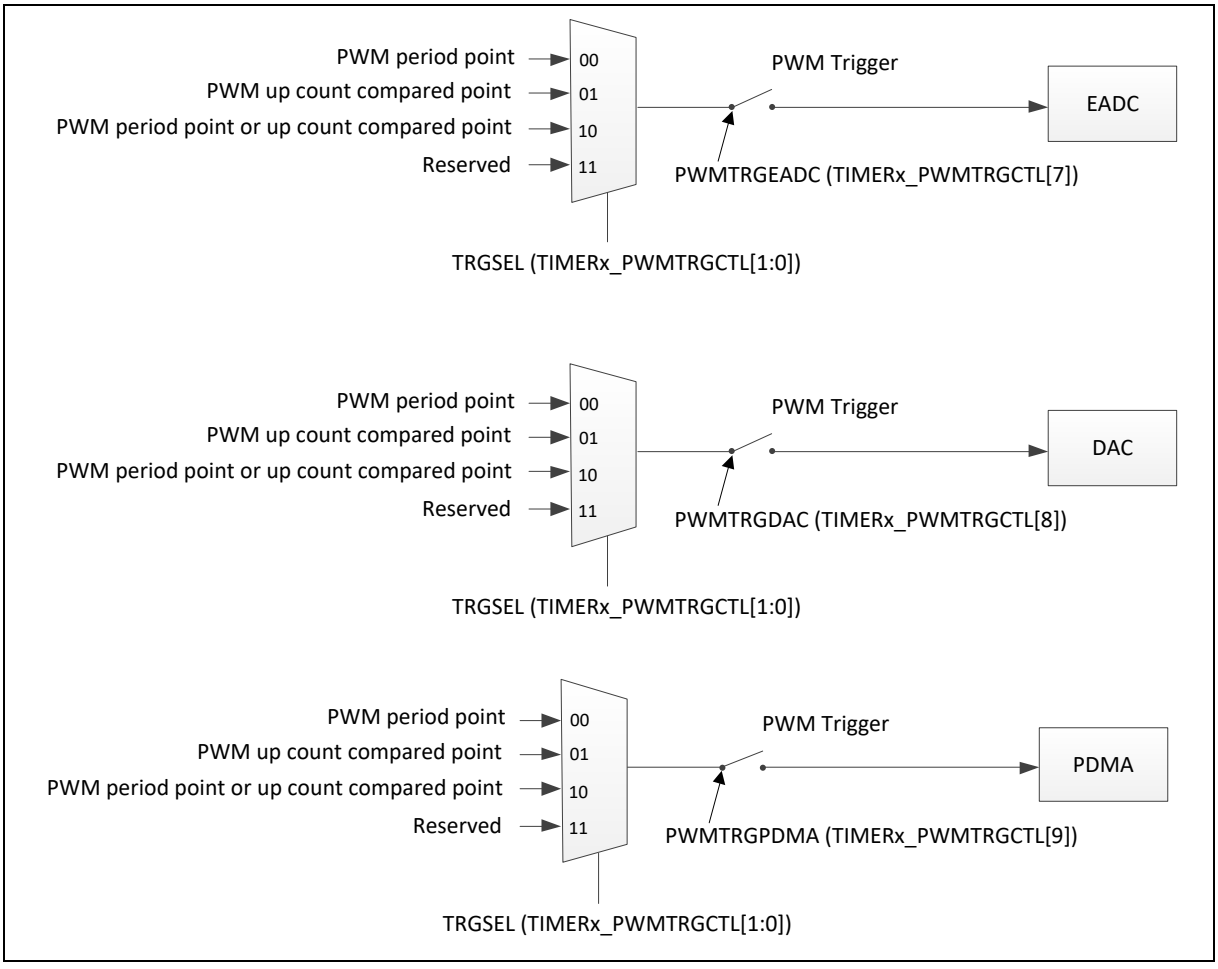


Figure 6.8-22 PWM Trigger Block Diagram

Functions / Events		Interrupt	Wake-Up PWMINTWKEN (TIMERx_PWMCTL[12])		Trigger
			= 0	= 1	
CNT = CMP	CMPUIEN = 1	Yes	No	Yes	EADC, DAC, PDMA (Note1)
	CMPUIEN = 0	No	No	No	No
CNT = PERIOD	PIEN = 1	Yes	No	Yes	EADC, DAC, PDMA (Note1)
	PIEN = 0	No	No	No	No

Table 6.8-3 PWM Interrupt, Wake-up and Trigger Events Comparison

Note: Set TRGSEL (TIMERx_PWMTRGCTL[1:0]) to select trigger event and sets PWMTRGEADC (TIMERx_PWMTRGCTL[7]), PWMTRGDAC (TIMERx_PWMTRGCTL[8]), PWMTRGPDMA (TIMERx_PWMTRGCTL[9]) to select which device will be triggered.

6.8.7.13 Mode Transition

If user needs to change timer mode to PWM mode or PWM mode to timer mode, user must do reset before change mode or check the previous mode is end.

The Timer mode to PWM mode transition steps are shown below.

1. Set Timer CNTEN (TIMERx_CTL[30]) as 0
2. Polling ACTSTS (TIMERx_CTL[25]). If ACTSTS is 0, user can set FUNCSEL (TIMERx_CTL[15]) as 1 to enable PWM mode.
3. PWM function settings.

The PWM mode to Timer mode transition steps are shown below.

1. Set PWM CNTEN (TIMERx_PWMCTL[0]) as 0.
2. Set FUNCSEL as 0 to disable PWM mode.
3. Polling FUNCSEL. If FUNCSEL is 0, user can set Timer settings.

6.8.8 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_PWMCTL	TMR01_BA+0x40	R/W	Timer0 PWM Control Register	0x0000_0000
TIMER0_PWMCLKPSC	TMR01_BA+0x44	R/W	Timer0 PWM Counter Clock Pre-scale Register	0x0000_0000
TIMER0_PWMCNTCLR	TMR01_BA+0x48	R/W	Timer0 PWM Clear Counter Register	0x0000_0000
TIMER0_PWMPERIOD	TMR01_BA+0x4C	R/W	Timer0 PWM Period Register	0x0000_0000
TIMER0_PWMCMPDAT	TMR01_BA+0x50	R/W	Timer0 PWM Comparator Register	0x0000_0000
TIMER0_PWMCNT	TMR01_BA+0x54	R	Timer0 PWM Counter Register	0x0000_0000
TIMER0_PWMPOLCTL	TMR01_BA+0x58	R/W	Timer0 PWM Pin Output Polar Control Register	0x0000_0000
TIMER0_PWMPOCTL	TMR01_BA+0x5C	R/W	Timer0 PWM Pin Output Control Register	0x0000_0000
TIMER0_PWMINTEN0	TMR01_BA+0x60	R/W	Timer0 PWM Interrupt Enable Register 0	0x0000_0000
TIMER0_PWMINTSTS0	TMR01_BA+0x64	R/W	Timer0 PWM Interrupt Status Register 0	0x0000_0000
TIMER0_PWMTRGCTL	TMR01_BA+0x68	R/W	Timer0 PWM Trigger Control Register	0x0000_0000
TIMER0_PWMSTATUS	TMR01_BA+0x6C	R/W	Timer0 PWM Status Register	0x0000_0000
TIMER0_PWMPBUF	TMR01_BA+0x70	R	Timer0 PWM Period Buffer Register	0x0000_0000
TIMER0_PWMCMPBUF	TMR01_BA+0x74	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_PWMCTL	TMR01_BA+0x140	R/W	Timer1 PWM Control Register	0x0000_0000
TIMER1_PWMCLKPSC	TMR01_BA+0x144	R/W	Timer1 PWM Counter Clock Pre-scale Register	0x0000_0000
TIMER1_PWMCNTCLR	TMR01_BA+0x148	R/W	Timer1 PWM Clear Counter Register	0x0000_0000
TIMER1_PWMPERIOD	TMR01_BA+0x14C	R/W	Timer1 PWM Period Register	0x0000_0000
TIMER1_PWMCMPDAT	TMR01_BA+0x150	R/W	Timer1 PWM Comparator Register	0x0000_0000
TIMER1_PWMCNT	TMR01_BA+0x154	R	Timer1 PWM Counter Register	0x0000_0000
TIMER1_PWMPOLCTL	TMR01_BA+0x158	R/W	Timer1 PWM Pin Output Polar Control Register	0x0000_0000
TIMER1_PWMPOCTL	TMR01_BA+0x15C	R/W	Timer1 PWM Pin Output Control Register	0x0000_0000
TIMER1_PWMINTEN0	TMR01_BA+0x160	R/W	Timer1 PWM Interrupt Enable Register 0	0x0000_0000
TIMER1_PWMINTSTS0	TMR01_BA+0x164	R/W	Timer1 PWM Interrupt Status Register 0	0x0000_0000
TIMER1_PWMTRGCTL	TMR01_BA+0x168	R/W	Timer1 PWM Trigger Control Register	0x0000_0000
TIMER1_PWMSTATUS	TMR01_BA+0x16C	R/W	Timer1 PWM Status Register	0x0000_0000
TIMER1_PWMPBUF	TMR01_BA+0x170	R	Timer1 PWM Period Buffer Register	0x0000_0000
TIMER1_PWMCMPBUF	TMR01_BA+0x174	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_PWMCTL	TMR23_BA+0x40	R/W	Timer2 PWM Control Register	0x0000_0000
TIMER2_PWMCLKPSC	TMR23_BA+0x44	R/W	Timer2 PWM Counter Clock Pre-scale Register	0x0000_0000
TIMER2_PWMCNTCLR	TMR23_BA+0x48	R/W	Timer2 PWM Clear Counter Register	0x0000_0000
TIMER2_PWMPERIOD	TMR23_BA+0x4C	R/W	Timer2 PWM Period Register	0x0000_0000
TIMER2_PWMCMPDAT	TMR23_BA+0x50	R/W	Timer2 PWM Comparator Register	0x0000_0000

TIMER2_PWMCNT	TMR23_BA+0x54	R	Timer2 PWM Counter Register	0x0000_0000
TIMER2_PWMPOLCTL	TMR23_BA+0x58	R/W	Timer2 PWM Pin Output Polar Control Register	0x0000_0000
TIMER2_PWMPOCTL	TMR23_BA+0x5C	R/W	Timer2 PWM Pin Output Control Register	0x0000_0000
TIMER2_PWMINTEN0	TMR23_BA+0x60	R/W	Timer2 PWM Interrupt Enable Register 0	0x0000_0000
TIMER2_PWMINTSTS0	TMR23_BA+0x64	R/W	Timer2 PWM Interrupt Status Register 0	0x0000_0000
TIMER2_PWMTRGCTL	TMR23_BA+0x68	R/W	Timer2 PWM Trigger Control Register	0x0000_0000
TIMER2_PWMSTATUS	TMR23_BA+0x6C	R/W	Timer2 PWM Status Register	0x0000_0000
TIMER2_PWMPBUF	TMR23_BA+0x70	R	Timer2 PWM Period Buffer Register	0x0000_0000
TIMER2_PWMCMPBUF	TMR23_BA+0x74	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_PWMCTL	TMR23_BA+0x140	R/W	Timer3 PWM Control Register	0x0000_0000
TIMER3_PWMCLKPSC	TMR23_BA+0x144	R/W	Timer3 PWM Counter Clock Pre-scale Register	0x0000_0000
TIMER3_PWMCNTCLR	TMR23_BA+0x148	R/W	Timer3 PWM Clear Counter Register	0x0000_0000
TIMER3_PWMPERIOD	TMR23_BA+0x14C	R/W	Timer3 PWM Period Register	0x0000_0000
TIMER3_PWMCMPDAT	TMR23_BA+0x150	R/W	Timer3 PWM Comparator Register	0x0000_0000
TIMER3_PWMCNT	TMR23_BA+0x154	R	Timer3 PWM Counter Register	0x0000_0000
TIMER3_PWMPOLCTL	TMR23_BA+0x158	R/W	Timer3 PWM Pin Output Polar Control Register	0x0000_0000
TIMER3_PWMPOCTL	TMR23_BA+0x15C	R/W	Timer3 PWM Pin Output Control Register	0x0000_0000
TIMER3_PWMINTEN0	TMR23_BA+0x160	R/W	Timer3 PWM Interrupt Enable Register 0	0x0000_0000
TIMER3_PWMINTSTS0	TMR23_BA+0x164	R/W	Timer3 PWM Interrupt Status Register 0	0x0000_0000
TIMER3_PWMTRGCTL	TMR23_BA+0x168	R/W	Timer3 PWM Trigger Control Register	0x0000_0000
TIMER3_PWMSTATUS	TMR23_BA+0x16C	R/W	Timer3 PWM Status Register	0x0000_0000
TIMER3_PWMPBUF	TMR23_BA+0x170	R	Timer3 PWM Period Buffer Register	0x0000_0000
TIMER3_PWMCMPBUF	TMR23_BA+0x174	R	Timer3 PWM Comparator Buffer Register	0x0000_0000

6.8.9 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	CAPSRC	TGLPINSEL	PERIOSEL	INTRGEN	Reserved		
15	14	13	12	11	10	9	8
FUNCSEL	Reserved						
7	6	5	4	3	2	1	0
PSC							

Bits	Description	
[31]	**ICEDEBUG**	**ICE Debug Mode Acknowledge Disable Bit (Write Protect)** 0 = ICE debug mode acknowledgement effects TIMER counting. TIMER counter will be held while CPU is held by ICE. 1 = ICE debug mode acknowledgement Disabled. TIMER counter will keep going no matter CPU is held by ICE or not. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[30]	**CNTEN**	**Timer Counting Enable Bit** 0 = Stops/Suspends counting. 1 = Starts counting. **Note 1:** 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. **Note 2:** This bit is auto-cleared by hardware in one-shot mode (TIMER_CTL[28:27] = 00) when the timer time-out interrupt flag TIF (TIMERx_INTSTS[0]) is generated. **Note 3:** 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.
[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]	**CAPSRC**	**Capture Pin Source Selection** 0 = Capture Function source is from TMx_EXT (x= 0~3) pin. 1 = Capture Function source is from internal ACMP output signal , internal clock (LIRC, HIRC), or external clock (HXT, LXT). **Note:** When CAPSRC = 1,User can set INTERCAPSEL (TIMERx_EXTCTL[10:8]) to decide which internal ACMP output signal or which clock is as timer capture source.
[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 updates 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 Bit** 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 ineffective and the read back value is always 0.
[18:16]	**Reserved**	Reserved.
[15]	**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.
[14:8]	**Reserved**	Reserved.
[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:** Update 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.
[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). **Note 1:** Never write 0x0 or 0x1 in CMPDAT field, or the core will run into unknown state. **Note 2:** 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.
[1]	**TWKF**	**Timer Wake-up Flag** This bit indicates the interrupt wake-up flag status of timer. 0 = Timer does not cause CPU wake-up. 1 = CPU wake-up from Idle or Power-down mode if timer time-out interrupt signal generated. **Note:** This bit is cleared by writing 1 to it.
[0]	**TIF**	**Timer Interrupt Flag** This bit indicates the interrupt flag status of Timer while 24-bit timer up counter CNT (TIMERx_CNT[23:0]) value reaches to CMPDAT (TIMERx_CMP[23:0]) value. 0 = No effect. 1 = CNT value matches the CMPDAT value. **Note:** This bit is cleared by writing 1 to it.

Timer Data Register (TIMERx_CNT)

Register	Offset	R/W	Description	Reset Value
TIMER0_CNT	TMR01_BA+0x0C	R/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.
[30:24]	**Reserved**	Reserved.
[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.
[23:0]	**CAPDAT**	**Timer Capture Data Register** When CAPEN (TIMERx_EXTCTL[3]) bit is set 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. **Note:** User 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.

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
CAPDIVSCL				Reserved			
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved	CAPEDGE			Reserved	INTERCAPSEL		
7	6	5	4	3	2	1	0
CNTDBEN	CAPDBEN	CAPIEN	CAPFUNCS	CAPEN	Reserved		CNTPHASE

Bits	Description	
[31:28]	**CAPDIVSCL**	**Timer Capture Source Divider Scale** **This bits indicate the divide scale for capture source divider** 0000 = Capture source/1. 0001 = Capture source/2. 0010 = Capture source/4. 0011 = Capture source/8. 0100 = Capture source/16. 0101 = Capture source/32. 0110 = Capture source/64. 0111 = Capture source/128. 1000 = Capture source/256. 1001~1111 =Reserved. **Note:** Sets INTERCAPSEL (TIMERx_EXTCTL[10:8]) and CAPSRC (TIMERx_CTL[22]) to select capture source.
[27:15]	**Reserved**	Reserved.
[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. **Note:** Set CAPSRC (TIMERx_CTL[22]) and INTERCAPSEL (TIMERx_EXTCTL[10:8]) to select capture source.
[11]	**Reserved**	Reserved.
[10:8]	**INTERCAPSEL**	**Internal Capture Source Select** 000 = Capture Function source is from internal ACMP0 output signal. 001 = Capture Function source is from internal ACMP1 output signal. 010 = Capture Function source is from HXT. 011 = Capture Function source is from LXT. 100 = Capture Function source is from HIRC. 101 = Capture Function source is from LIRC. 110 = Reserved. 111 = Reserved. **Note:** these bits only available when CAPSRC (TIMERx_CTL[22]) is 1.
[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 or ACMP output de-bounce Disabled. 1 = TMx_EXT (x= 0~3) pin de-bounce or ACMP output de-bounce Enabled. **Note:** If this bit is enabled, the edge detection of TMx_EXT pin or ACMP output is detected with de-bounce circuit.
[5]	**CAPIEN**	**Timer External Capture Interrupt Enable Bit** 0 = TMx_EXT (x= 0~3) pin, ACMP, internal clock, or external clock detection Interrupt Disabled. 1 = TMx_EXT (x= 0~3) pin, ACMP, internal clock, or external clock 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 (x= 0~3) pin, ACMP, internal clock, or external clock 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. **Note 1:** 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. **Note 2:** 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 Capture Enable Bit** This bit enables the capture input function. 0 =Capture source Disabled. 1 =Capture source Enabled. **Note:** When CAPEN is 1, user can set INTERCAPSEL (TIMERx_EXTCTL [10:8]) and CAPSRC (TIMERx_CTL[22]) to select capture source.
[2:1]	**Reserved**	Reserved.

[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.
[0]	**CAPIF**	**Timer External Capture Interrupt Flag** This bit indicates the timer external capture interrupt flag status. 0 = TMx_EXT (x= 0~3) pin, ACMP, internal clock, or external clock interrupt did not occur. 1 = TMx_EXT (x= 0~3) pin, ACMP, internal clock, or external clock interrupt occurred. **Note 1:** This bit is cleared by writing 1 to it. **Note 2:** 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, ACMP, internal clock, or external clock matched the CAPEDGE (TIMERx_EXTCTL[2:1]) setting, this bit will set to 1 by hardware. **Note 3:** 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**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**TRGPDMA**	**TRGDAC**	**TRGEADC**	**TRGPWM**	**TRGSSEL**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[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]	**TRGDAC**	**Trigger DAC Enable Bit** If this bit is set to 1, timer time-out interrupt or capture interrupt can be triggered DAC. 0 = Timer interrupt trigger DAC Disabled. 1 = Timer interrupt trigger DAC Enabled. **Note:** If TRGSSEL (TIMERx_TRGCTL[0]) = 0, time-out interrupt signal will trigger DAC. If TRGSSEL (TIMERx_TRGCTL[0]) = 1, capture interrupt signal will trigger DAC.
[2]	**TRGEADC**	**Trigger EADC Enable Bit** If this bit is set to 1, each timer time-out event or capture event can be triggered EADC conversion. 0 = Timer interrupt trigger EADC Disabled. 1 = Timer interrupt trigger EADC Enabled. **Note:** If TRGSSEL (TIMERx_TRGCTL[0]) = 0, time-out interrupt signal will trigger EADC conversion. If TRGSSEL (TIMERx_TRGCTL[0]) = 1, capture interrupt signal will trigger EADC conversion.
[1]	**TRGPWM**	**Trigger EPWM/BPWM Enable Bit** If this bit is set to 1, each timer time-out event or capture event can be as EPWM/BPWM counter clock source.

		0 = Timer interrupt trigger EPWM/BPWM Disabled. 1 = Timer interrupt trigger EPWM/BPWM Enabled. **Note:** If TRGSSEL (TIMERx_TRGCTL[0]) = 0, time-out interrupt signal as EPWM/BPWM counter clock source. If TRGSSEL (TIMERx_TRGCTL[0]) = 1, capture interrupt signal as EPWM/BPWM 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 EPWM, PDMA, DAC, and EADC. 1 = Capture interrupt signal is used to internal trigger EPWM, PDMA, DAC, and EADC.

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							
15	14	13	12	11	10	9	8
Reserved			PWMINTWKEN	Reserved			
7	6	5	4	3	2	1	0
Reserved				CNTMODE	Reserved		CNTEN

Bits	Description	
[31]	**DBGTRIOFF**	**ICE Debug Mode Acknowledge Disable Bit (Write Protect)** 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 bit is write protected. Refer to SYS_REGLCTL control register.
[30]	**DBGHALT**	**ICE Debug Mode Counter Halt (Write Protect)** If debug mode counter halt is enabled, PWM counter will keep current value until exit ICE debug mode. 0 = ICE debug mode counter halt Disabled. 1 = ICE debug mode counter halt Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL control register.
[29:13]	**Reserved**	Reserved.
[12]	**PWMINTWKEN**	**PWM Interrupt Wake-up Enable Bit** If PWM interrupt occurs when chip is in Power-down mode, PWMINTWKEN can determine whether chip wake-up occurs or not. 0 = PWM interrupt wake-up Disabled. 1 = PWM interrupt wake-up Enabled.
[11:4]	**Reserved**	Reserved.
[3]	**CNTMODE**	**PWM Counter Mode** 0 = Auto-reload mode. 1 = One-shot mode.

[2:1]	**Reserved**	Reserved.
[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 Pre-scale Register (TIMERx_PWMCLKPSC)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMCLKPSC	TMR01_BA+0x44	R/W	Timer0 PWM Counter Clock Pre-scale Register	0x0000_0000
TIMER1_PWMCLKPSC	TMR01_BA+0x144	R/W	Timer1 PWM Counter Clock Pre-scale Register	0x0000_0000
TIMER2_PWMCLKPSC	TMR23_BA+0x44	R/W	Timer2 PWM Counter Clock Pre-scale Register	0x0000_0000
TIMER3_PWMCLKPSC	TMR23_BA+0x144	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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CLKPSC							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**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+0x48	R/W	Timer0 PWM Clear Counter Register	0x0000_0000
TIMER1_PWMCNTCLR	TMR01_BA+0x148	R/W	Timer1 PWM Clear Counter Register	0x0000_0000
TIMER2_PWMCNTCLR	TMR23_BA+0x48	R/W	Timer2 PWM Clear Counter Register	0x0000_0000
TIMER3_PWMCNTCLR	TMR23_BA+0x148	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.
[0]	**CNTCLR**	**Clear PWM Counter Control Bit** It is automatically cleared by hardware. 0 = No effect. 1 = Clear 16-bit PWM counter to 0x0000 in up count type. **Note:** Timer peripheral clock source should be set as PCLK to ensure that this bit can be automatically cleared by hardware.

Timer PWM Period Register (TIMERx_PWMPERIOD)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMPERIOD	TMR01_BA+0x4C	R/W	Timer0 PWM Period Register	0x0000_0000
TIMER1_PWMPERIOD	TMR01_BA+0x14C	R/W	Timer1 PWM Period Register	0x0000_0000
TIMER2_PWMPERIOD	TMR23_BA+0x4C	R/W	Timer2 PWM Period Register	0x0000_0000
TIMER3_PWMPERIOD	TMR23_BA+0x14C	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.
[15:0]	**PERIOD**	**PWM Period Register** In up count type: PWM counter counts from 0 to PERIOD, and restarts from 0. In up count type: PWM period time = (PERIOD + 1) * (CLKPSC + 1) * TMRx_PWMCLK.

Timer PWM Comparator Register (TIMERx_PWMCMPDAT)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMCMPDAT	TMR01_BA+0x50	R/W	Timer0 PWM Comparator Register	0x0000_0000
TIMER1_PWMCMPDAT	TMR01_BA+0x150	R/W	Timer1 PWM Comparator Register	0x0000_0000
TIMER2_PWMCMPDAT	TMR23_BA+0x50	R/W	Timer2 PWM Comparator Register	0x0000_0000
TIMER3_PWMCMPDAT	TMR23_BA+0x150	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.
[15:0]	**CMP**	**PWM Comparator Register** PWM CMP is used to compare with PWM CNT to generate PWM output waveform, interrupt events and trigger EADC, PDMA, and DAC start convert.

Timer PWM Counter Register (TIMERx_PWMCNT)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMCNT	TMR01_BA+0x54	R	Timer0 PWM Counter Register	0x0000_0000
TIMER1_PWMCNT	TMR01_BA+0x154	R	Timer1 PWM Counter Register	0x0000_0000
TIMER2_PWMCNT	TMR23_BA+0x54	R	Timer2 PWM Counter Register	0x0000_0000
TIMER3_PWMCNT	TMR23_BA+0x154	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							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[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 Pin Output Polar Control Register (TIMERx_PWMPOLCTL)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMPOLCTL	TMR01_BA+0x58	R/W	Timer0 PWM Pin Output Polar Control Register	0x0000_0000
TIMER1_PWMPOLCTL	TMR01_BA+0x158	R/W	Timer1 PWM Pin Output Polar Control Register	0x0000_0000
TIMER2_PWMPOLCTL	TMR23_BA+0x58	R/W	Timer2 PWM Pin Output Polar Control Register	0x0000_0000
TIMER3_PWMPOLCTL	TMR23_BA+0x158	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							**PINV**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**PINV**	**PWMx Output Pin Polar Control Bit** The bit is used to control polarity state of PWMx_OUT pin. 0 = PWMx_OUT pin polar inverse Disabled. 1 = PWMx_OUT polar inverse Enabled. **Note:** Sets POSEL (TIMERx_PWMPOCTL[8]) to select TMx or TMx_EXT as PWMx output pin.

Timer PWM Pin Output Control Register (TIMERx PWMPOCTL)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMPOCTL	TMR01_BA+0x5C	R/W	Timer0 PWM Pin Output Control Register	0x0000_0000
TIMER1_PWMPOCTL	TMR01_BA+0x15C	R/W	Timer1 PWM Pin Output Control Register	0x0000_0000
TIMER2_PWMPOCTL	TMR23_BA+0x5C	R/W	Timer2 PWM Pin Output Control Register	0x0000_0000
TIMER3_PWMPOCTL	TMR23_BA+0x15C	R/W	Timer3 PWM Pin 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							POSEL
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							POEN

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**POSEL**	**PWM Output Pin Select** 0 = PWMx_OUT pin is TMx. 1 = PWMx_OUT pin is TMx_EXT.
[7:1]	**Reserved**	Reserved.
[0]	**POEN**	**PWMx Output Pin Enable Bit** 0 = PWMx_OUT pin at tri-state mode. 1 = PWMx_OUT pin in output mode. **Note:** Set POSEL (TIMERx_PWMPOCTL[8]) to select TMx or TMx_EXT as PWMx output pin.

Timer PWM Interrupt Enable Register 0 (TIMERx_PWMINTEN0)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMINTEN0	TMR01_BA+0x60	R/W	Timer0 PWM Interrupt Enable Register 0	0x0000_0000
TIMER1_PWMINTEN0	TMR01_BA+0x160	R/W	Timer1 PWM Interrupt Enable Register 0	0x0000_0000
TIMER2_PWMINTEN0	TMR23_BA+0x60	R/W	Timer2 PWM Interrupt Enable Register 0	0x0000_0000
TIMER3_PWMINTEN0	TMR23_BA+0x160	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					CMPUIEN	PIEN	Reserved

Bits	Description	
[31:3]	**Reserved**	Reserved.
[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.
[0]	**Reserved**	Reserved.

Timer PWM Interrupt Status Register 0 (TIMERx_PWMINTSTS0)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMINTSTS0	TMR01_BA+0x64	R/W	Timer0 PWM Interrupt Status Register 0	0x0000_0000
TIMER1_PWMINTSTS0	TMR01_BA+0x164	R/W	Timer1 PWM Interrupt Status Register 0	0x0000_0000
TIMER2_PWMINTSTS0	TMR23_BA+0x64	R/W	Timer2 PWM Interrupt Status Register 0	0x0000_0000
TIMER3_PWMINTSTS0	TMR23_BA+0x164	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					CMPUIF	PIF	Reserved

Bits	Description	
[31:3]	**Reserved**	Reserved.
[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. **Note 1:** If CMP equal to PERIOD, there is no CMPUIF flag in up count type. **Note 2:** 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. **Note:** This bit is cleared by writing 1 to it.
[0]	**Reserved**	Reserved.

Timer PWM Trigger Control Register (TIMERx_PWMTRGCTL)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMTRGCTL	TMR01_BA+0x68	R/W	Timer0 PWM Trigger Control Register	0x0000_0000
TIMER1_PWMTRGCTL	TMR01_BA+0x168	R/W	Timer1 PWM Trigger Control Register	0x0000_0000
TIMER2_PWMTRGCTL	TMR23_BA+0x68	R/W	Timer2 PWM Trigger Control Register	0x0000_0000
TIMER3_PWMTRGCTL	TMR23_BA+0x168	R/W	Timer3 PWM 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						PWMTRGPDMA	PWMTRGDAC
7	6	5	4	3	2	1	0
PWMTRGEADC	Reserved					TRGSEL	

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**PWMTRGPDMA**	**PWM Counter Event Trigger PDMA Conversion Enable Bit** If this bit is set to 1, PWM can trigger PDMA conversion. 0 = PWM trigger PDMA Disabled. 1 = PWM trigger PDMA Enabled. **Note:** Set TRGSEL (TIMERx_PWMTRGCTL[1:0]) to select PWM trigger conversion source.
[8]	**PWMTRGDAC**	**PWM Counter Event Trigger DAC Conversion Enable Bit** If this bit is set to 1, PWM can trigger DAC conversion. 0 = PWM trigger DAC Disabled. 1 = PWM trigger DAC Enabled. **Note:** Set TRGSEL (TIMERx_PWMTRGCTL[1:0]) to select PWM trigger conversion source.
[7]	**PWMTRGEADC**	**PWM Counter Event Trigger EADC Conversion Enable Bit** 0 = PWM counter event trigger EADC conversion Disabled. 1 = PWM counter event trigger EADC conversion Enabled. **Note:** Set TRGSEL (TIMERx_PWMTRGCTL[1:0]) to select PWM trigger conversion source.
[6:2]	**Reserved**	Reserved.
[1:0]	**TRGSEL**	**PWM Counter Event Source Select to Trigger Conversion** 00 = Trigger conversion at period point (PIF). 01 = Trigger conversion at compare up count point (CMPUIF). 10 = Trigger conversion at period or compare up count point (PIF or CMPUIF). 11 = Reserved.

Timer PWM Status Register (TIMERx_PWMSTATUS)

Register	Offset	R/W	Description	Reset Value
TIMER0_PWMSTATUS	TMR01_BA+0x6C	R/W	Timer0 PWM Status Register	0x0000_0000
TIMER1_PWMSTATUS	TMR01_BA+0x16C	R/W	Timer1 PWM Status Register	0x0000_0000
TIMER2_PWMSTATUS	TMR23_BA+0x6C	R/W	Timer2 PWM Status Register	0x0000_0000
TIMER3_PWMSTATUS	TMR23_BA+0x16C	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					**PDMATRGF**	**DACTRGF**	**EADCTRGF**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**PWMINTWKF**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**CNTMAXF**

Bits	Description	
[31:19]	**Reserved**	Reserved.
[18]	**PDMATRGF**	**Trigger PDMA Start Conversion Flag** 0 = PWM counter event trigger PDMA start conversion has not occurred. 1 = PWM counter event trigger PDMA start conversion has occurred. **Note:** This bit is cleared by writing 1 to it.
[17]	**DACTRGF**	**Trigger DAC Start Conversion Flag** 0 = PWM counter event trigger DAC start conversion has not occurred. 1 = PWM counter event trigger DAC start conversion has occurred. **Note:** This bit is cleared by writing 1 to it.
[16]	**EADCTRGF**	**Trigger EADC Start Conversion Flag** 0 = PWM counter event trigger EADC start conversion is not occurred. 1 = PWM counter event trigger EADC start conversion has occurred. **Note:** This bit is cleared by writing 1 to it.
[15:9]	**Reserved**	Reserved.
[8]	**PWMINTWKF**	**PWM Interrupt Wake-up Flag** 0 = PWM interrupt wake-up has not occurred. 1 = PWM interrupt wake-up has occurred. **Note:** This bit is cleared by writing 1 to it.
[7:1]	**Reserved**	Reserved.
[0]	**CNTMAXF**	**PWM Counter Equal to 0xFFFF Flag**

		0 = The PWM counter value never reached its maximum value 0xFFFF. 1 = 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+0x70	R	Timer0 PWM Period Buffer Register	0x0000_0000
TIMER1_PWMPBUF	TMR01_BA+0x170	R	Timer1 PWM Period Buffer Register	0x0000_0000
TIMER2_PWMPBUF	TMR23_BA+0x70	R	Timer2 PWM Period Buffer Register	0x0000_0000
TIMER3_PWMPBUF	TMR23_BA+0x170	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.
[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+0x74	R	Timer0 PWM Comparator Buffer Register	0x0000_0000
TIMER1_PWMCMPBUF	TMR01_BA+0x174	R	Timer1 PWM Comparator Buffer Register	0x0000_0000
TIMER2_PWMCMPBUF	TMR23_BA+0x74	R	Timer2 PWM Comparator Buffer Register	0x0000_0000
TIMER3_PWMCMPBUF	TMR23_BA+0x174	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.
[15:0]	**CMPBUF**	**PWM Comparator Buffer Register (Read Only)** Used as CMP active register.

6.9 Watchdog Timer (WDT)

6.9.1 Overview

The Watchdog Timer (WDT) is used 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

- 20-bit free running up counter for WDT time-out interval
- Selectable time-out interval (2^4 ~ 2^{20}) and the time-out interval is 0.416 ms ~ 27.306 s if WDT_CLK = 38.4 kHz.
- System kept in reset state for a period of (1 / WDT_CLK) * 63
- Supports selectable WDT reset delay period, including 1026、130、18 or 3 WDT_CLK reset delay period
- Supports to force WDT enabled after chip powered on or reset by setting CWDTEN[2:0] in Config0 register
- Supports WDT time-out wake-up function only if WDT clock source is selected as 38.4 kHz or LXT.

6.9.3 Block Diagram

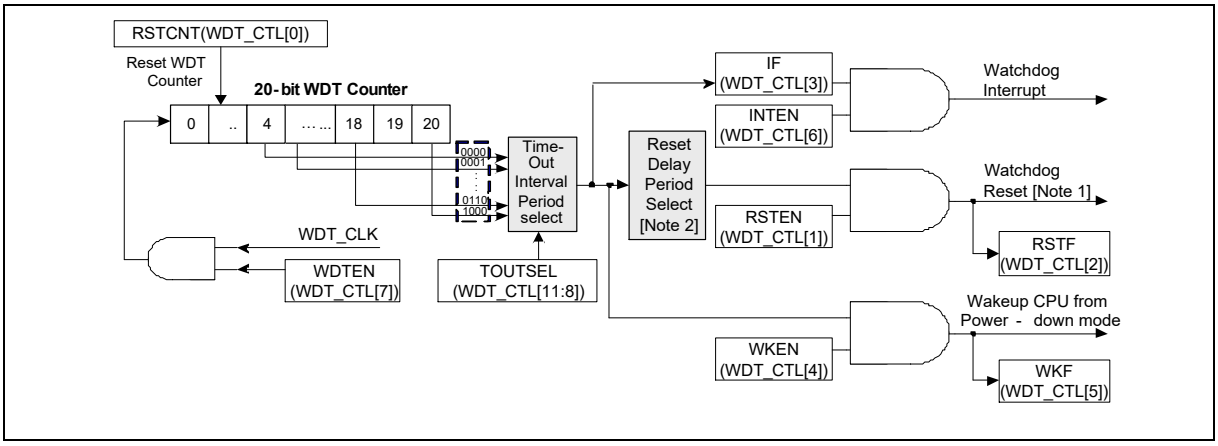


Figure 6.9-1 Watchdog Timer Block Diagram

Note1: WDT resets CPU and lasts 63 WDT_CLK.

Note2: 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 powered on or reset in CWDTEN[2:0] (CWDTEN[2] is Config0[31], CWDTEN[1:0] is Config0[4:3])

The WDT clock control is shown in Figure 6.9-2.

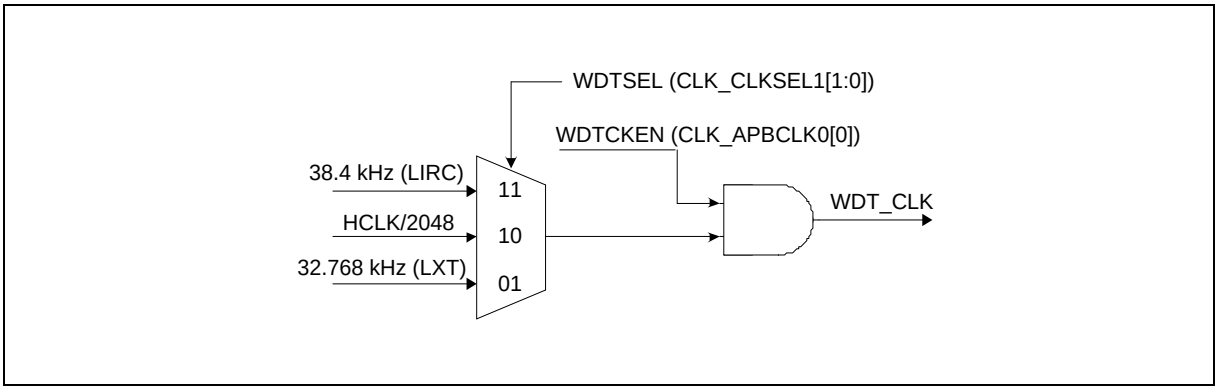


Figure 6.9-2 Watchdog Timer Clock Control

6.9.5 Functional Description

The WDT includes an 20-bit free running up counter with programmable time-out intervals. Table 6.9-1 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

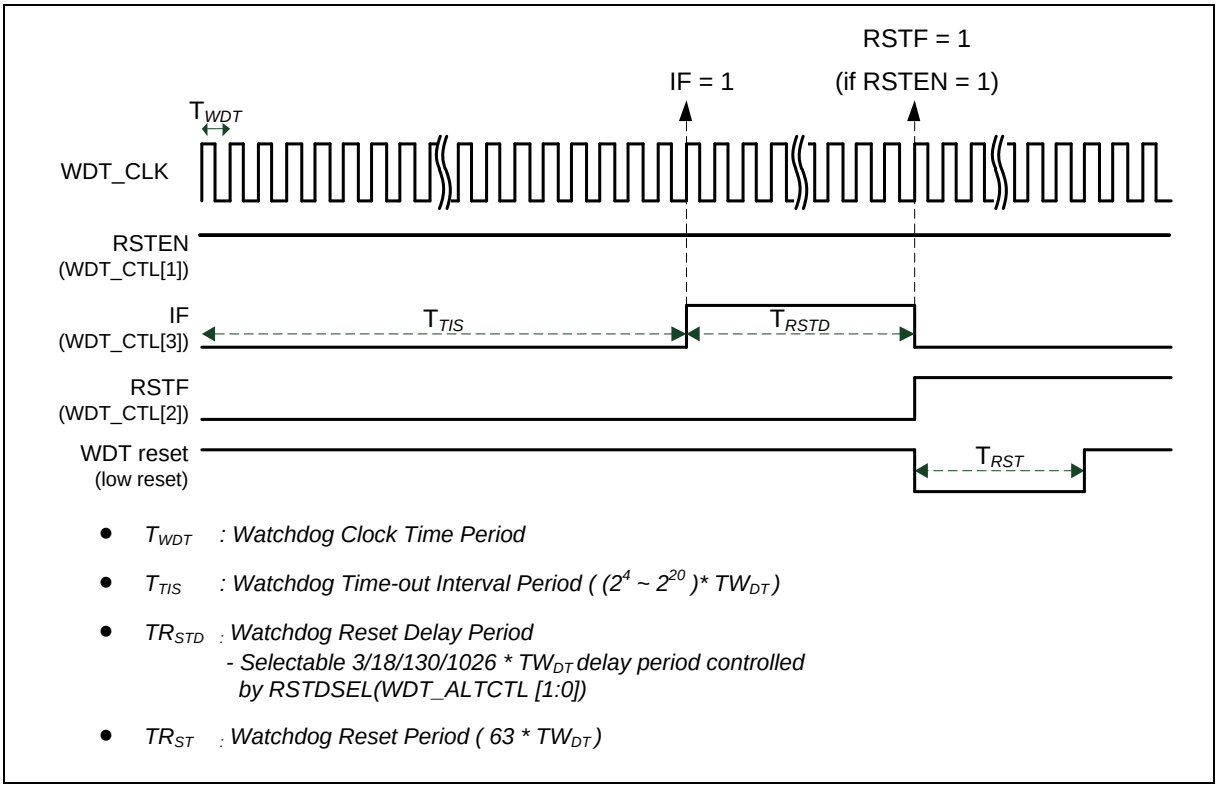
Setting WDTEN (WDTCR[7]) to 1 will enable the WDT function and the WDT counter to start counting up. The SYNC (WDT_CTL[30]) can be indicated whether enable/disable WDTEN function is completed or not. There are eight time-out interval period can be selected by setting TOUTSEL (WDTCR[11:8]). When the WDT up counter reaches the TOUTSEL (WDTCR[11:8]) 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 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]) or set WDT_RSTCNT to reset the 20-bit WDT up counter value to avoid generate WDT time-out reset signal before the T_{RSTD} reset delay period expires. Moreover, user should set RSTDSEL (WDT_ALTCTL [1:0]) to select reset delay period to clear WDT counter. If the WDT up counter value has not been cleared after the specific T_{RSTD} delay period expires, the WDT control will set RSTF (WDT_CTL[2]) to 1 if RSTEN (WDT_CTL[1]) bit is enabled, then chip enters to reset state immediately. Refer to Figure 6.9-3. T_{RST} reset period will keep last 63 WDT clocks then chip restart executing program from reset vector (0x0000_0000). The RSTF (WDT_CTL[2]) will keep 1 after WDT time-out resets 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}
0000	$2^4 * T_{WDT}$	(3/18/130/1026) * T_{WDT}
0001	$2^6 * T_{WDT}$	(3/18/130/1026) * T_{WDT}
0010	$2^8 * T_{WDT}$	(3/18/130/1026) * T_{WDT}
0011	$2^{10} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
0100	$2^{12} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
0101	$2^{14} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
0110	$2^{16} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
0111	$2^{18} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
1000	$2^{20} * T_{WDT}$	(3/18/130/1026) * T_{WDT}

Table 6.9-1 Watchdog Timer Time-out Interval Period Selection



- T_{WDT} : *Watchdog Clock Time Period*
- T_{TIS} : *Watchdog Time-out Interval Period ((2^4 ~ 2^{20})* TW_{DT})*
- TR_{STD} : *Watchdog Reset Delay Period*
  *- Selectable 3/18/130/1026 * TW_{DT} delay period controlled by RSTDSEL(WDT_ALTCTL [1:0])*
- TR_{ST} : *Watchdog Reset Period (63 * TW_{DT})*

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 38.4 kHz or LXT, system can be woken up from Power-down mode while WDT time-out interrupt signal is generated and WKEN (WDT_CTL[4]) enabled. Note that user should set LXTEN (CLK_PWRCTL [1]) or LIRCEN (CLK_PWRCTL [3]) to select clock source before system enters Power-down mode because the system peripheral clock are disabled when system is in Power-down mode. In the meanwhile, the WKF (WDT_CTL[5]) will be set to 1 automatically, and user can check WKF (WDT_CTL[5]) status by software to recognize the system has been woken 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_0800
WDT_ALTCTL	WDT_BA+0x04	R/W	WDT Alternative Control Register	0x0000_0000
WDT_RSTCNT	WDT_BA+0x08	W	WDT Reset Counter Register	0x0000_0000

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_0800

31	30	29	28	27	26	25	24
ICEDEBUG	**SYNC**	**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 Bit (Write Protect)** 0 = ICE debug mode acknowledgement affects WDT counting. WDT up counter will be held while CPU is held by ICE. 1 = ICE debug mode acknowledgement Disabled. WDT up counter will keep going no matter CPU is held by ICE or not. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[30]	**SYNC**	**WDT Enable Control SYNC Flag Indicator (Read Only)** If user executes enable/disable WDTEN (WDT_CTL[7]), this flag can be indicated enable/disable WDTEN function is completed or not. 0 = Set WDTEN bit is completed. 1 = Set WDTEN bit is synchronizing and not become active yet. **Note:** Performing enable or disable WDTEN bit needs 2 * WDT_CLK period to become active.
[29:12]	**Reserved**	Reserved.
[11:8]	**TOUTSEL**	**WDT Time-out Interval Selection (Write Protect)** These three bits select the time-out interval period for the WDT. 0000 = 2^4 * WDT_CLK. 0001 = 2^6 * WDT_CLK. 0010 = 2^8 * WDT_CLK. 0011 = 2^{10} * WDT_CLK. 0100 = 2^{12} * WDT_CLK. 0101 = 2^{14} * WDT_CLK. 0110 = 2^{16} * WDT_CLK. 0111 = 2^{18} * WDT_CLK. 1000 = 2^{20} * WDT_CLK. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[7]	**WDTEN**	**WDT Enable Bit (Write Protect)**

		0 = WDT Disabled (This action will reset the internal up counter value). 1 = WDT Enabled. **Note 1:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note 2:** If CWDTEN[2:0] (combined by Config0[31] and Config0[4:3]) bits is not configured to 111, this bit is forced as 1 and user cannot change this bit to 0.
[6]	**INTEN**	**WDT Time-out Interrupt Enable Bit (Write Protect)** If this bit is enabled, the WDT time-out interrupt signal is generated and inform to CPU. 0 = WDT time-out interrupt Disabled. 1 = WDT time-out interrupt Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[5]	**WKF**	**WDT Time-out Wake-up Flag (Write Protect)** This bit indicates the interrupt wake-up flag status of WDT 0 = WDT does not cause chip wake-up. 1 = Chip wake-up from Idle or Power-down mode if WDT time-out interrupt signal generated. **Note 1:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note 2:** This bit is cleared by writing 1 to it.
[4]	**WKEN**	**WDT Time-out Wake-up Function Control (Write Protect)** If this bit is set to 1, while WDT time-out interrupt flag IF (WDT_CTL[3]) is generated to 1 and interrupt enable bit INTEN (WDT_CTL[6]) is enabled, the WDT time-out interrupt signal will generate a wake-up trigger event to chip. 0 = Wake-up trigger event Disabled if WDT time-out interrupt signal generated. 1 = Wake-up trigger event Enabled if WDT time-out interrupt signal generated. **Note 1:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note 2:** Chip can be woken up by WDT time-out interrupt signal generated only if WDT clock source is selected to 38.4 kHz internal low speed RC oscillator (LIRC) or LXT.
[3]	**IF**	**WDT Time-out Interrupt Flag** This bit will set to 1 while WDT up counter value reaches the selected WDT time-out interval 0 = WDT time-out interrupt did not occur. 1 = WDT time-out interrupt occurred. **Note:** This bit is cleared by writing 1 to it.
[2]	**RSTF**	**WDT Time-out Reset Flag** This bit indicates the system has been reset by WDT time-out reset or not. 0 = WDT time-out reset did not occur. 1 = WDT time-out reset occurred. **Note:** This bit is cleared by writing 1 to it.
[1]	**RSTEN**	**WDT Time-out Reset Enable Bit (Write Protect)** Setting this bit will enable the WDT time-out reset function If the WDT up counter value has not been cleared after the specific WDT reset delay period expires. 0 = WDT time-out reset function Disabled. 1 = WDT time-out reset function Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[0]	**RSTCNT**	**Reset WDT Up Counter (Write Protect)** 0 = No effect. 1 = Reset the internal 20-bit WDT up counter value. **Note 1:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note 2:** This bit will be automatically cleared by hardware.

WDT Alternative Control Register (WDT_ALTCTL)

Register	Offset	R/W	Description	Reset Value
WDT_ALTCTL	WDT_BA+0x04	R/W	WDT Alternative Control Register	0x0000_0000

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

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1:0]	**RSTDSEL**	**WDT Reset Delay Selection (Write Protect)** When WDT time-out happened, user has a time named WDT Reset Delay Period to clear WDT counter by setting RSTCNT (WDT_CTL[0]) to prevent WDT time-out reset happened. User can select a suitable setting of RSTDSEL for different WDT Reset Delay Period. 00 = WDT Reset Delay Period is 1026 * WDT_CLK. 01 = WDT Reset Delay Period is 130 * WDT_CLK. 10 = WDT Reset Delay Period is 18 * WDT_CLK. 11 = WDT Reset Delay Period is 3 * WDT_CLK. **Note 1:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note 2:** This register will be reset to 0 if WDT time-out reset happened.

WDT Reset Counter Register (WDT_RSTCNT)

Register	Offset	R/W	Description	Reset Value
WDT_RSTCNT	WDT_BA+0x08	W	WDT Reset Counter Register	0x0000_0000

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

Bits	Description	
[31:0]	RSTCNT	**WDT Reset Counter Register** Writing 0x00005AA5 to this field will reset the internal 20-bit WDT up counter value to 0. **Note 1:** Performing RSTCNT to reset counter needs 2 * WDT_CLK period to become active. **Note 2:** RSTCNT (WDT_CTL[0]) bit is a write protected bit. RSTCNT (WDT_RSTCNT[31:0]) bits are not write protected.

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 running 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

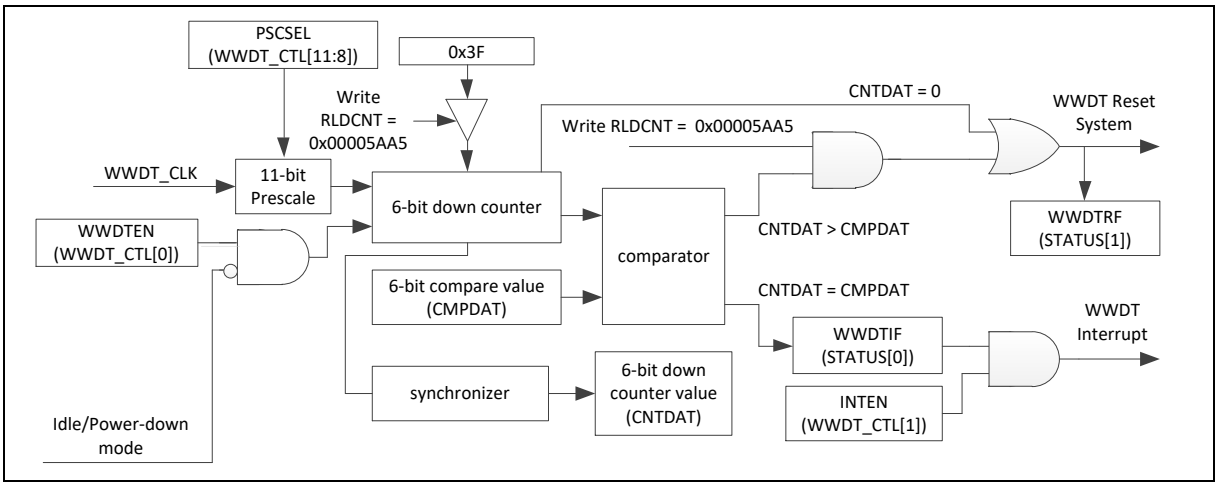


Figure 6.10-1 WWDT Block Diagram

6.10.4 Basic Configuration

- Clock Source Configuration
  - Select the source of WWDT peripheral clock on WWDTSEL (CLK_CLKSEL1[31:30])
  - Enable WWDT peripheral clock in WDTCKEN (CLK_APBCLK0[0]).

The WWDT clock control is shown in Figure 6.10-2.

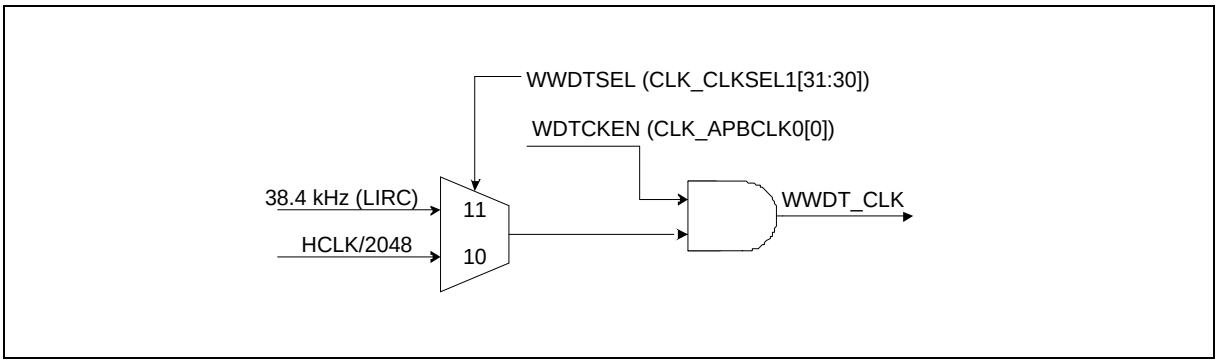


Figure 6.10-2 WWDT Clock Control

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 38.4 kHz internal low speed RC oscillator (LIRC) with a programmable 11-bit prescale counter value which controlled by PSCSEL (WWDT_CTL[11:8]). Also, the correlate of PSCSEL (WWDT_CTL[11:8]) and prescale value are listed in Table 6.10-1.

PSCSEL	Prescaler Value	Max. Time-Out Period	Max. Time-Out Interval (WWDT_CLK=38.4 kHz)
0000	1	1 * 64 * T_{WWDT}	1.664 ms
0001	2	2 * 64 * T_{WWDT}	3.328 ms
0010	4	4 * 64 * T_{WWDT}	6.656 ms
0011	8	8 * 64 * T_{WWDT}	13.312 ms
0100	16	16 * 64 * T_{WWDT}	26.624 ms
0101	32	32 * 64 * T_{WWDT}	53.248 ms
0110	64	64 * 64 * T_{WWDT}	106.496 ms
0111	128	128 * 64 * T_{WWDT}	212.992 ms
1000	192	192 * 64 * T_{WWDT}	0.32 s
1001	256	256 * 64 * T_{WWDT}	0.426 s
1010	384	384 * 64 * T_{WWDT}	0.639 s
1011	512	512 * 64 * T_{WWDT}	0.852 s
1100	768	768 * 64 * T_{WWDT}	1.278 s
1101	1024	1024 * 64 * T_{WWDT}	1.704 s
1110	1536	1536 * 64 * T_{WWDT}	2.554 s
1111	2048	2048 * 64 * T_{WWDT}	3.4078 s

Table 6.10-1 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.

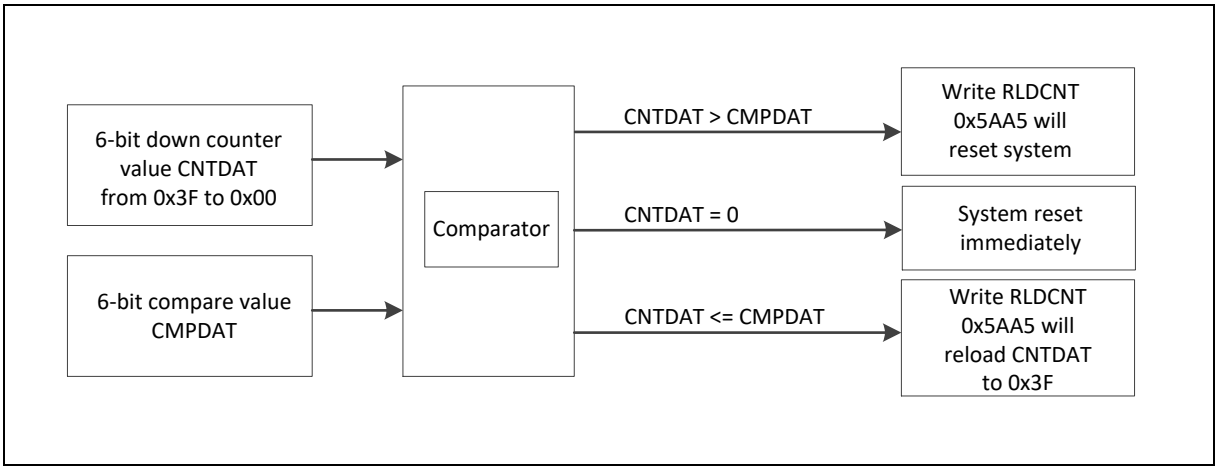


Figure 6.10-3 WWDT Reset and Reload Behavior

If the 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.

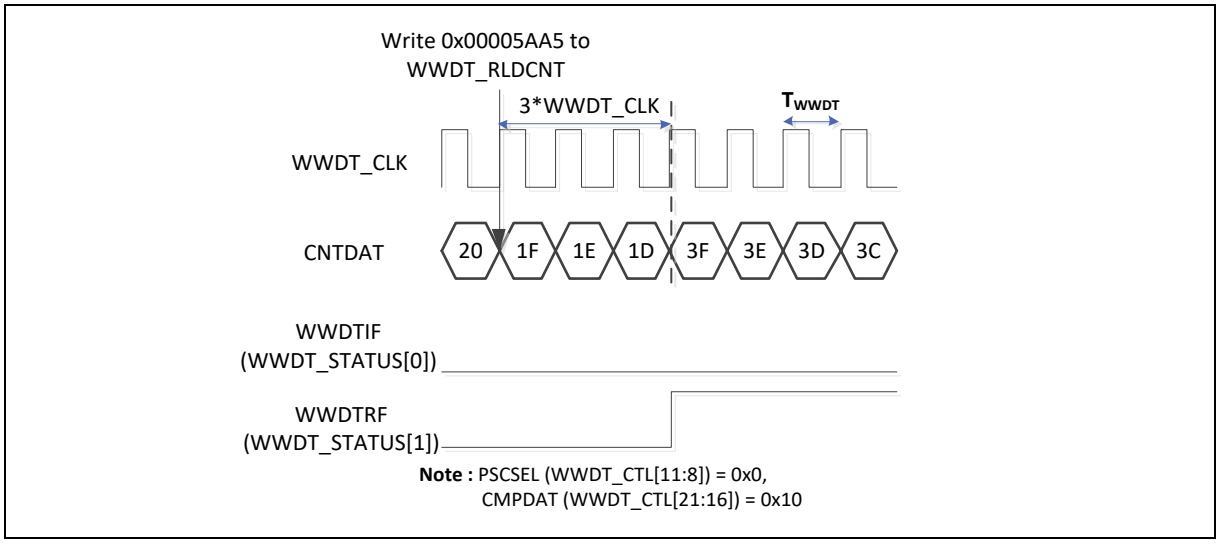


Figure 6.10-4 WWDT Reload Counter When CNTDAT > CMPDAT

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 the WWDT counter value from reaching 0 and generate WWDT reset system signal to inform system reset. Figure 6.10-5 shows the waveform of WWDT reload counter when CNTDAT is less than CMPDAT and Figure 6.10-6 shows the WWDT generates reset system signal (WWDTRF) if user does not write 0x00005AA5 to WWDT_RLD before WWDT counter value reaches 0.

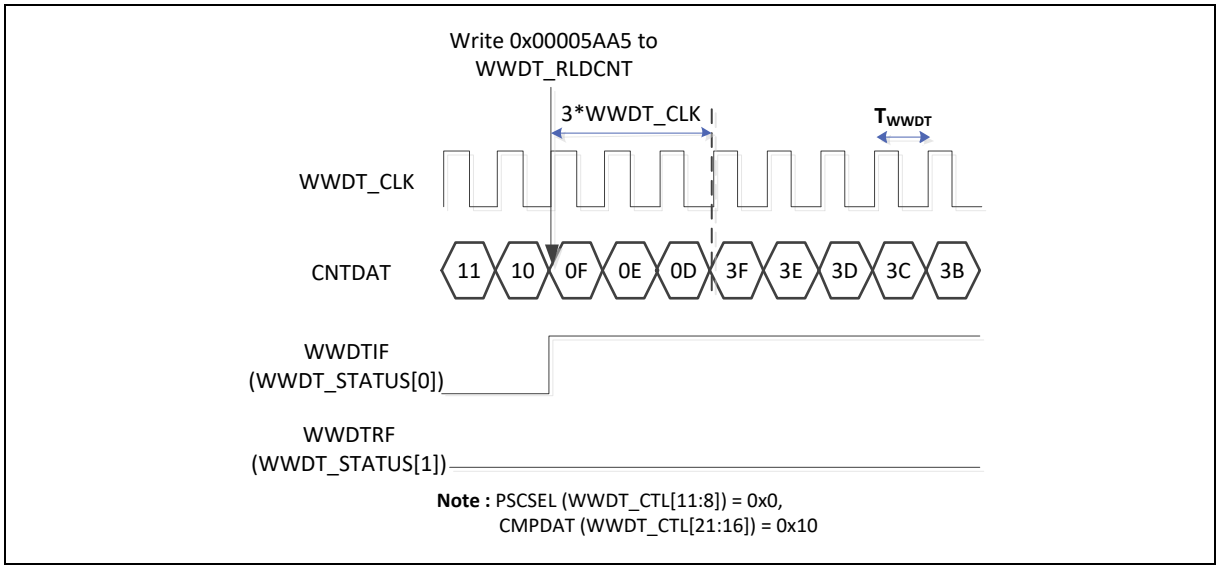


Figure 6.10-5 WWDT Reload Counter When WWDT_CNT < WINCMP

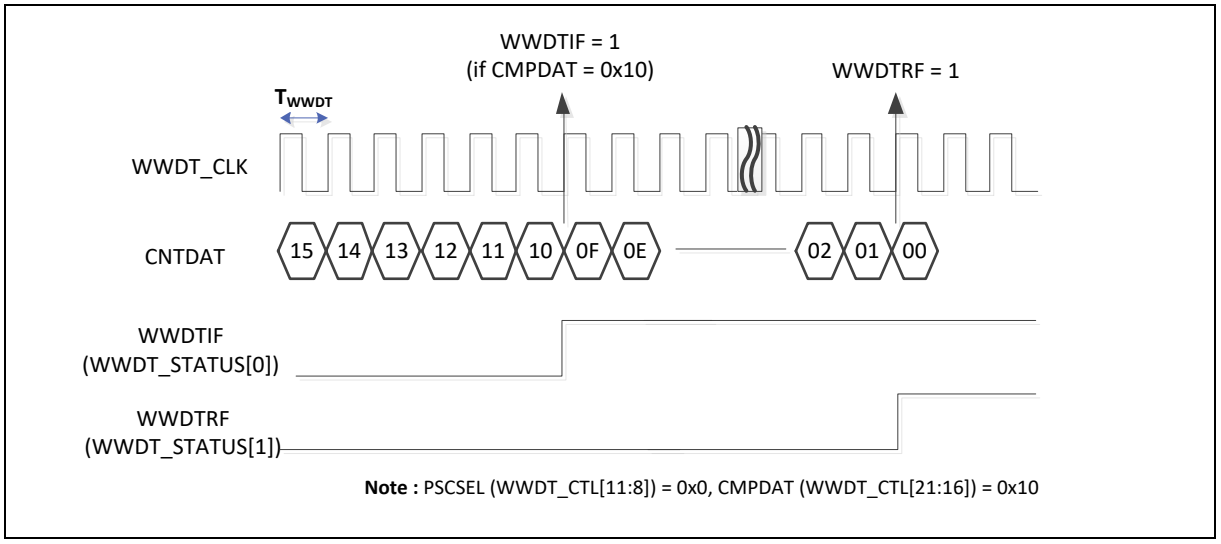


Figure 6.10-6 WWDT Interrupt and Reset Signals

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. Note that if user sets PSCSEL (WWDT_CTL[11:8]) to 0000, the counter prescale value should be as 1, and the CMPDAT (WWDT_CTL[21:16]) must be larger than 2. Otherwise, writing WWDT_RLDCNT register to reload WWDT counter value to 0x3F is unavailable, WWDTIF (WWDT_STATUS[0]) is generated, and WWDT reset system event always happened. The WWDT CMPDAT setting limitation is shown in Table 6.10-2.

If user sets CMPDATA as 0x3F and 0x0, the interrupt doesn't occur. The reset occurs when WWDT counts to 0x0, so the interrupt doesn't occur when CMPDATA is 0x0.

PSCSEL	Prescale Value	Valid CMPDAT Value
0000	1	0x3 ~ 0x3E

0001	2	0x2 ~ 0x3E
Others	Others	0x1 ~ 0x3E

Table 6.10-2 CMPDAT Setting Limitation

6.10.5.5 WWDT ICE Debug

When ICE is connected to MCU, the WWDT counter is counting (or not) by ICEDEBUG (WWDT_CTL[31]). The default value of ICEDEBUG is 0. The 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

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 be generated 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 written 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 Bit** 0 = ICE debug mode acknowledgement effects WWDT counting. WWDT down counter will be held while CPU is held by ICE. 1 = ICE debug mode acknowledgement Disabled. **Note:** WWDT down counter will keep going no matter CPU is held by ICE or not.
[30:22]	**Reserved**	Reserved.
[21:16]	**CMPDAT**	**WWDT Window Compare** 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.
[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.
[1]	**INTEN**	**WWDT Interrupt Enable 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 Bit** 0 = WWDT counter is stopped. 1 = WWDT counter starts counting.

WWDT Status Register (WWDT_STATUS)

Register	Offset	R/W	Description	Reset Value
WWDT_STATUS	WWDT_BA+0x08	R/W	WWDT Status Register	0x0000_0000

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

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**WWDTRF**	**WWDT Timer-out Reset Flag** 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.
[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 messages 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) settings 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 1 Hz clock output.
- Supports chip wake-up from Idle or Power-down mode while a RTC interrupt signal is generated.
- Supports Daylight Saving Time software control in RTC_DSTCTL.

6.11.3 Block Diagram

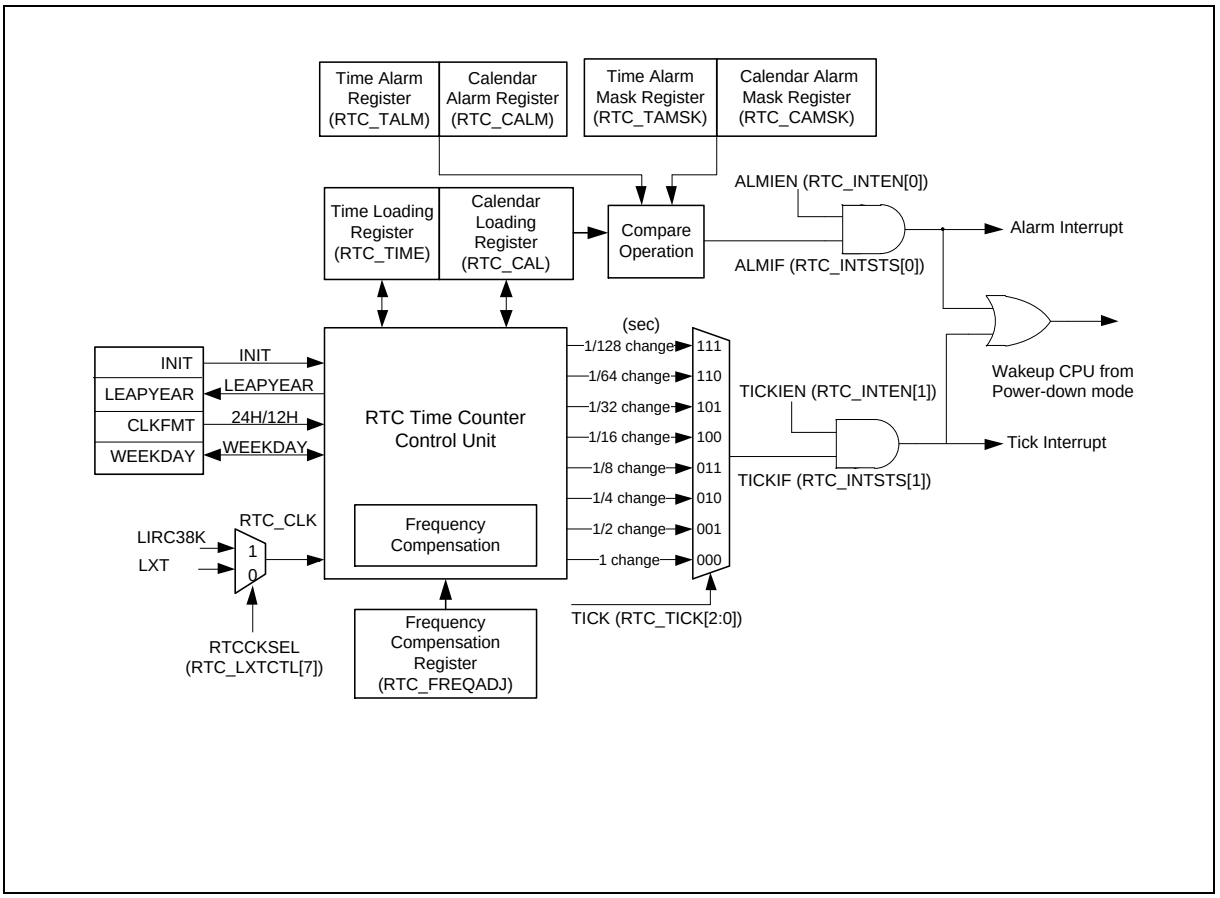


Figure 6.11-1 RTC Block Diagram

6.11.4 Basic Configuration

- Clock Source Configuration
  - The RTC controller clock source is enabled by RTCCKEN (APBCLK0[1])
  - RTC Time Counter source is selected from LXT or LIRC by RTCCKSEL (RTC_LXTCTL[7]).

6.11.5 Functional Description

6.11.5.1 RTC Initiation

When a RTC block is powered 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 RTC_INIT register is written as 0xa5eb1357, the RTC will be in normal active state permanently. User can read Active bit (RTC_INIT[0]) to check the RTC is at normal active state or reset state.

6.11.5.2 RTC Read/Write Enable

The RTC control registers access attribute when Active(RTC_INIT[0]) is 1 and 0 are shown in Table 6.11-1 .

Register	Active = 0	Active = 1
RTC_INIT	R/W	R/W

RTC_FREQADJ	R/W	R/W
RTC_TIME	Not available	R/W
RTC_CAL	Not available	R/W
RTC_CLKFMT	Not available	R/W
RTC_WEEKDAY	Not available	R/W
RTC_TALM	Not available	R/W
RTC_CALM	Not available	R/W
RTC_LEAPYEAR	Not available	R
RTC_INTEN	R/W	R/W
RTC_INTSTS	R/W	R/W
RTC_TICK	Not available	R/W
RTC_TAMSK	Not available	R/W
RTC_CAMSK	Not available	R/W
RTC_LXTCTL	R/W	R/W
RTC_DSTCTL	Not available	R/W
RTC_TEST	R/W	R/W
RTC_ACCCTL	R/W	R/W
RTC_VERSION	R/W	R/W

Table 6.11-1 RTC Read/Write Enable

6.11.5.3 Frequency Compensation

Frequency compensation circuit supports dynamic compensation to adjust compensation value without the compensation circuit reset. The enable bit for dynamic compensation is DYNCOMPEN (RTC_CLKFMT[16]).

If dynamic compensation is enabled, the minimal interval to continuous writing RTC_FREQADJ is 0.024 seconds, and FCRBUSY(RTC_FREQADJ[31]) flag is used to indicate that new register write operation is prohibited. And the compensate value will load into compensation circuit after 0.024~1.016 seconds when write to RTC_FREQADJ.

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

Integer Part: 32773 => RTC_FREQADJ[12:8] = 0x15, Refer the INTEGER(RTC_FREQADJ[12:8]) to get detail setting value.

Fraction Part: 0.65 X 64 = 41.6(0x2A) => RTC_FREQADJ[5:0]=0x2A

Example 2:

Frequency counter measurement: 32763.25 Hz

Integer part: 32763=> RTC_FREQADJ[12:8] = 0x0B, Refer the INTEGER(RTC_FREQADJ[12:8]) to get detail setting value.

Fraction part: 0.25 X 64 = 16(0x10) => RTC_FREQADJ[5:0] = 0x10

Note: The value of RTC_FREQADJ register will be the default value (0x0000_1000) while the compensation is not executed. User can utilize a frequency counter to measure RTC clock source via clock output function in manufacturing. Meanwhile, user can check the result of RTC frequency compensation using the clock output function .

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.)

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 + 0x20) (24HEN = 0) (PM Time + 0x20)	
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 6.11-212/24 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 Daylight Saving Time

The RTC controller also provides RTC_DSTCTL register to store the control settings of daylight saving time application. User can read RTC_DSTCTL value to check current RTC date/time counter runs in daylight saving time mode or normal mode.

6.11.5.10 1 Hz clock output

The RTC controller provides 1Hz clock output to CLKO function pin. User can set CLK1HZEN (CLK_CLKOCTL[6]) to 1 and enable RTC, 1Hz clock will output to CLKO function pin.

6.11.5.11 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:
   1) HOUR:
      RTC_TIME[21:16]: 0x32 (0x12+0x20) combined by TENHR (RTC_TIME[21:20]) is 0x3, HR (RTC_TIME[19:16]) is 0x2.
   2) MIN:
      RTC_TIME[14:8]: 0x59 combined by TENMIN (RTC_TIME[14:12]) is 0x5, MIN (RTC_TIME[11:8]) is 0x9.
   3) SEC:
      RTC_TIME[6:0]: 0x30 combined by TENSEC (RTC_TIME[6:4]) is 0x3, SEC (RTC_TIME[3:0]) is 0x0.

Table 6.11-3 shows registers value after first powered on.

Register	Reset State
RTC_INIT	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
RTC_DSTCTL	0

Table 6.11-3 Register Value After Powered 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_FREQADJ	RTC_BA+0x08	R/W	RTC Frequency Compensation Register	0x0000_1000
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
RTC_DSTCTL	RTC_BA+0x110	R/W	RTC Daylight Saving Time Control Register	0x0000_0000

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							**ACTIVE**

Bits	Description	
[31:1]	**INIT**	**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 leave 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]	**ACTIVE**	**RTC Active Status (Read Only)** 0 = RTC is at reset state. 1 = RTC is at normal active state.

RTC Frequency Compensation Register (RTC_FREQADJ)

Register	Offset	R/W	Description	Reset Value
RTC_FREQADJ	RTC_BA+0x08	R/W	RTC Frequency Compensation Register	0x0000_1000

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

Bits	Description	
[31]	**FCRBUSY**	**Frequency Compensation Register Write Operation Busy (Read Only)** 0 = The new register write operation is acceptable. 1 = The last write operation is in progress and new register write operation prohibited. **Note:** This bit is only used when DYNCOMPEN(RTC_CLKFMT[16]) is enabled.
[30:13]	**Reserved**	Reserved.
[12:8]	**INTEGER**	**Integer Part** 00000 = Integer part of detected value is 32752. 00001 = Integer part of detected value is 32753. 00010 = Integer part of detected value is 32754. 00011 = Integer part of detected value is 32755. 00100 = Integer part of detected value is 32756. 00101 = Integer part of detected value is 32757. 00110 = Integer part of detected value is 32758. 00111 = Integer part of detected value is 32759. 01000 = Integer part of detected value is 32760. 01001 = Integer part of detected value is 32761. 01010 = Integer part of detected value is 32762. 01011 = Integer part of detected value is 32763. 01100 = Integer part of detected value is 32764. 01101 = Integer part of detected value is 32765. 01110 = Integer part of detected value is 32766. 01111 = Integer part of detected value is 32767. 10000 = Integer part of detected value is 32768. 10001 = Integer part of detected value is 32769. 10010 = Integer part of detected value is 32770. 10011 = Integer part of detected value is 32771. 10100 = Integer part of detected value is 32772. 10101 = Integer part of detected value is 32773.

		10110 = Integer part of detected value is 32774. 10111 = Integer part of detected value is 32775. 11000 = Integer part of detected value is 32776. 11001 = Integer part of detected value is 32777. 11010 = Integer part of detected value is 32778. 11011 = Integer part of detected value is 32779. 11100 = Integer part of detected value is 32780. 11101 = Integer part of detected value is 32781. 11110 = Integer part of detected value is 32782. 11111 = Integer part of detected value is 32783.
[7:6]	**Reserved**	Reserved.
[5:0]	**FRACTION**	**Fraction Part** Formula: FRACTION = (fraction part of detected value) x 64. **Note:** Digit in FCR must be expressed as hexadecimal number.

Note: If dynamic compensation is not enable, i.e. DYNCOMPEN(RTC_CLKFMT[16]) = 0, FREQADJ's counter will be reset for start to compensate when writing RTC_FREQADJ. Imply RTC Time will be restarted.

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.
[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.
[14:12]	**TENMIN**	10-Min Time Digit (0~5)
[11:8]	**MIN**	1-Min Time Digit (0~9)
[7]	**Reserved**	Reserved.
[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.
3. FREQADJ's counter will be reset for start to Compensatie when writing RTC_TIME, RTC_CAL, RTC_WEEKDAY. Imply RTC Time will be restarted.

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.
[23:20]	**TENYEAR**	10-Year Calendar Digit (0~9)
[19:16]	**YEAR**	1-Year Calendar Digit (0~9)
[15:13]	**Reserved**	Reserved.
[12]	**TENMON**	10-Month Calendar Digit (0~1)
[11:8]	**MON**	1-Month Calendar Digit (0~9)
[7:6]	**Reserved**	Reserved.
[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.
3. FREQADJ's counter will be reset for start to Compensatie when writing RTC_TIME, RTC_CAL, RTC_WEEKDAY. Imply RTC Time will be restarted.

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							**DYNCOMPEN**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**24HEN**

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**DYNCOMPEN**	**Dynamic Compensation Enable Bit** 0 = Dynamic Compensation Disabled. 1 = Dynamic Compensation Enabled.
[15:1]	**Reserved**	Reserved.
[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.
[2:0]	**WEEKDAY**	**Day of the Week Register** 000 = Sunday. 001 = Monday. 010 = Tuesday. 011 = Wednesday. 100 = Thursday. 101 = Friday. 110 = Saturday. 111 = Reserved.

Note: FREQADJ's counter will be reset for start to Compensatie when writing RTC_TIME, RTC_CAL, RTC_WEEKDAY. Imply RTC Time will be restarted.

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	
[31:24]	**Reserved**	Reserved.
[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.
[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.
[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.

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.
[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.
[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.
[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.

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.
[0]	**LEAPYEAR**	**Leap Year Indication (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.
[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.
[1]	**TICKIF**	**RTC Time Tick Interrupt Flag** 0 = Tick condition did not occur. 1 = Tick condition occurred. **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.
[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.

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.
[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.
[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
RTCCKSEL	Reserved			GAIN			Reserved

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**RTCCKSEL**	**RTC Clock Source Selection** 0 = Clock source from 32 kHz crystal or external XI_32K input . 1 = Clock source from internal low speed RC oscillator (LIRC38K).
[6:4]	**Reserved**	Reserved.
[3:1]	**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. 000 = L0 mode (ESR=35K; CL =25pF). 001 = L1 mode (ESR=35K; CL =25pF). 010 = L2 mode (ESR=35K; CL =25pF). 011 = L3 mode (ESR=70K; CL =25pF). 100 = L4 mode (ESR=70K; CL =25pF). 101 = L5 mode (ESR=70K; CL =25pF). 110 = L6 mode (ESR=90K; CL =25pF). 111 = L7 mode (ESR=90K; CL =25pF).
[0]	**Reserved**	Reserved.

RTC Daylight Saving Time Control Register (RTC_DSTCTL)

Register	Offset	R/W	Description	Reset Value
RTC_DSTCTL	RTC_BA+0x110	R/W	RTC Daylight Saving Time Control Register	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**DSBAK**	**Daylight Saving Back** 0= Daylight Saving Change is not performed. 1= Daylight Saving Change is performed.
[1]	**SUBHR**	**Subtract 1 Hour** 0 = No effect. 1 = Indicates RTC hour digit has been subtracted one hour for winter time change.
[0]	**ADDHR**	**Add 1 Hour** 0 = No effect. 1 = Indicates RTC hour digit has been added one hour for summer time change.

6.12 EPWM Generator and Capture Timer (EPWM)

6.12.1 Overview

The chip provides two EPWM generators — EPWM0 and EPWM1. Each EPWM supports 6 channels of EPWM output or input capture. There is a 12-bit prescaler to support flexible clock to the 16-bit EPWM counter with 16-bit comparator. The EPWM counter supports up, down and up-down counter types. EPWM uses comparator compared with counter to generate events. These events use to generate EPWM pulse, interrupt and trigger signal for EADC/DAC to start conversion.

The EPWM generator supports two standard EPWM 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 EPWM pulse with 12-bit dead-time generator and another free trigger comparator to generate trigger signal for EADC. For EPWM output control unit, it supports polarity output, independent pin mask and brake functions.

The EPWM generator also supports input capture function. It supports latch EPWM 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.

6.12.2 Features

6.12.2.1 EPWM Function Features

- Supports maximum clock frequency up to maximum PLL frequency
- Supports up to two EPWM modules, each module provides 6 output channels
- Supports independent mode for EPWM output/Capture input channel
- Supports complementary mode for 3 complementary paired EPWM output channel
  - Dead-time insertion independent control with 12-bit resolution
  - Synchronous function for phase control
  - Two compared values during one period
- Supports 12-bit prescaler from 1 to 4096
- Supports 16-bit resolution EPWM 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 EPWM pin
- Supports brake function
  - Brake source from pin, analog comparator and system safety events (clock failed, SRAM parity error, Brown-out detection and CPU lockup).
  - Noise filter for brake source from pin
  - Leading edge blanking (LEB) function for brake source from analog comparator
  - 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:
  - EPWM counter matches 0, period value or compared value
  - Brake condition happened
- Supports trigger EADC/DAC on the following events:
  - EPWM counter matches 0, period value or compared value
  - EPWM counter match free trigger comparator compared value (only for EADC)
  - Support EPWM trigger EADC event prescaler feature
- Supports PDMA transfer for Interrupt Flag Accumulator function
- Supports EPWM output accumulator stop counter mode
- Supports Fault Detect Function.
- Supports External Pin Trigger function

6.12.2.2 Capture Function Features

- Supports 3-bit capture input noise filter
- 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 EPWM all channels

6.12.3 Block Diagram

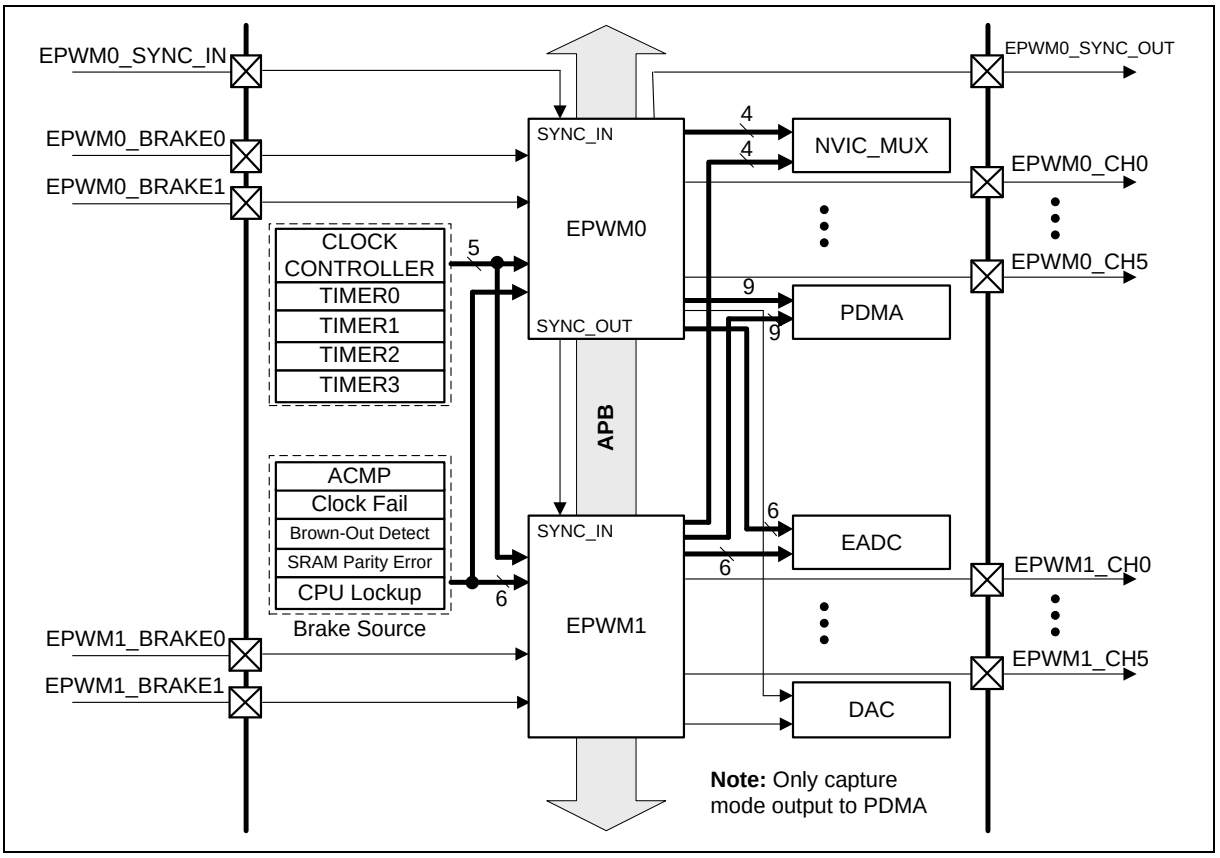


Figure 6.12-1 EPWM Generator Overview Block Diagram

EPWM Clock frequency can be set equal or double to PCLK frequency as Figure 6.12-2. For the detailed register setting, please refer to Table 6.12-1. Each EPWM generator has three clock source inputs, each clock source can be selected from EPWM Clock or four TIMER trigger EPWM outputs as Figure 6.12-3 by ECLKSRC0 (EPWM_CLKSRC[2:0]) for EPWM_CLK0, ECLKSRC2 (EPWM_CLKSRC[10:8]) for EPWM_CLK2 and ECLKSRC4 (EPWM_CLKSRC[18:16]) for EPWM_CLK4.

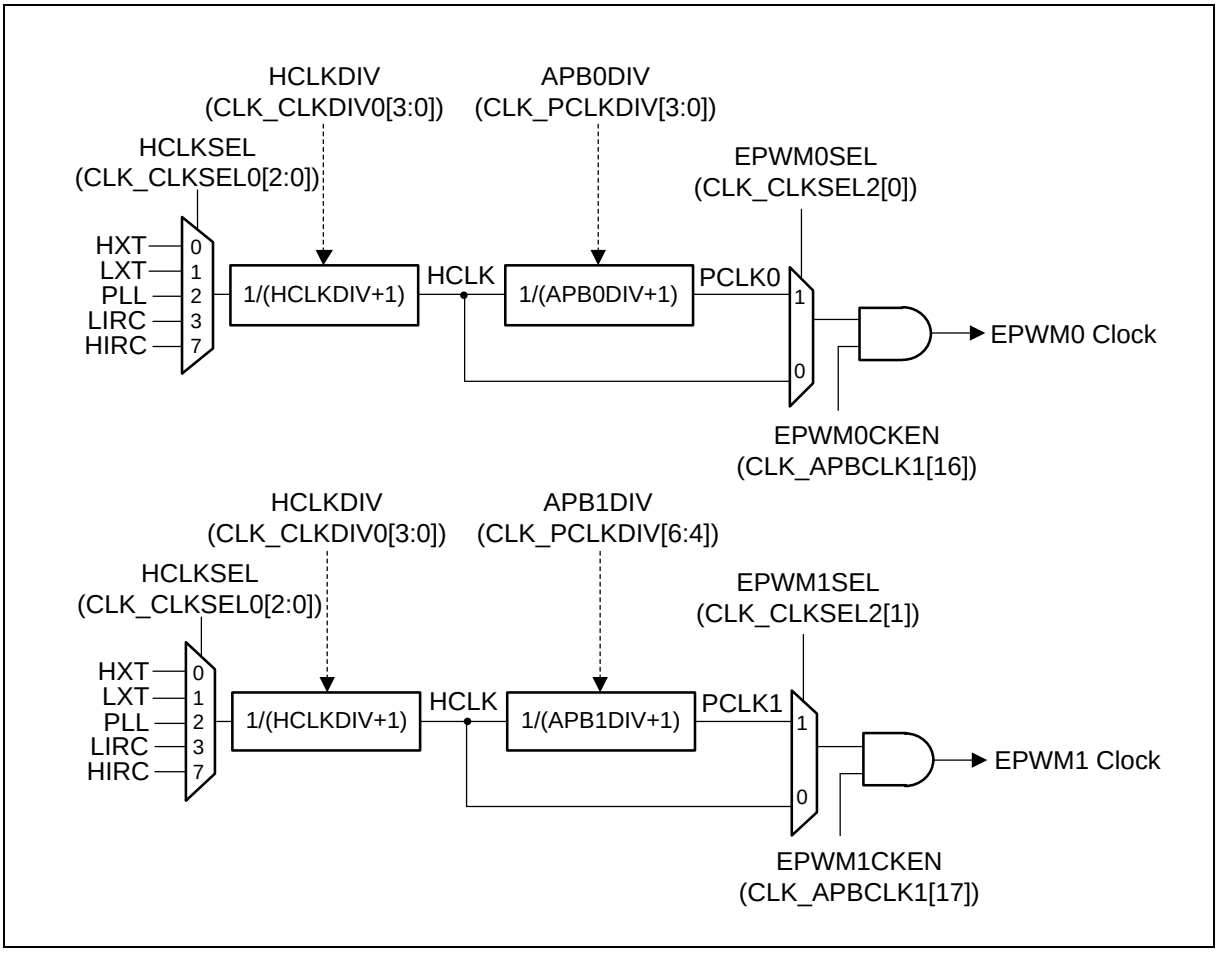


Figure 6.12-2 EPWM Clock Source Control

Frequency Ratio PCLK:EPWM Clock	HCLK	PCLK	EPWM Clock	HCLKSEL CLK_CLKSEL0[2:0]	HCLKDIV CLK_CLKDIV0[3:0]	APBnDIV (CLK_PCLKDIVn [2+4n:4n]), N Denotes 0 Or 1	EPWMnSEL (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	HCLK	HCLK/ 2	HCLK	Don't care	Don't care	1	0

Table 6.12-1 EPWM Clock Source Control Registers Setting Table

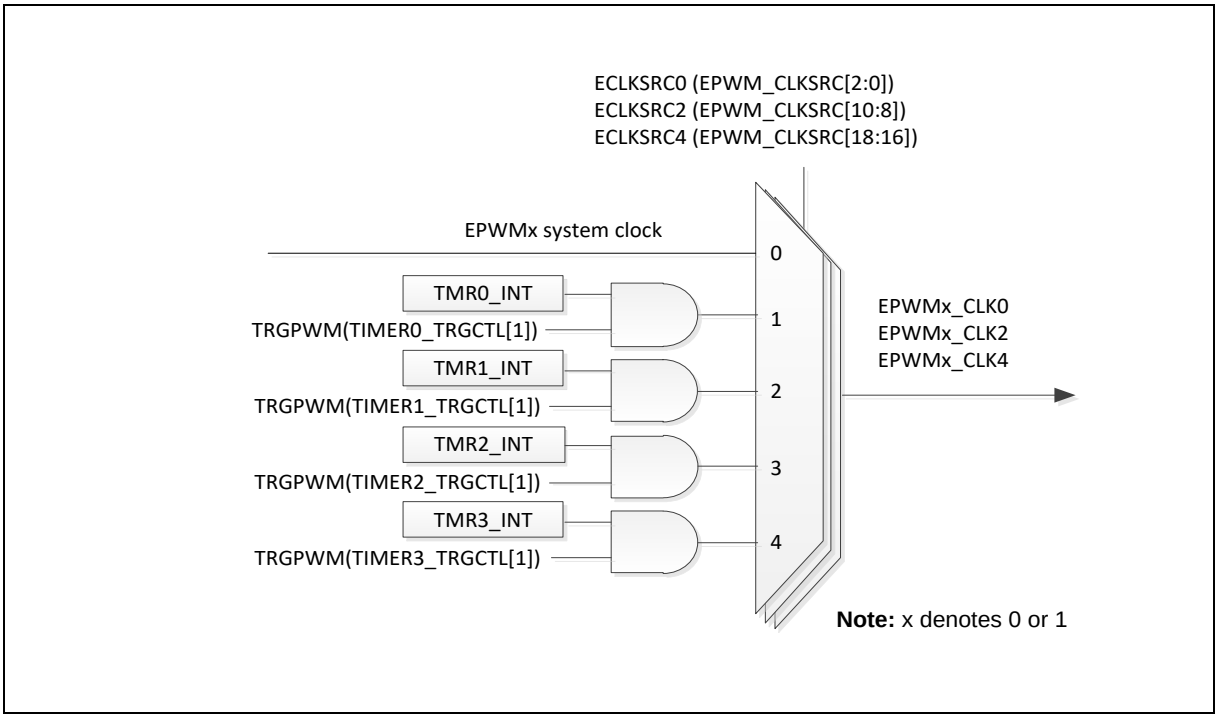


Figure 6.12-3 EPWM Clock Source Control

Figure 6.12-4 and Figure 6.12-5 illustrate the architecture of EPWM independent mode and complementary mode. No matter independent mode or complementary mode, paired channels' (EPWM_CH0 and EPWM_CH1, EPWM_CH2 and EPWM_CH3, EPWM_CH4 and EPWM_CH5) counters both come from the same clock source and prescaler. When counter count to 0, PERIOD (EPWM_PERIODn[15:0]) or equal to comparator, events will be generated. These events are passed to corresponding generators to generate EPWM pulse, interrupt signal and trigger signal for EADC/DAC to start conversion. Output control is used to change EPWM 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.

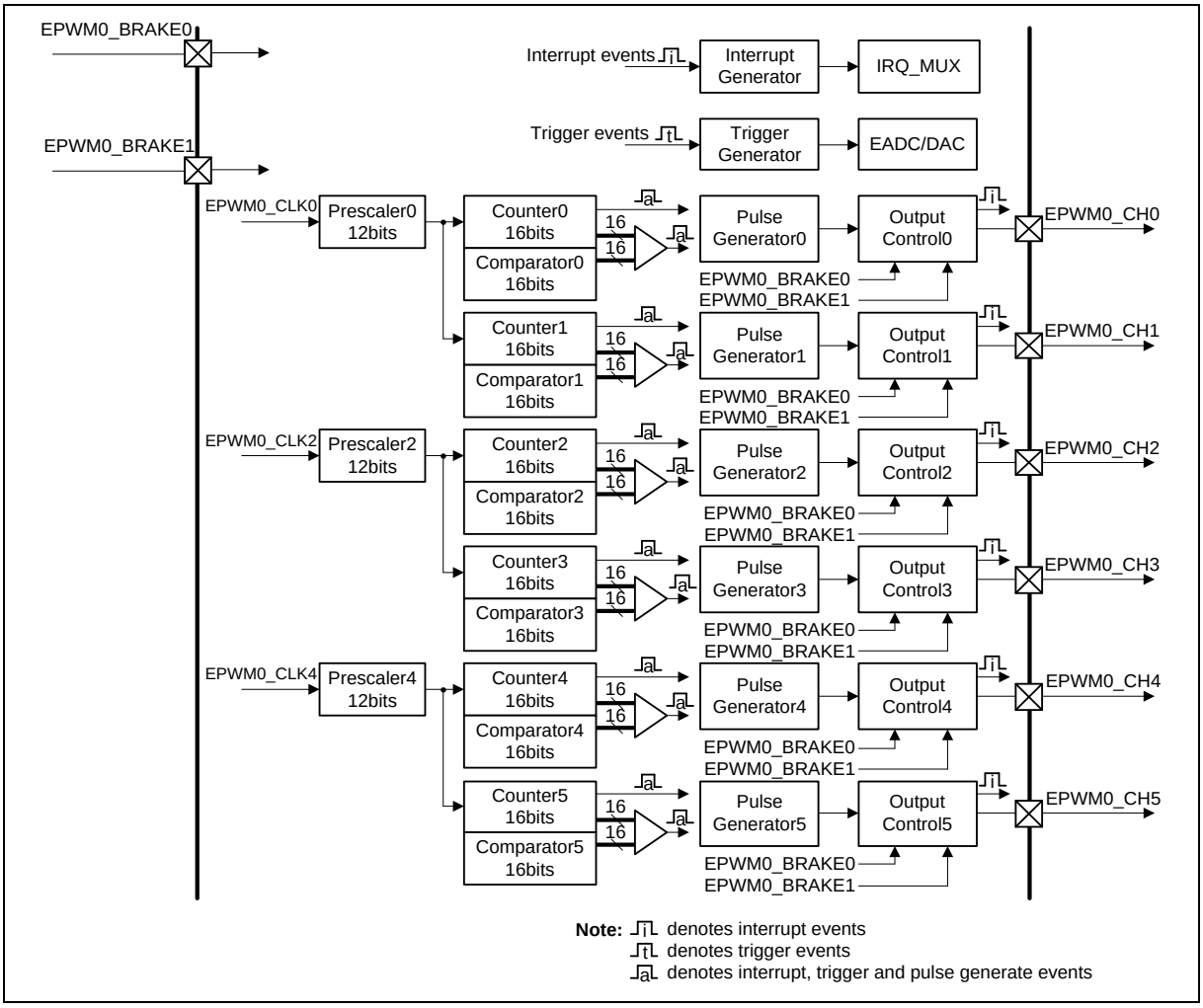


Figure 6.12-4 EPWM Independent Mode Architecture Diagram

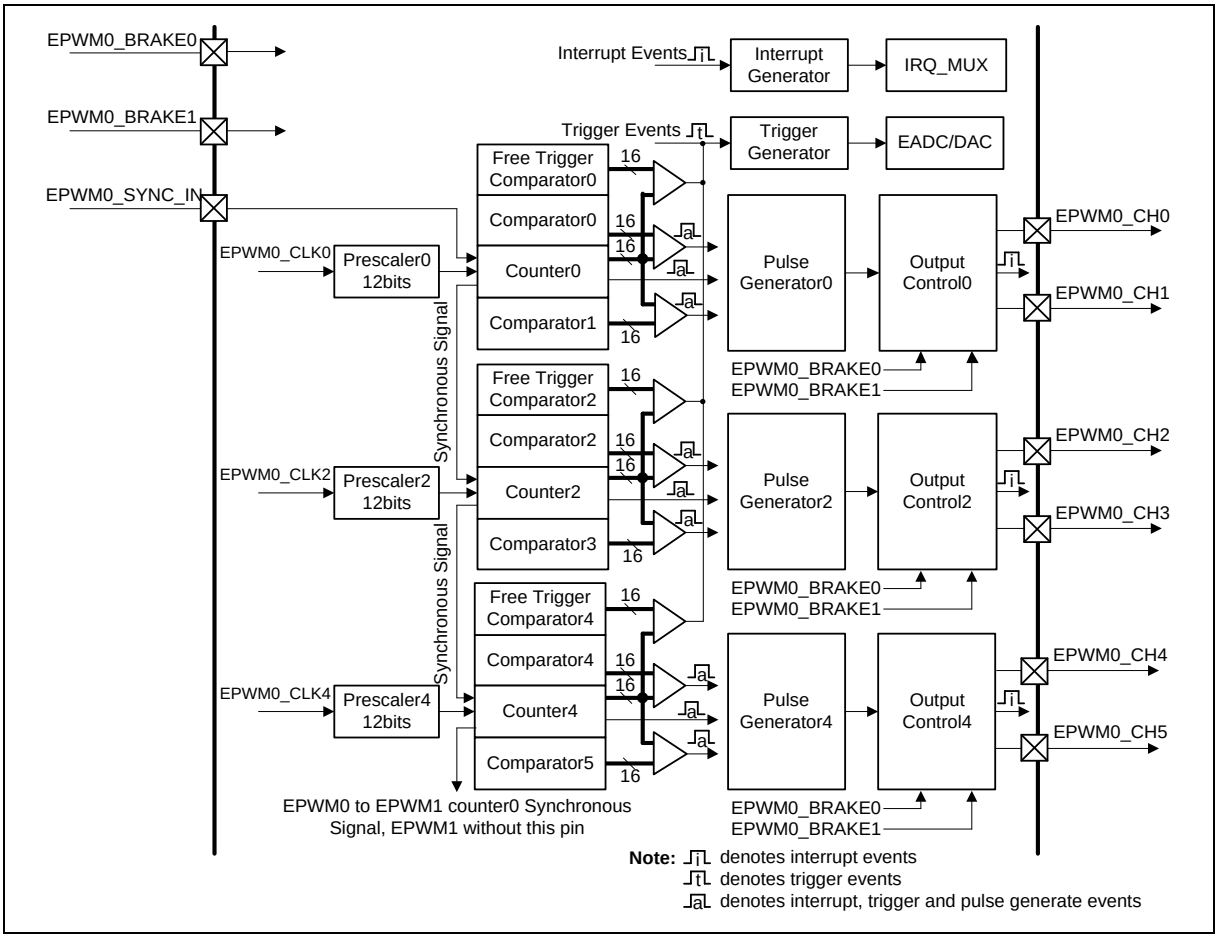


Figure 6.12-5 EPWM Complementary Mode Architecture Diagram

6.12.4 Basic Configuration

6.12.4.1 EPWM0 Basic Configuration

- Clock Source Configuration
  - Select the source of EPWM0 peripheral clock on EPWM0SEL (CLK_CLKSEL2[0])
  - Enable EPWM0 peripheral clock in EPWM0CKEN (CLK_APBCLK1[16]).
- Reset Configuration
  - Reset EPWM0 in EPWM0RST SYS_IPRST2[16]
- EPWM1 Basic Configuration
- Clock Source Configuration
  - Select the source of EPWM1 peripheral clock on EPWM1SEL (CLK_CLKSEL2[1])
  - Enable EPWM1 peripheral clock in EPWM1CKEN (CLK_APBCLK1[17]).
- Reset Configuration
  - Reset EPWM1 in EPWM1RST (SYS_IPRST2[17])

6.12.5 Functional Description

6.12.5.1 EPWM Prescaler

The EPWM prescaler is used to divide clock source, prescaler counting CLKPSC +1 times, EPWM counter only count once. The prescale double buffer is setting by CLKPSC (EPWM_CLKPSCn[11:0], n = 0~5) bits.Figure 6.12-6 is an example of EPWM channel 0 prescale waveform. The prescale counter will reload CLKPSC at the begin of the next prescale counter down-count.

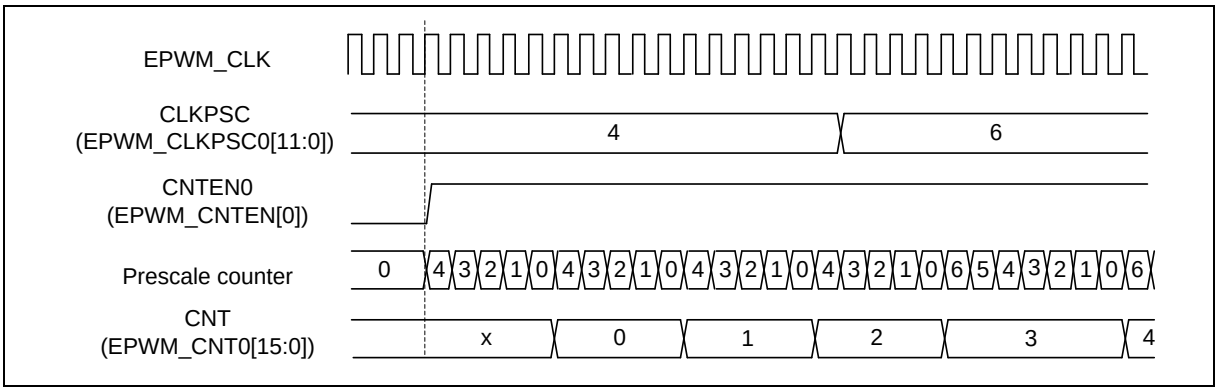


Figure 6.12-6 EPWM_CH0 Prescaler Waveform in Up Counter Type

6.12.5.2 EPWM Counter

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

For EPWM channel0, CNT(EPWM_CNT0[15:0]) can clear to 0x00 by CNTCLR0 (EPWM_CNTCLR[0]). CNT will be cleared when prescale counter count to 0, and CNTCLR0 will be set 0 by hardware automatically.

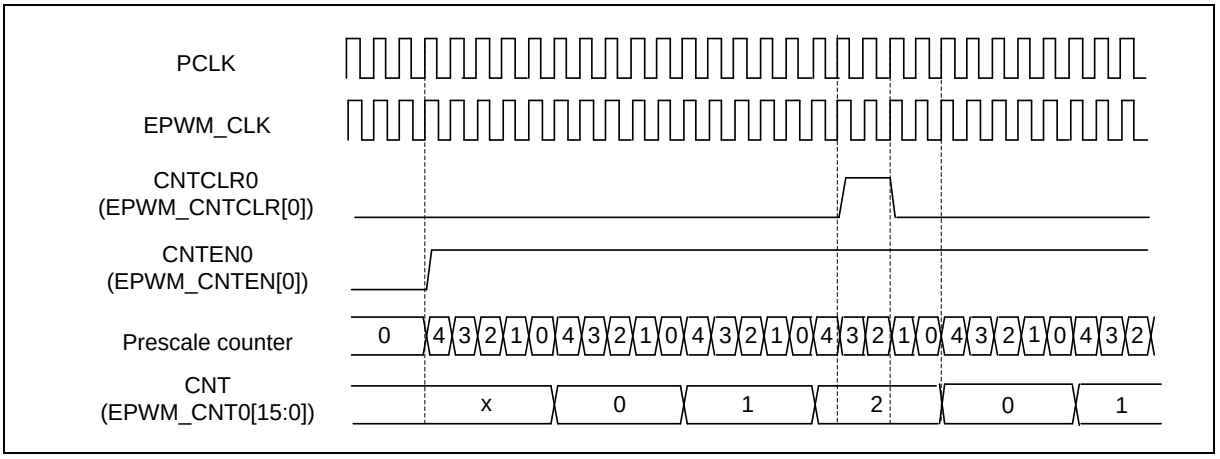


Figure 6.12-7 EPWM Counter Waveform when Setting Clear Counter

6.12.5.3 Up Counter Type

When EPWM counter is set to up counter type, CNTTYPEn (EPWM_CTL1[2n+1:2n], n = 0,1..5) is 0x0, it starts up-counting from 0 to PERIOD (EPWM_PERIODn[15:0], where n denotes channel number) to complete a EPWM period. The current counter value can be read from CNT (EPWM_CNTn[15:0]) bits. EPWM generates zero point event when the counter counts to 0 and prescale counts to 0. EPWM generates period point event when the counter counts to PERIOD and prescale counts to 0. Figure 6.12-8 shows an example of up counter, wherein

EPWM period time = (PERIOD+1) *(CLKPSC+1)* EPWMx_CLK.

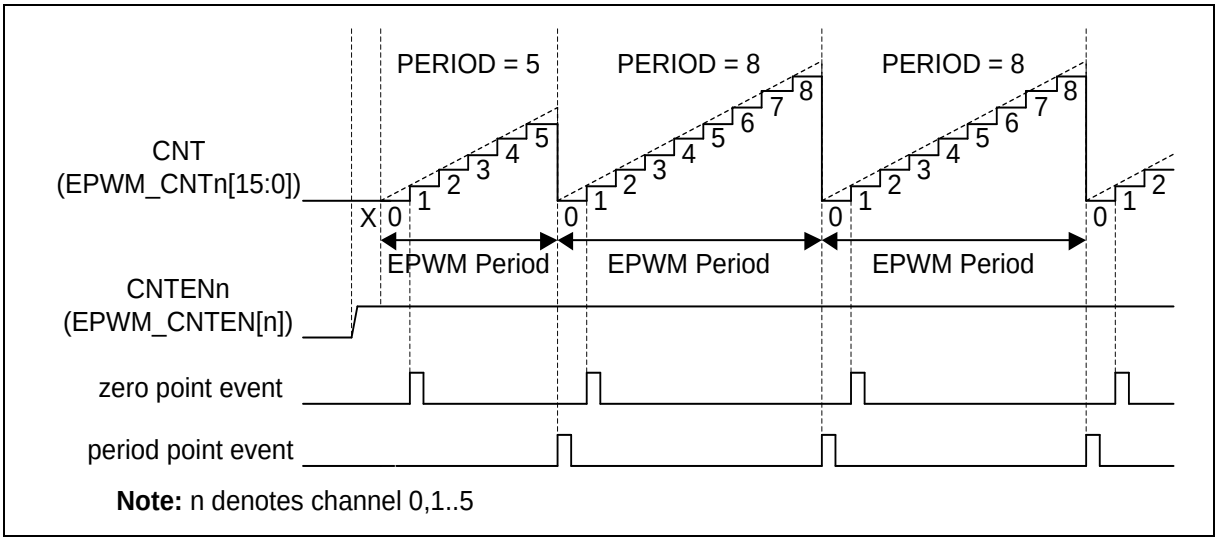


Figure 6.12-8 EPWM Up Counter Type

6.12.5.4 Down Counter Type

When EPWM counter is set to down counter type, CNTTYPEn (EPWM_CTL1[2n+1:2n], n = 0,1..5) is 0x1, it starts down-counting from PERIOD to 0 to complete a EPWM period. The current counter value can be read from CNT (EPWM_CNTn[15:0]) bits. EPWM generates zero point event when the counter counts to 0 and prescale counts to 0. EPWM generates period point event when the counter counts to PERIOD and prescale counts to 0. Figure 6.12-9 shows an example of down counter, wherein

EPWM period time = (PERIOD+1) * (CLKPSC+1) * EPWMx_CLK.

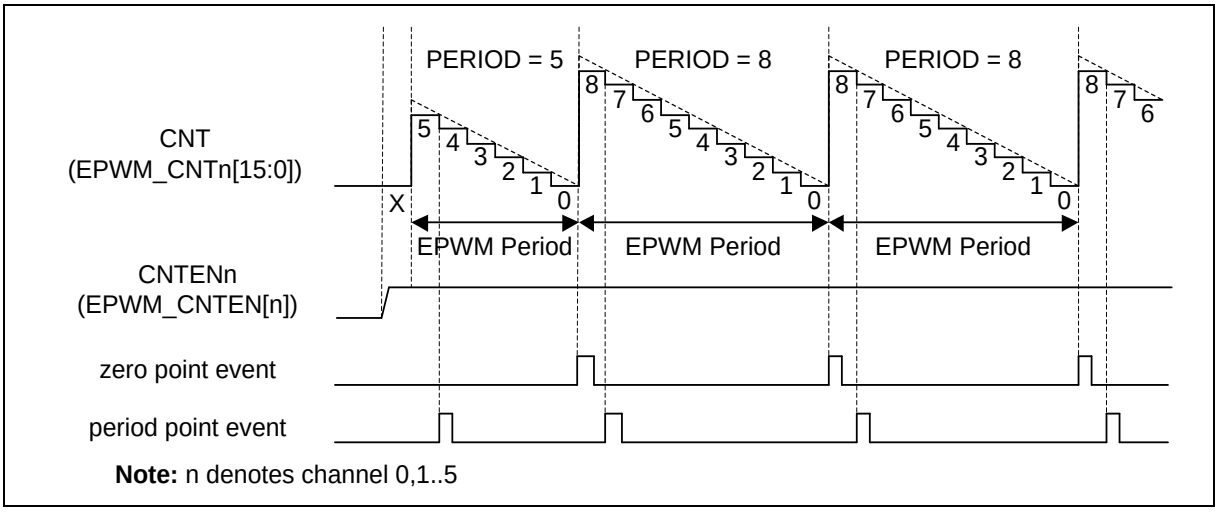


Figure 6.12-9 EPWM Down Counter Type

6.12.5.5 Up-Down Counter Type

When EPWM counter is set to up-down count type, CNTTYPEn (EPWM_CTL1[2n+1:2n], n = 0,1..5) is 0x2, it starts counting-up from 0 to PERIOD and then starts counting down to 0 to complete a EPWM period. The current counter value can be read from CNT (EPWM_CNTn[15:0]) bits. EPWM generates

zero point event when the counter counts to 0 and prescale counts to 0. EPWM generates center point event which is equal to period point event when the counter counts to PERIOD. Figure 6.12-10 shows an example of up-down counter, wherein

EPWM period time =(2*PERIOD) * (CLKPSC+1) * EPWMx_CLK.

The DIRF (EPWM_CNTn[16]) bit is counter direction indicator flag, where high is up counting, and low is down counting.

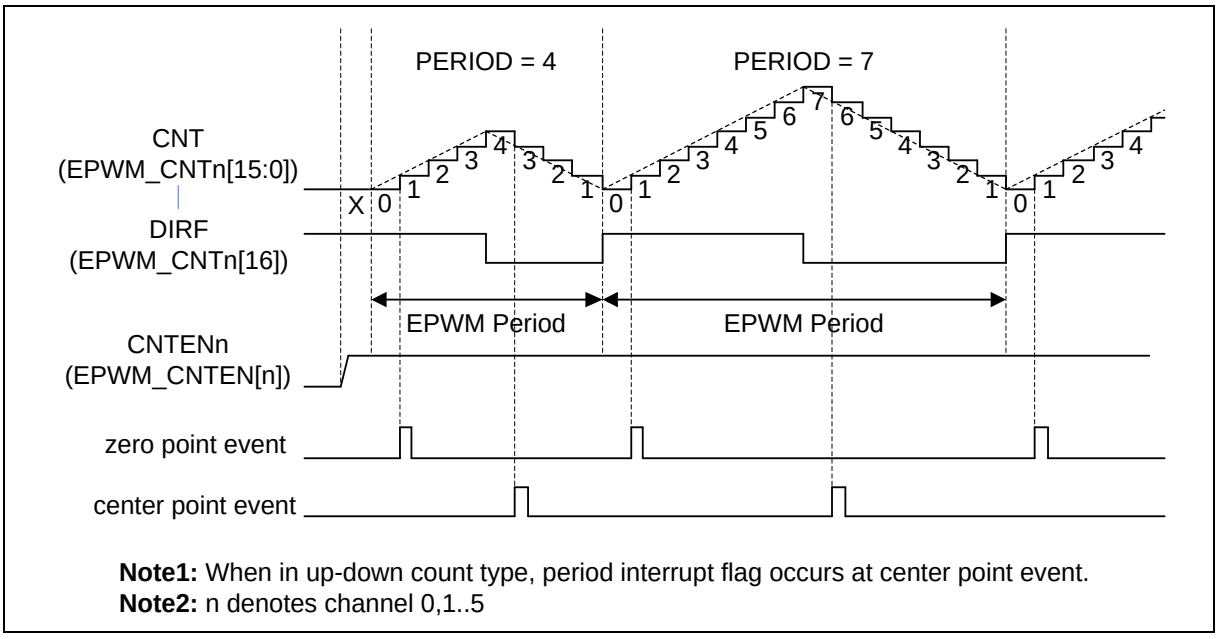
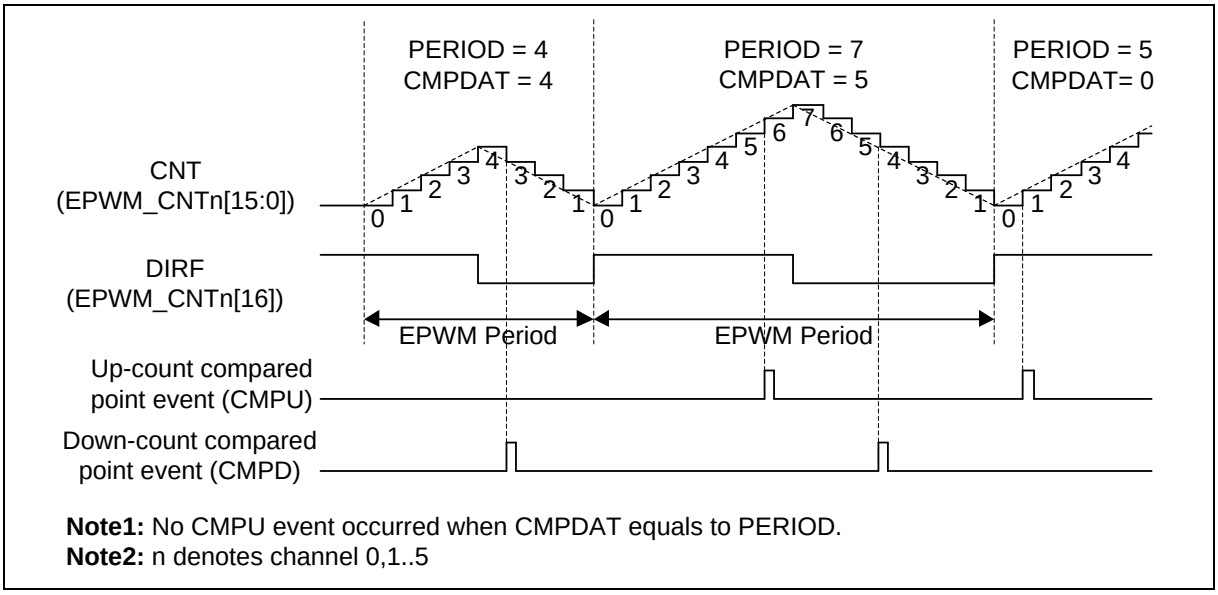


Figure 6.12-10 EPWM Up-Down Counter Type

6.12.5.6 EPWM Comparator

There are two kinds of comparator registers : one is EPWM_CMPDATn(n = 0,1..5), and the other is EPWM_FTCMPDATn_m(n = 0,2,4, m = 1,3,5) register. EPWM_CMPDATn is a basic comparator register of EPWM 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 EPWM_CMPDATn and EPWM_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 EPWM_CMPDAT0 register, EPWM generates a compared point event and uses the event to generate EPWM pulse, interrupt or use to trigger EADC/DAC. In up-down counter type, two events will be generated in a EPWM period as shown in Figure 6.12-11. The CMPU is up count compared point event and CMPD is down count compared point event.



Note1: No CMPU event occurred when CMPDAT equals to PERIOD.
Note2: n denotes channel 0,1..5

Figure 6.12-11 EPWM Compared point Events in Up-Down Counter Type

EPWM_FTCMPDATn_m is a free trigger comparator register. Each complementary paired channel only supports one free trigger comparator. The value of EPWM_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 EPWM_FTCMPDATn_m register, FTCMDn (EPWM_FTCI[10:8], n=0,2,4) indicator is set in down count type and FTCMUn (EPWM_FTCI[2:0], n=0,2,4) indicator is set in up count type. In addition, EPWM generates an event and only uses to trigger EADC.

6.12.5.7 EPWM 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 asynchronously operation problem due to software and hardware asynchronism.

The EPWM provides PBUF(EPWM_PBUFn[15:0]) as the active EPWM_PERIODn buffer register, CMPBUF(EPWM_CMPBUFn[15:0]) as the active EPWM_CMPDATn buffer register, FTCMBUF(EPWM_FTCBUFn_m[15:0]) as the active EPWM_FTCMPDATn_m buffer register and CPSCBUF(EPWM_CPSCBUFn[15:0]) as the active EPWM_CLKPSCn buffer register. The concept of double buffering is used in loading modes, which are described in the following sections. For example, as shown Figure 6.12-12, in period loading mode, writing PERIOD(EPWM_PERIODn[15:0]), CMPDAT(EPWM_CMPDATn[15:0]) and FTCMP(EPWM_FTCMPDATn_m[15:0]) buffers through software, EPWM will load new values to their buffer PBUF(EPWM_PBUFn[15:0]), CMPBUF(EPWM_CMPBUFn[15:0]) and FTCMPBUF(EPWM_FTCBUFn_m[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.

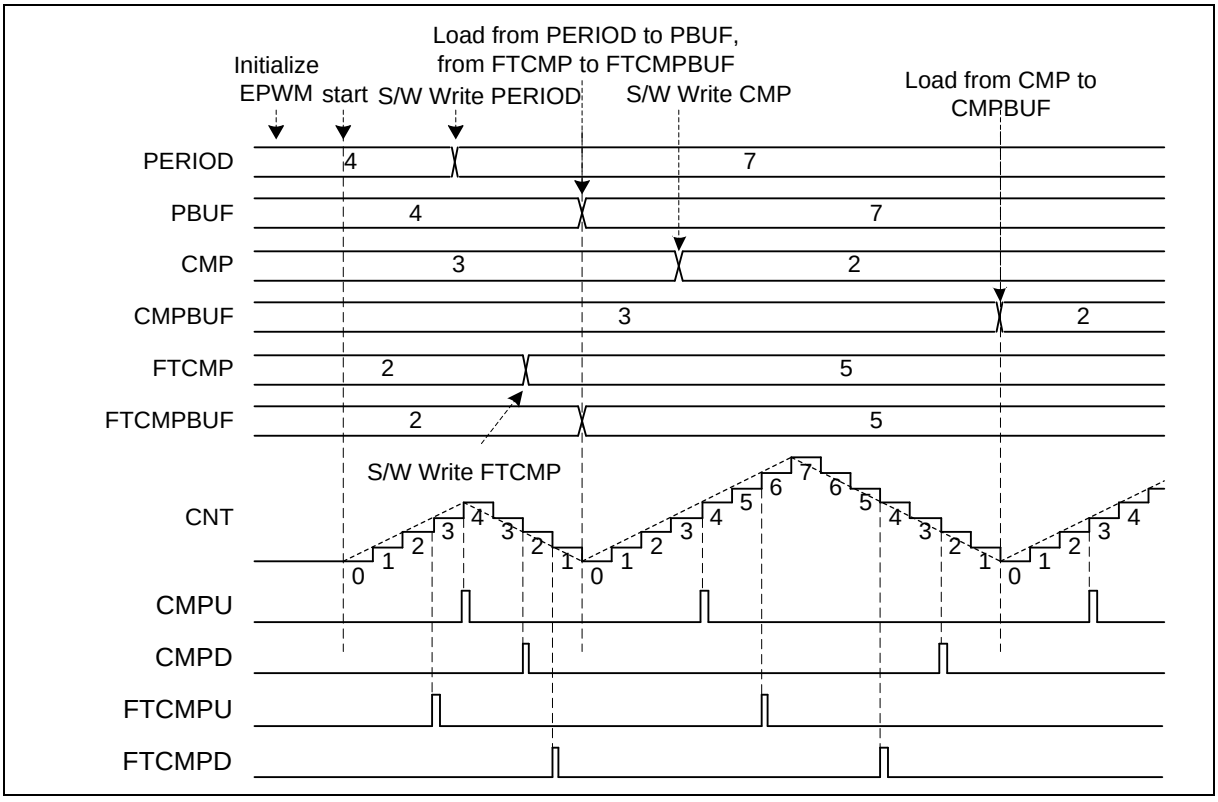


Figure 6.12-12 EPWM Double Buffering Illustration

6.12.5.8 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 EPWM_CTL0 register are set to 0, EPWM operates at period Loading mode. In period Loading mode, CLKPSC(EPWM_CLKPSCn_m[11:0]), PERIOD(EPWM_PERIODn[15:0]), CMP(EPWM_CMPDATn[15:0]) and FTCMPDAT (EPWM_FTCMPDATn_m[15:0]) will all load to their active CPSCBUF(EPWM_CPSCBUFn[11:0]), PBUF(EPWM_PBUFn[15:0]), CMPBUF(EPWM_CMPBUFn[15:0]) and FTCMPBUF(EPWM_FTCBUFn_m[15:0]) registers while each period is completed. For example, after EPWM counter up counts from 0 to PERIOD in the up-counter operation or down counts from PERIOD to 0 in the down-counter operation or counts up from 0 to PERIOD and then counts down to 0 in the up-down counter operation.

Figure 6.12-13 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. CMP also follows this rule. The following describes steps sequence of Figure 6.12-13. User can know the PERIOD and CMP update condition, by watching EPWM period and CMPU event.

1. Software writes CMP DATA1 to CMP at point 1.
2. Hardware loads CMP DATA1 to CMPBUF at the end of EPWM 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 EPWM 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 EPWM period at point 6.

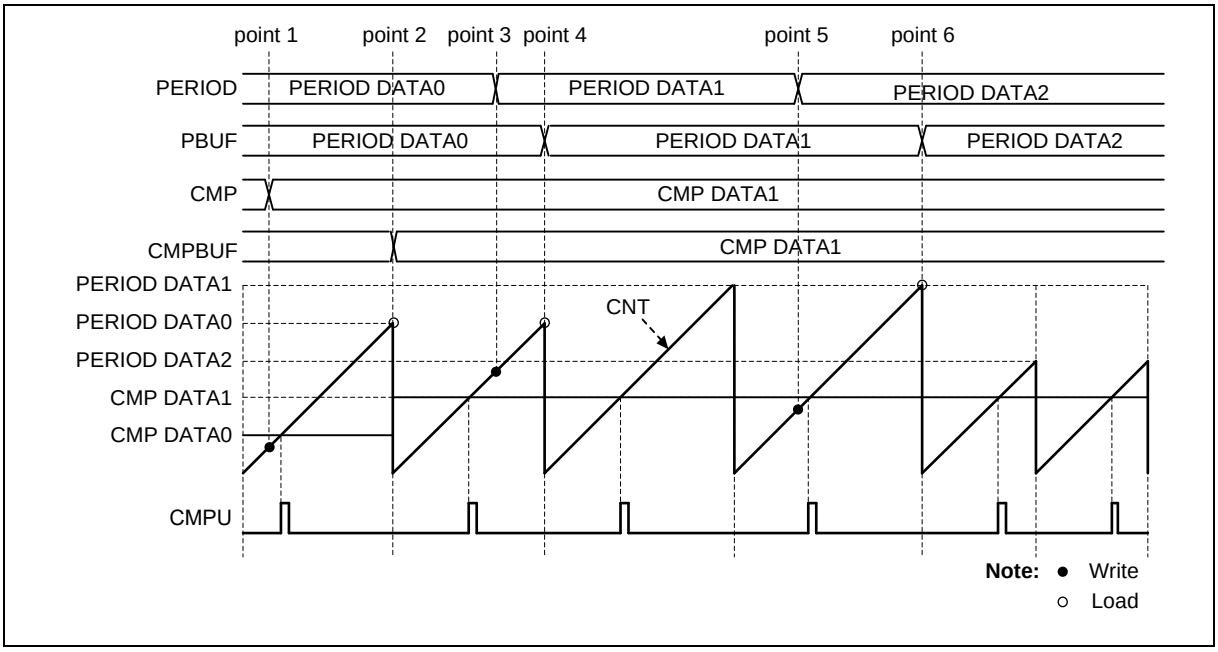


Figure 6.12-13 Period Loading in Up-Count Mode

6.12.5.9 Immediately Loading Mode

If the IMMLDENn (EPWM_CTL0[21:16]) bit is set to 1, EPWM operates at immediately loading mode. In immediately loading mode, when user update CLKPSC(EPWM_CLKPSCn_m[11:0]), PERIOD(EPWM_PERIODn[15:0]), CMP(EPWM_CMPDATn[15:0]) or FTCMPDAT (EPWM_FTCMPDATn_m[15:0]), CLKPSC, PERIOD, CMP or FTCMPDAT will be load to active CPSCBUF(EPWM_CPSCBUFn_m[15:0]), PBUF(EPWM_PBUFn[15:0]), CMPBUF (EPWM_CMPBUFn[15:0]) or FTCMPBUF (EPWM_FTCBUF[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 0, the flag CNTMAXF(EPWMx_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 6.12-14 shows an example and its steps sequence is described below.

1. Software writes CMP DATA1 and hardware immediately loading CMP 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.

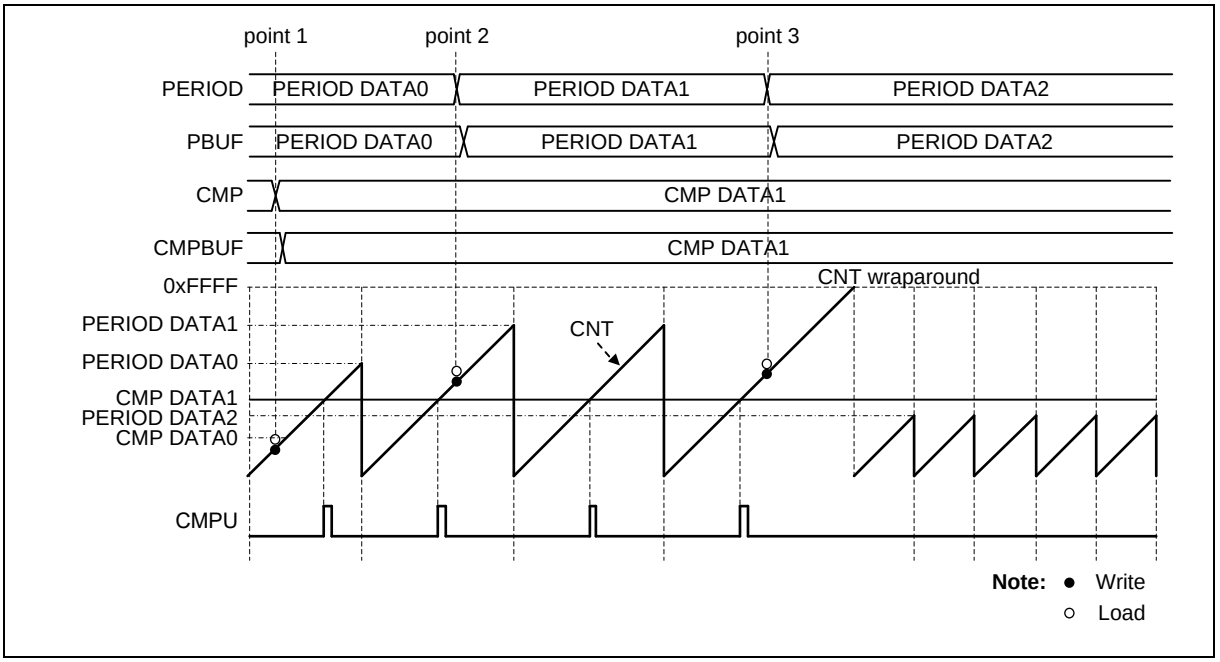


Figure 6.12-14 Immediately Loading in Up-Count Mode

6.12.5.10 Window Loading Mode

When the WINLDENn (EPWM_CTL0[13:8]) bit is set to 1, EPWM operates at window loading mode. In Window loading mode, CLKPSC(EPWM_CLKPSCn_m[11:0]), PERIOD(EPWM_PERIODn[15:0]), CMP(EPWM_CMPDATn[15:0]) and FTCMPDAT (EPWM_FTCMPDATn_m[15:0]) will all load to their active CPSCBUF(EPWM_CPSCBUFn[11:0]), PBUF(EPWMPBUFn[15:0]), CMPBUF(EPWM_CMPBUFn[15:0]) and FTCMPBUF(EPWM_FTCBUFn_m[15:0]) registers while each period is completed, but CMPBUF loading are valid only when load window is opened. Every channel n's load window is opened by setting the corresponding LOADn (EPWM_LOAD[5:0]) to 1, and hardware will close the window at the end of EPWM period. Figure 6.12-15 shows an example and its steps sequence is described below.

1. Software writes CMP DATA1 at point 1, and the load window is not opened at this period so CMP will not load to CMPBUF.
2. Software writes LOAD to open the load window at point2.
3. Software writes CLKPSC DATA1 and PERIOD DATA1 at point 3.
4. At point 4, load window has been opened, hardware loads CLKPSC DATA1, PERIOD DATA1 and CMP DATA1 to their buffer and closes the load window at the end of EPWM period.
5. Software writes CLKPSC DATA2 and PERIOD DATA2 at point 5.
6. Hardware loads CLKPSC DATA2 and PERIOD DATA2 to their buffer at the end of EPWM period at point 6.
7. Software writes CLKPSC DATA3 and 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 EPWM period at point 9.

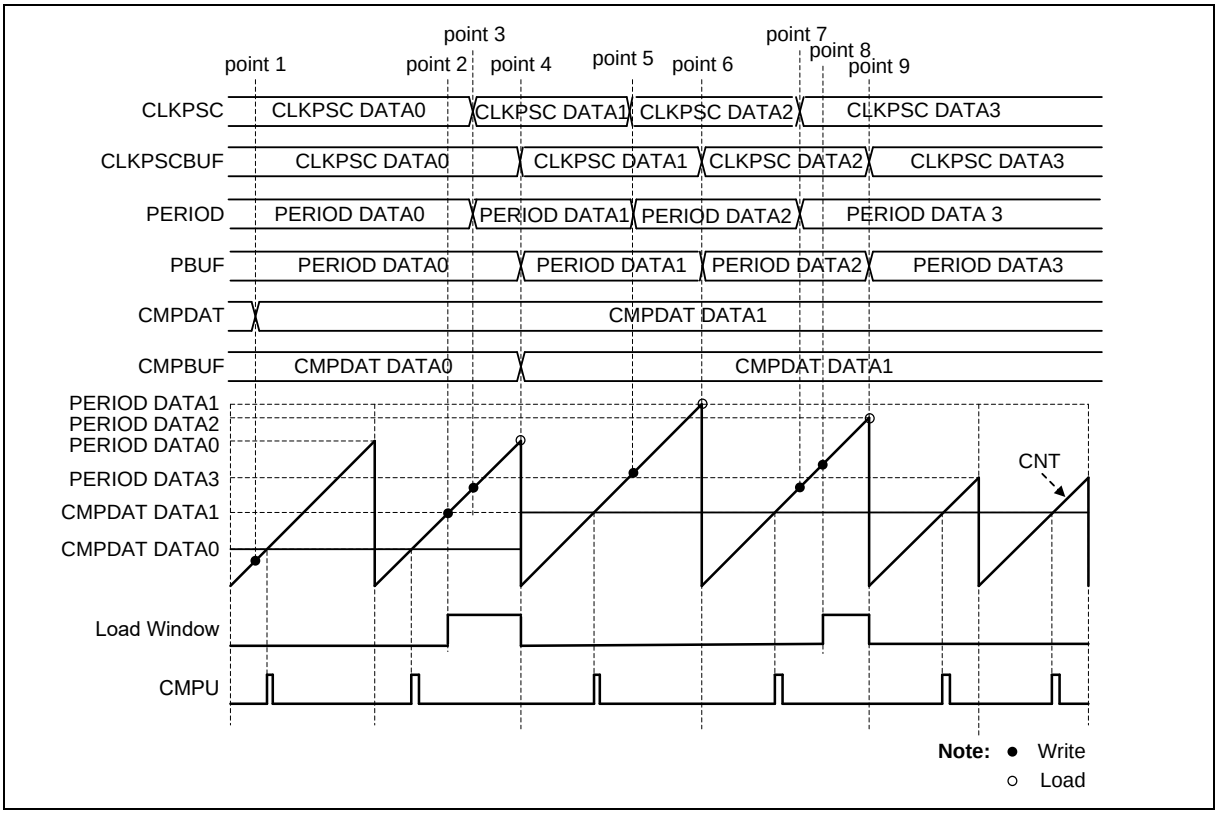


Figure 6.12-15 Window Loading in Up-Count Mode

6.12.5.11 Center Loading Mode

When the CTRLDn (EPWM_CTL0[5:0]) bit is set to 1 and EPWM counter is set to up-down count type, CNTTYPEn (EPWM_CTL1[2n+1:2n], n = 0,1..5) is 0x2, EPWM operates at center loading mode. In center loading mode, CMP(EPWM_CMPDATn[15:0]) and FTCMPDAT (EPWM_FTCMPDATn_m[15:0]) will load to active CMPBUF(EPWM_CMPBUFn[15:0]) and FTCMPBUF(EPWM_FTCBUFn_m[15:0]) buffer in center of each period, that is, counter counts to PERIOD. CLKPSC(EPWM_CLKPSCn_m[11:0]) and PERIOD(EPWM_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(EPWM_CMPDATn[15:0]) will load to active CMPBUF buffer in center of each period, but it is valid only at the interval of load window. Figure 6.12-16 shows an example and its steps sequence is described below.

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

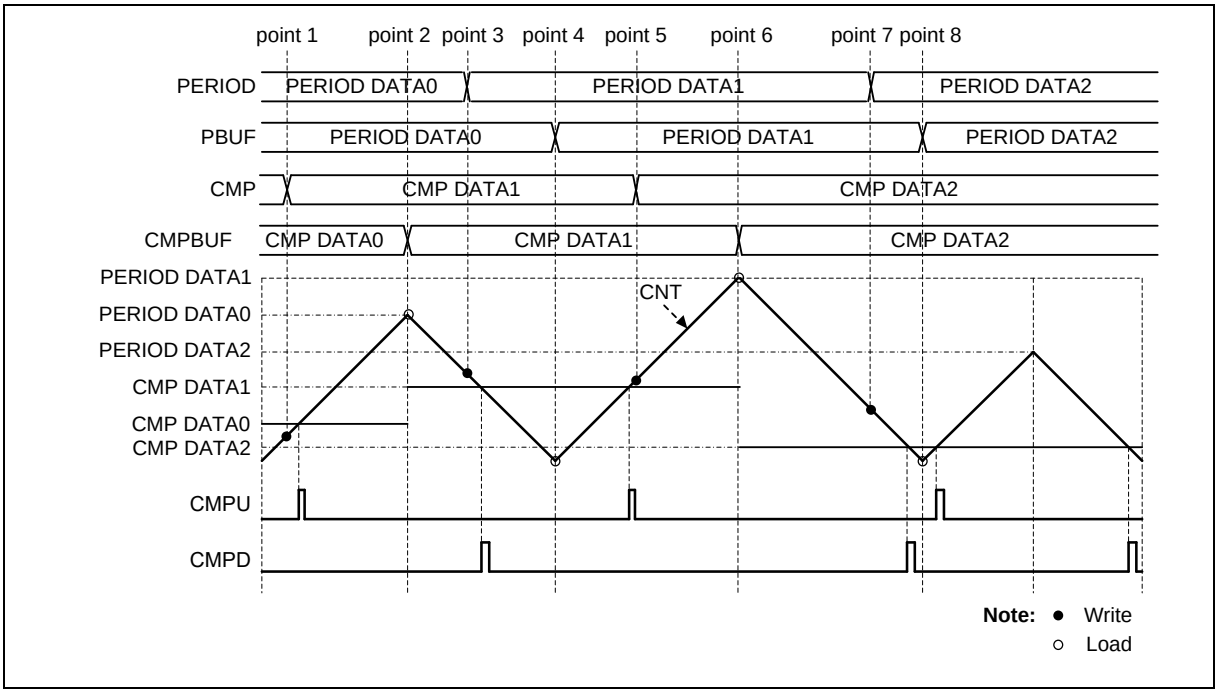


Figure 6.12-16 Center Loading in Up-Down-Count Mode

6.12.5.12 EPWM Counter Operation Mode

The EPWM counter supports two operation modes: One-shot mode and Auto-reload mode. EPWM counter will operate in One-shot mode if CNTMODEn (EPWM_CTL1[21:16]) bit is set to 1, and operate in Auto-reload mode if set to 0.

In One-shot mode, EPWM_CMPDATn and EPWM_PERIODn registers should be written first and then set CNTENn (EPWM_CNTEN[5:0]) bit as 1 to enable EPWM prescaler and counter start running. After EPWM counter counted a period, counter value will keep 0.

User can re-start next one-shot by writing new value to CMP(EPWM_CMPDATn[15:0]) bits. If one-shot counter still running, to update EPWM_CMPDATn register will cause next one-shot as continuous one-shot. Besides, to write EPWM_CMPDATn register twice under continuous one-shot operation, latest value in EPWM_CMPDATn register is valid at next one-shot period and only generate one-shot pulse once. Moreover, if user wants to clear counter within one-shot operation and starts next one-shot, user should monitor counter value to check counter has cleared and then writes EPWM_CMPDATn register. Figure 6.12-17 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 CMP DATA1 which is equal to CMP DATA0 at point 2 and hardware immediately loading CMP DATA1 to CMPBUF, this event also trigger one-shot.
3. Software writes CMP DATA2 and re-trigger next one-shot (continuous one-shot) at point 3.
4. Software writes CMP DATA3 to cover CMP DATA2 and re-trigger next one-shot at point 4.
5. Period loading CMP DATA3 to CMPBUF at point 5.
6. There are no new CMP write in the previous period, and the counter value is kept as 0 at point 6.

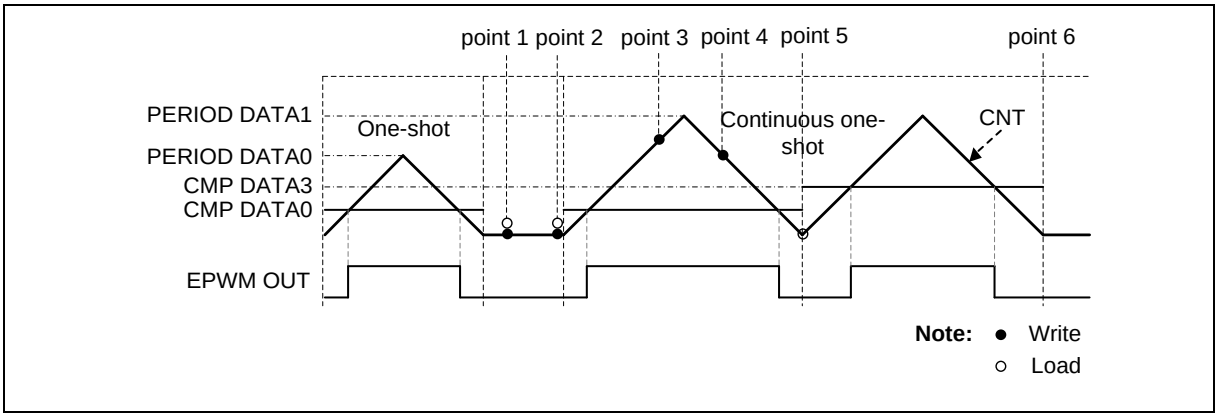


Figure 6.12-17 EPWM One-shot Mode Output Waveform

In Auto-reload mode, EPWM_CMPDATn and EPWM_PERIODn registers should be written first and then the CNTENn(EPWM_CNTEN[n]) bit is set to 1 to enable EPWM prescaler and start to run counter. The value of CLKPSC(EPWM_CLKPSCn_m[11:0]), PERIOD(EPWM_PERIODn[15:0]) and CMP(EPWM_CMPDATn[15:0]) will auto reload to their active buffer according different loading mode. If PERIOD(EPWM_PERIODn[15:0]) is set to 0, EPWM counter will be set to 0.

6.12.5.13 EPWM Pulse Generator

The EPWM pulse generator uses counter and comparator events to generate EPWM 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 EPWM waveform to do nothing (X), set Low (L), set High (H) or toggle (T) by setting the EPWM_WGCTL0 and EPWM_WGCTL1 registers. Using these points can easily generate asymmetric EPWM pulse or variant waveform as shown in Figure 6.12-18. In the figure, EPWM is in complementary mode, there are two comparators n and m to generate EPWM 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 EPWM outputs to generate complement paired waveforms. CMPU denotes CNT(EPWM_CNTn[15:0]) is equal to CMP(EPWM_CMPDATn[15:0]) when counting up. CMPD denotes CNT bits is equal to CMP bits when counting down.

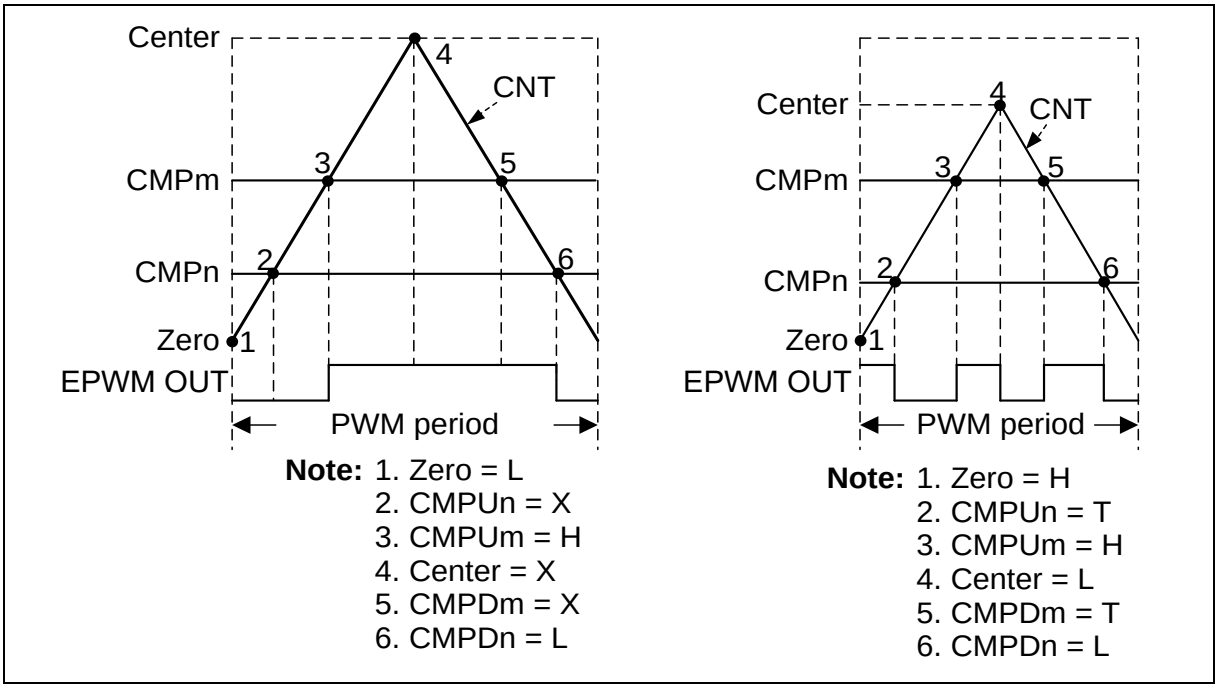


Figure 6.12-18 EPWM Pulse Generation

User can also use software event to force EPWM output. Software triggers SWETRGn (EPWM_SWEOFTRG[n], n=0~5) event can decide EPWM waveform to do nothing (X), set Low (L), set High (H) or toggle (T) by setting OUTACTSn (EPWM_SWEOFCTL[2n+1:2n]).

Figure 6.12-19 shows software event effect EPWM output according to OUTACTS0 setting. Besides, software event has the highest priority. If other events and software event happen at the same time, software event will be serviced first.

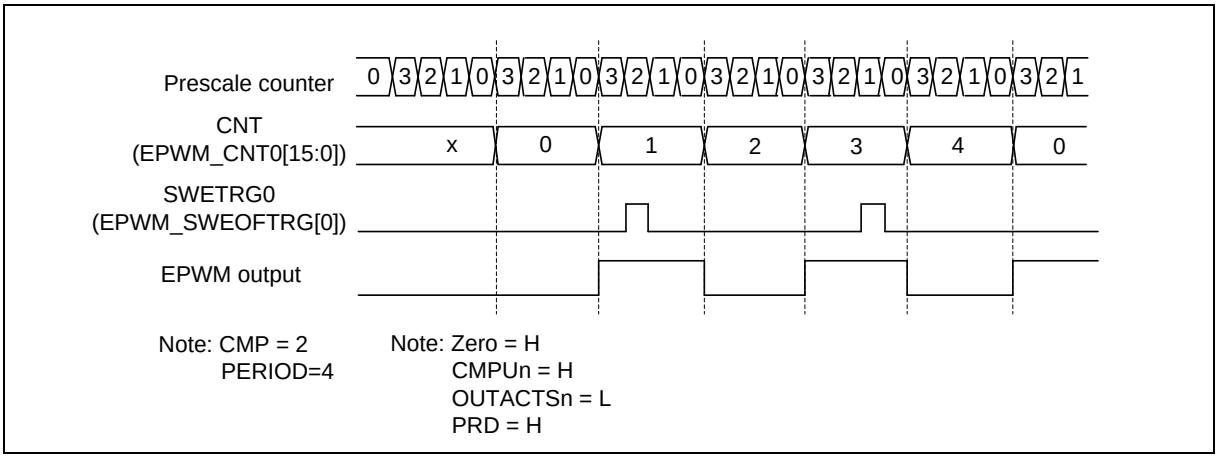


Figure 6.12-19 Software Event Output Force

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 6.12-2), down counter type (Table 6.12-3) and up-down counter type (Table 6.12-4). By using event priority, user can easily generate 0% to 100% duty pulse as shown in Figure 6.12-20.

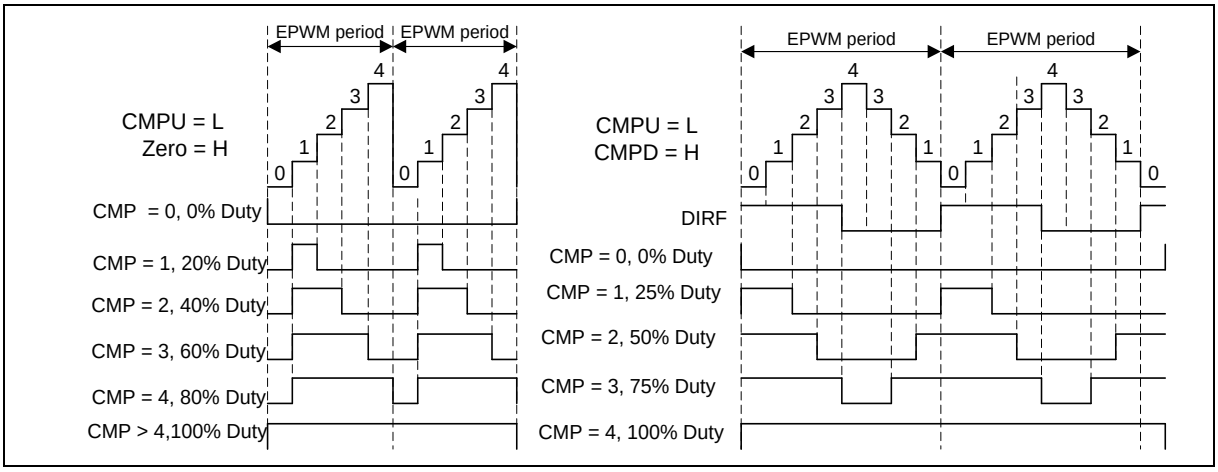


Figure 6.12-20 EPWM 0% to 100% Pulse Generation

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 = 0)

Table 6.12-2 EPWM Pulse Generation Event Priority for Up-Counter

Priority	Down Event
1 (Highest)	Zero event (CNT = 0)
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 6.12-3 EPWM 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 (Lowest)	Zero event (CNT = 0)	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 6.12-4 EPWM Pulse Generation Event Priority for Up-Down-Counter

6.12.5.14 EPWM Output Mode

The EPWM 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.

6.12.5.15 Independent mode

By default, the EPWM is operating in independent mode, independent mode is enabled when channel n corresponding OUTMODEn (EPWM_CTL1[26:24]) bit is set to 0. In this mode six EPWM channels: EPWM_CH0, EPWM_CH1, EPWM_CH2, EPWM_CH3, EPWM_CH4 and EPWM_CH5 are running off its own period and duty as shown in Figure 6.12-21.

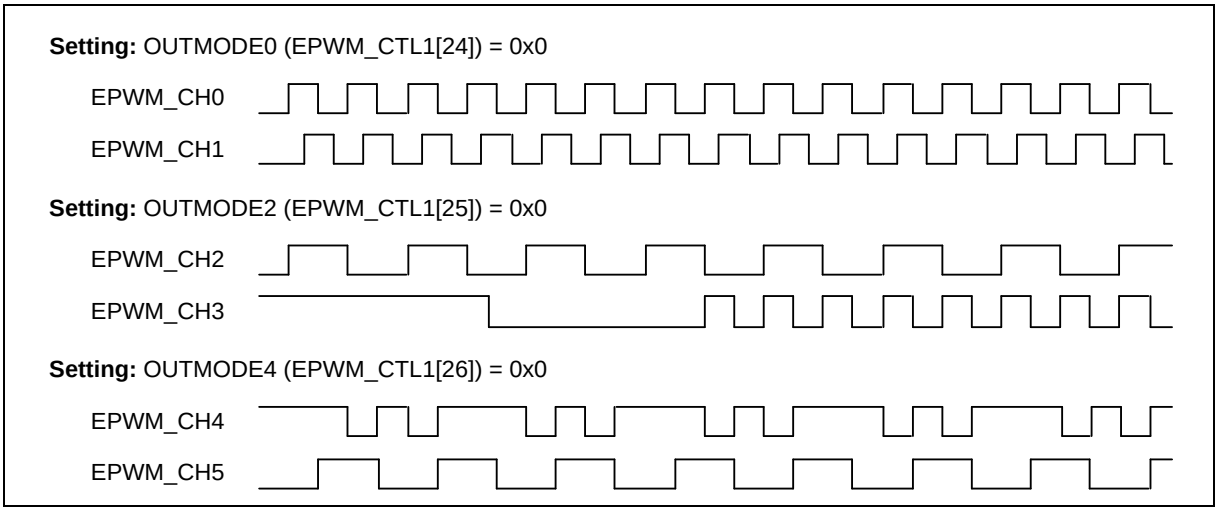


Figure 6.12-21 EPWM Independent Mode Waveform

6.12.5.16 Complementary Mode

Complementary mode is enabled when the pair channel corresponding OUTMODEn (EPWM_CTL1[26:24]) bit set to 1. In this mode there are 3 EPWM generators utilized for complementary mode, with total of 3 EPWM output paired pins in this module. In Complimentary modes, the internal odd EPWM signal must always be the complement of the corresponding even EPWM signal. EPWM_CH1 will be the complement of EPWM_CH0. EPWM_CH3 will be the complement of EPWM_CH2 and EPWM_CH5 will be the complement of EPWM_CH4 as shown in Figure 6.12-22.

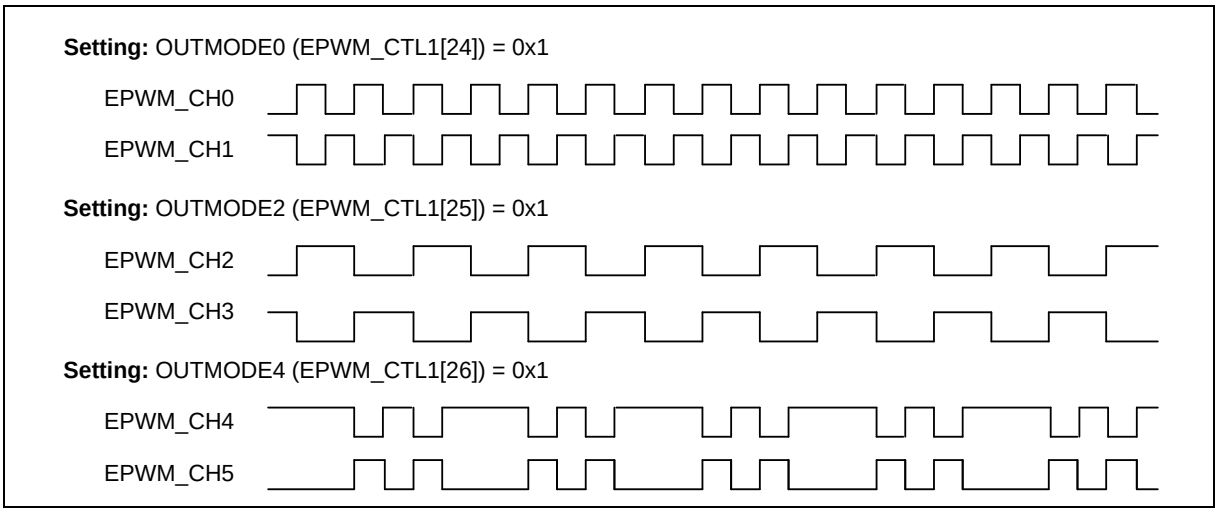


Figure 6.12-22 EPWM Complementary Mode Waveform

6.12.5.17 EPWM Output Function

Based on the output mode, there are two output functions: group and synchronous functions for advanced output control. Group function, forces the EPWM_CH2 and EPWM_CH4 synchronous with EPWM_CH0 generator and forces the EPWM_CH3 and EPWM_CH5 synchronous with EPWM_CH1, may simplify updating duty control in DC and BLDC motor applications. Besides, Synchronous function makes any channel of EPWM0 and EPWM1 in phase, user can control phase value and direction.

6.12.5.18 Group Function

Group function is enabled when GROUPEN (EPWM_CTL0[24]) is set to 1, no matter in independent or complementary mode. This control allows all even EPWM channels output to be controllable by EPWM_PERIOD0 and EPWM_CMPDAT0 registers and all odd EPWM channels output to be controllable by EPWM_PERIOD1 and EPWM_CMPDAT1 registers. That is, user only needs to set EPWM_CH0 to get EPWM_CH0, EPWM_CH2 and EPWM_CH4 output the same pulse, and set EPWM_CH1 to get EPWM_CH1, EPWM_CH3 and EPWM_CH5 output the same pulse, as shown in Figure 6.12-23. When operating group function, OUTMODE0, OUTMODE2 and OUTMODE4 bits of CTL1 register must all set to 0 for independent mode or all set to 1 for complementary mode.

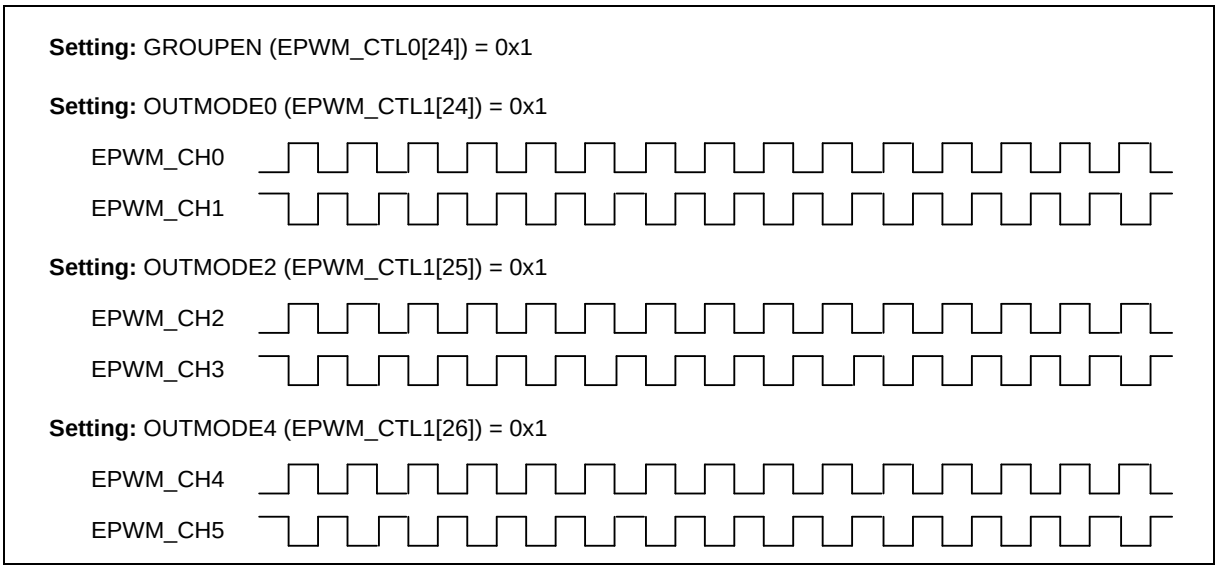


Figure 6.12-23 EPWM Group Function Waveform

6.12.5.19 Synchronous Function

Synchronous function can only be enabled when complementary mode is enabled. Figure 6.12-25 is counter synchronous function block diagram. Every counter of EPWM pairs has a SYNC_IN and a SYNC_OUT signals. The SYNC_IN signal for the first EPWM0 pair counter comes from EPWM0_SYNC_IN pin, and the others come from the SYNC_OUT signal of the previous EPWM pair counter. The input signal from EPWM0_SYNC_IN pin will be filtered by a 3-bit noise filter as Figure 6.12-24. In addition, it can be inversed by setting the bit SINPINV (EPWM_SYNC[23]) to realize the polarity setup for the input signal. The noise filter sampling clock can be selected by setting bits SFLTCSEL (EPWM_SYNC[19:17]) to fit different noise properties. Moreover, by setting the bits SFLTCNT (EPWM_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 (EPWM_SYNC[16]) will enable the noise filter function. By default, it is disabled.

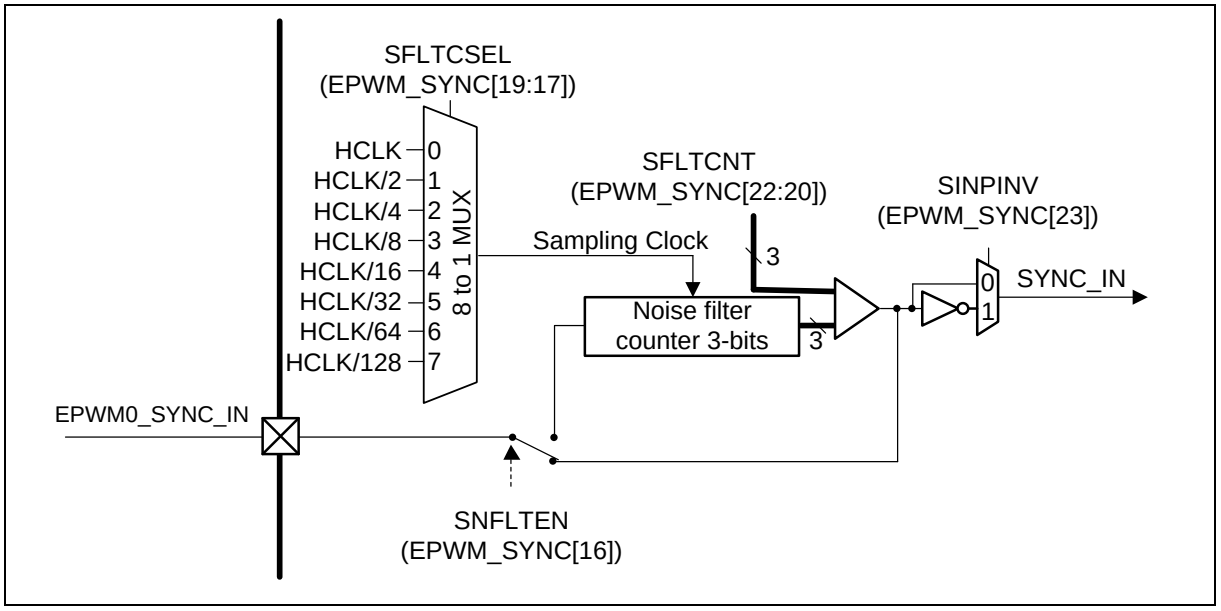


Figure 6.12-24 EPWM SYNC_IN Noise Filter Block Diagram

User can use SINSRCn (EPWM_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 EPWM0_SYNC_IN pin is high or setting SWSYNCn (EPWM_SWSYNC[2:0]) to 1. Synchronizing source can also be selected as CNT = 0 or CNT = EPWM_CMPDATm register (if being the up-down counter type, it will synchronize twice in a EPWM period) to trigger a sync event or to disable SYNC_OUT signal.

When the PHSENn (EPWM_SYNC[2:0]) is enabled and the synchronous source has a happening event, the counter will load a value from the PHS (EPWM_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 (EPWM_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 EPWM enable. To start EPWM and BPWM counters in the same time, user have to set the EPWM Synchronous Start Control Register (EPWM_SSCTL[5:0]) to enable the channel counters which are planned to start counting together, and select the SSRC(EPWM_SSCTL[9:8]) to choose the Synchronous Start source, followed by setting the EPWM Synchronous Start Trigger Register CNTSEN (EPWM_SSTRG[0]).

For applications, please do not use Group and Synchronous function simultaneously because the Synchronous function will be inactive.

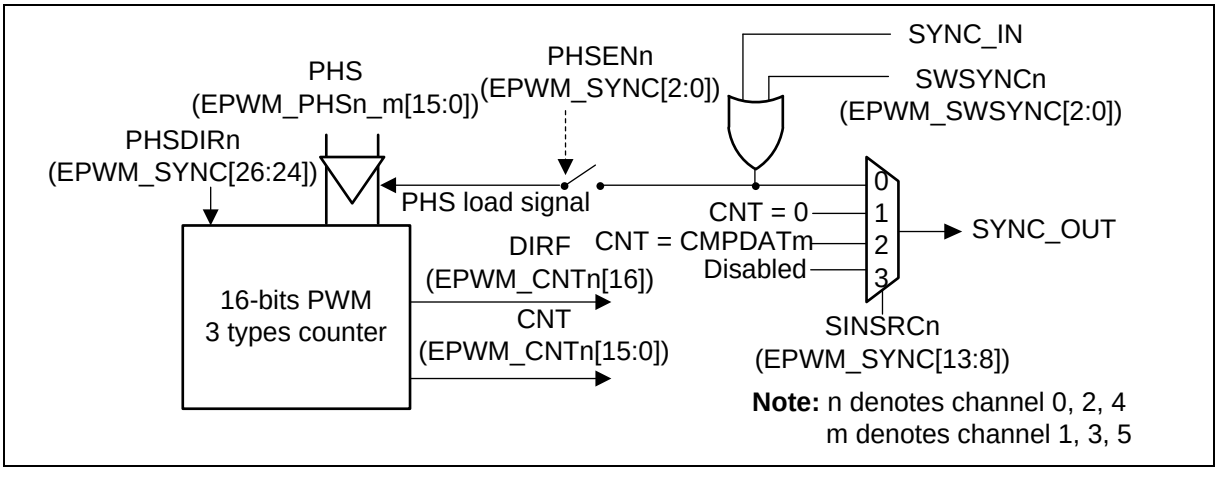


Figure 6.12-25 EPWM Counter Synchronous Function Block Diagram

Figure 6.12-26 is an example of the synchronous function in the up-down counter type. In the example, synchronizing source comes from the external EPWM SYNC_IN signal. At the beginning, the output waveform of EPWM_CH0, EPWM_CH2 and EPWM_CH4 are in the same phase. Then at Point A, the EPWM 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(EPWM_PHSn_m[15:0]) as well as the counting direction in the PHSDIRn (EPWM_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 6.12-26.

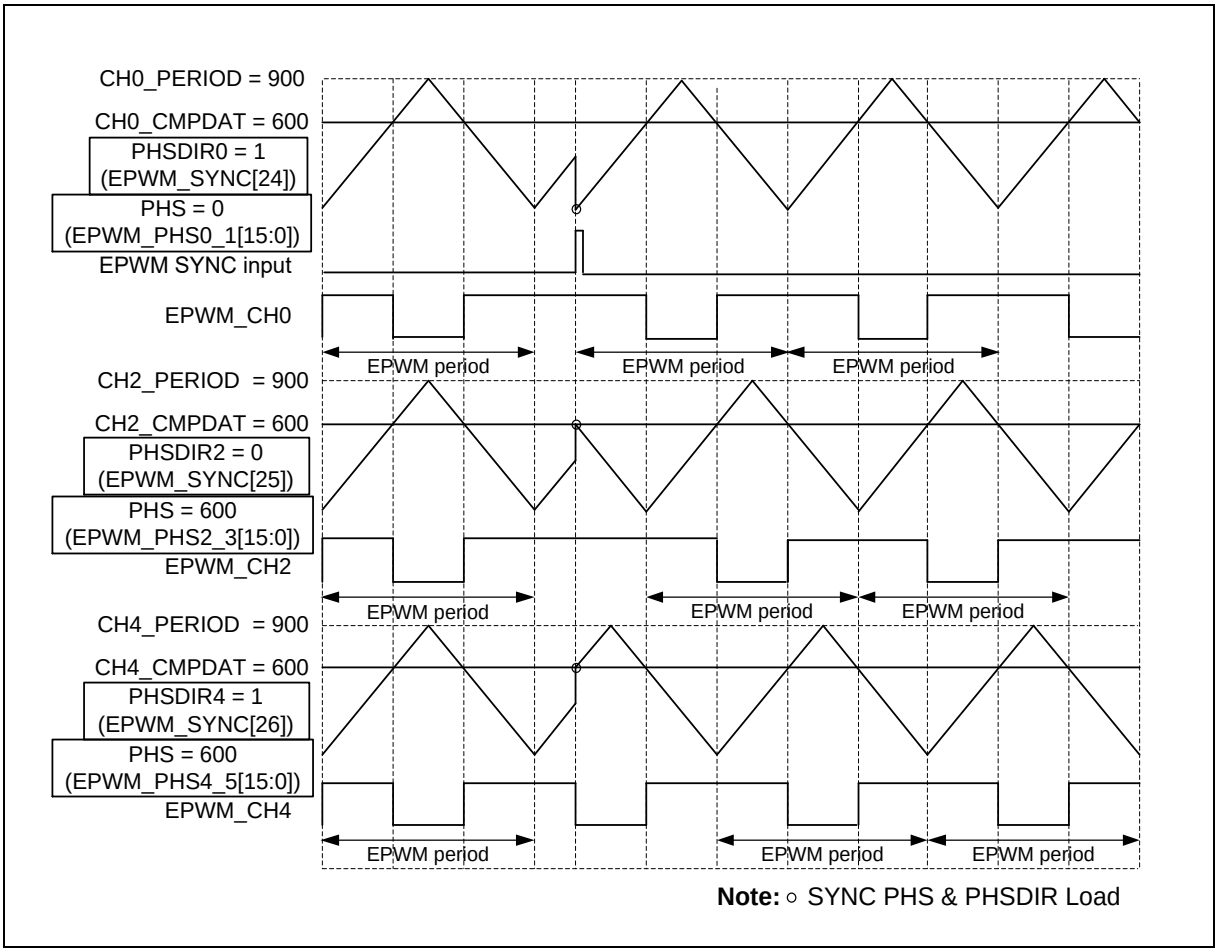


Figure 6.12-26 EPWM Synchronous Function with Synchronize source from SYNC_IN Signal

6.12.5.20 EPWM Output Control

After EPWM pulse generation, there are four to six steps to control the output of EPWM channels. In independent mode, there are Mask, Brake, Pin Polarity and Output Enable four steps as shown in Figure 6.12-27. In complementary mode, it needs two more steps to precede these four steps, Complementary channels and Dead-Time Insertion as shown in Figure 6.12-28.

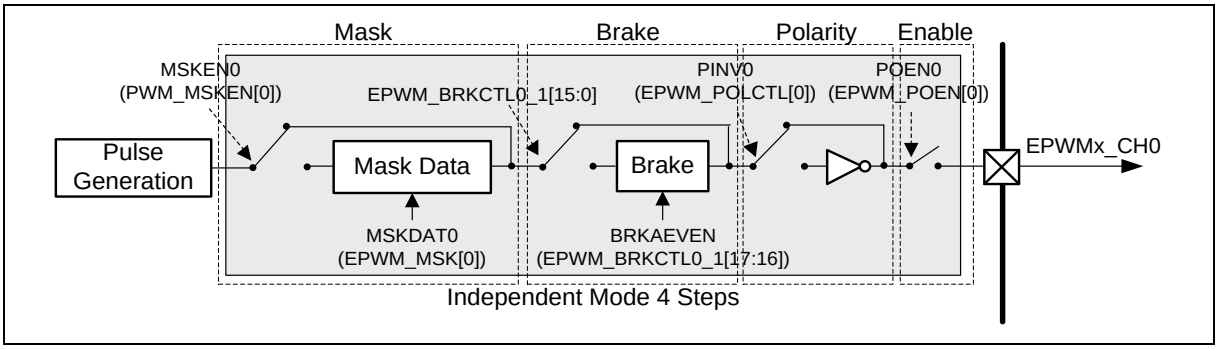


Figure 6.12-27 EPWMx_CH0 Output Control in Independent Mode

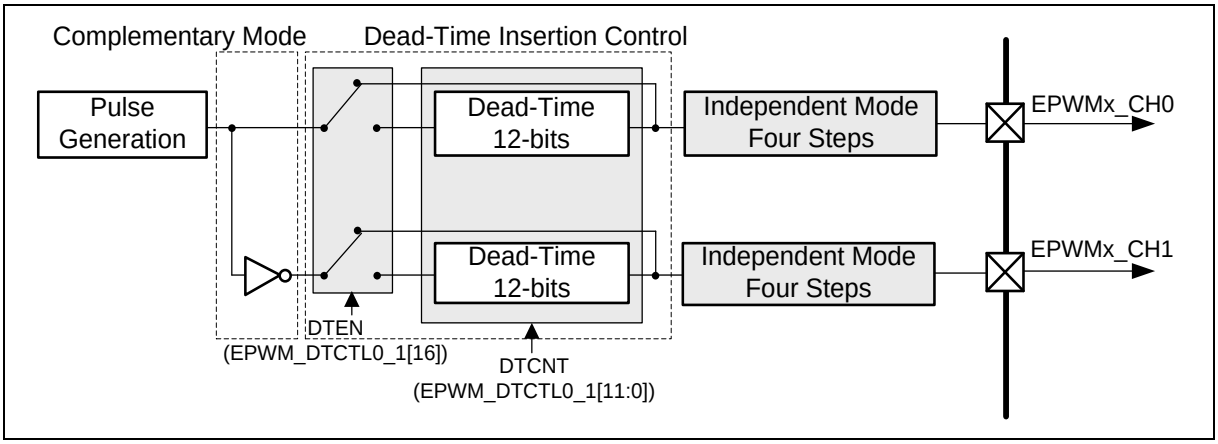


Figure 6.12-28 EPWMx_CH0 and EPWMx_CH1 Output Control in Complementary Mode

6.12.5.21 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 RDTEN (EPWM_DTCTL[n/2]) bit to enable rising dead-time function and FDTEN (EPWM_DTCTL[n/2+8]) bit to enable falling dead-time function. Setting RDTCNT (EPWM_RDTCNTn_m[11:0], n=0,2,4, m=n+1) and FDTCNT (EPWM_FDTCNTn_m[11:0], n=0,2,4, m=n+1) will control dead-time rising and falling period. Rising and falling dead-time can be calculated from the following formula:

Rising Dead-time = (RDTCNT (EPWM_RDTCNTn_m[11:0])+1) * EPWMx_CLK period

Falling Dead-time = (FDTCNT (EPWM_FDTCNTn_m[11:0])+1) * EPWMx_CLK period

Dead-time insertion clock source can be selected from prescaler output by setting DTCKSELn (EPWM_DTCTL[16+n], n=0,2,4) to 1. By default, clock source comes from EPWM_CLK, which is prescaler input. Then the rising and falling dead-time can be calculated from the following formula:

Rising Dead-time = (RDTCNT (EPWM_RDTCNTn_m[11:0])+1) *

(CLKPSC(EPWM_CLKPSCn[11:0])+1) * EPWMx_CLK period

Falling Dead-time = (FDTCNT (EPWM_FDTCNTn_m[11:0])+1) *

(CLKPSC(EPWM_CLKPSCn[11:0])+1) * EPWMx_CLK period

Please note that the EPWM_RDTCNTn_m, EPWM_FDTCNTn_m and EPWM_DTCTL are write-protected registers.

Figure 6.12-29 indicates the dead-time insertion for one pair of EPWM signals.

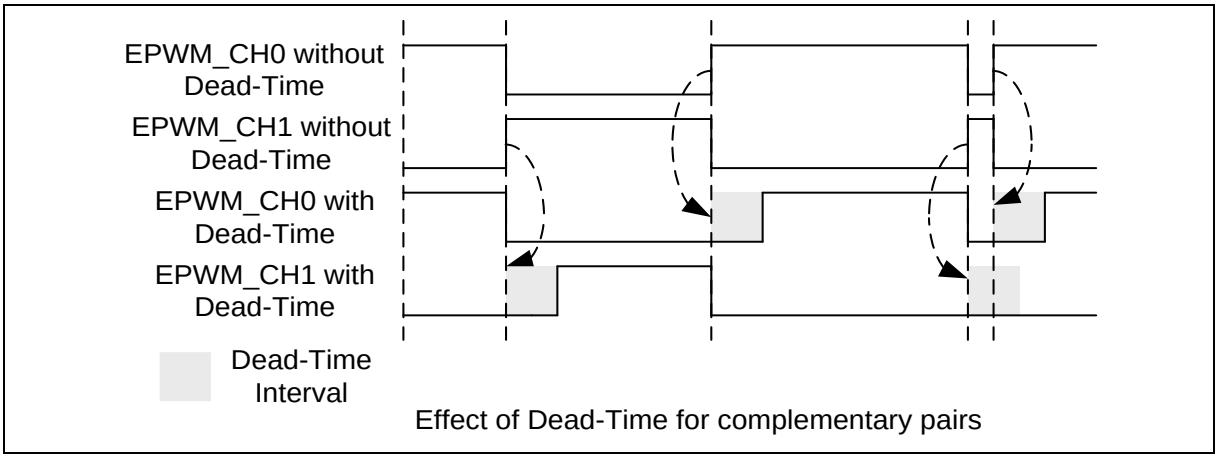


Figure 6.12-29 Dead-Time Insertion

6.12.5.22 EPWM Mask Output Function

Each of the EPWM channel output value can be manually overridden with the settings in the EPWM Mask Enable Control Register (EPWM_MSKEN) and the EPWM Masked Data Register (EPWM_MSK) With these settings, the EPWM channel outputs can be assigned to specified logic states independent of the duty cycle comparison units. The EPWM mask bits are useful when controlling various types of Electrically Commutated Motor (ECM) like a BLDC motor. The EPWM_MSKEN register contains six bits, MSKENn(EPWM_MSKEN[5:0]). If the MSKENn is set to active-high, the EPWM channel n output will be overridden. The EPWM_MSK register contains six bits, MSKDATn(EPWM_MSK[5:0]). The bit value of the MSKDATn determines the state value of the EPWM channel n output when the channel is overridden. Figure 6.12-30 shows an example of how EPWM mask control can be used for the override feature.

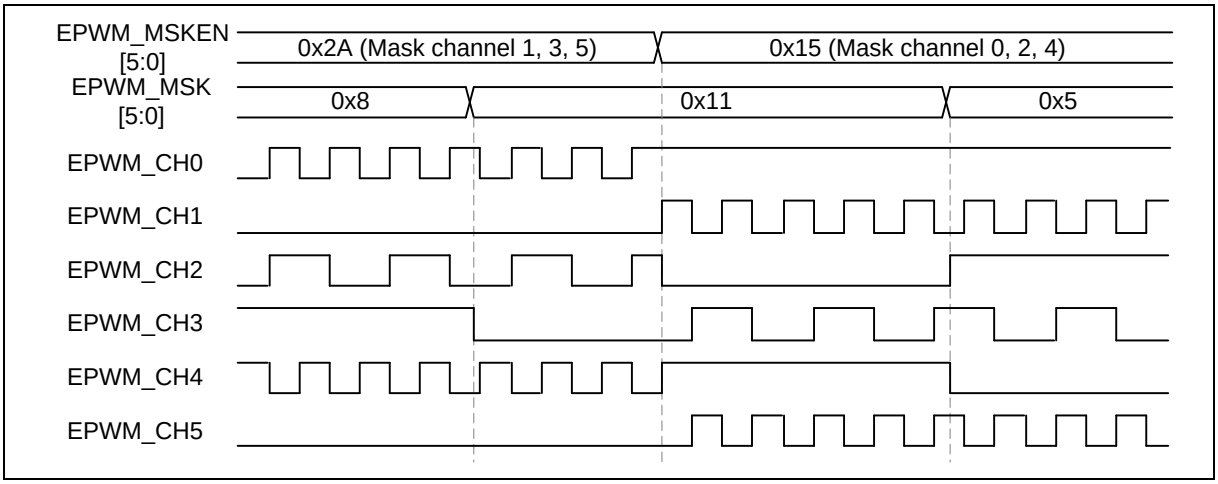


Figure 6.12-30 Illustration of Mask Control Waveform

6.12.5.23 EPWM Brake

Each EPWM module has two external input brake control signals. User can select active brake pin source is from EPWMx_BRAKEy pin by BKxSRC bits of EPWM_BNF register(x=0,1, y=0,1). The external signals will be filtered by a 3-bit noise filter. User can enable the noise filter function by BRKxNFEN bits of EPWM_BNF, and noise filter sampling clock can be selected by setting BRKxNFSEL bits of EPWM_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 EPWM_BNF register to realize the polarity setup for the brake control signals. Set BRKxPINV bit to 0, brake event will occurred when EPWMx_BRAKEy(x=0,1, y=0,1) pin status is from low to high; set BRKxPINV to 1, brake event will occurred when EPWMx_BRAKEy pin status is from high to low.

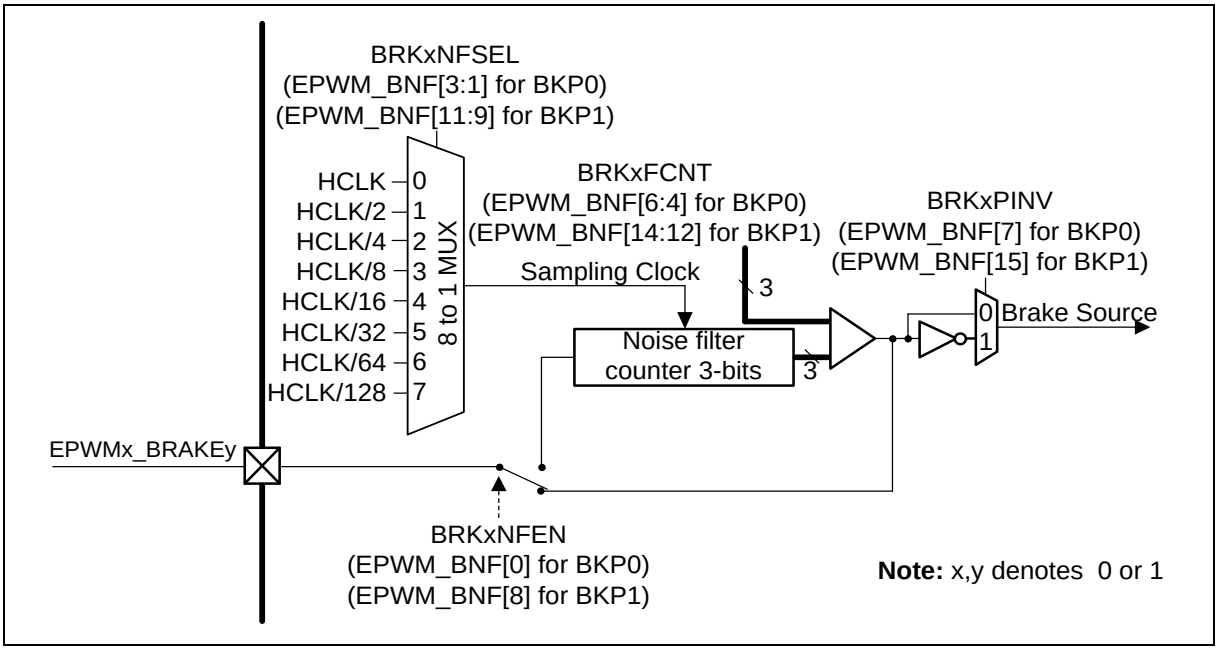


Figure 6.12-31 Brake Noise Filter Block Diagram

For Complementatry 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 EPWM brake function, as shown Figure 6.12-32. To control paired channels to output safety state, user can setup BRKAEVEN (EPWM_BRKCTL0_1[17:16]) for even channels and BRKAODD (EPWM_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 (EPWM_INTEN1[2:0]) is enabled, the brake function generates BRK_INT. This interrupt needs software to clear, and the BRKESTSn (EPWM_INTSTS1[21:16]) brake state will keep until the next EPWM 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 (EPWM_INTEN1[10:8]) is also enabled, the brake function will generate BRK_INT, but BRKLSTSn (EPWM_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 EPWM waveform period when brake condition removed without clear interrupt.

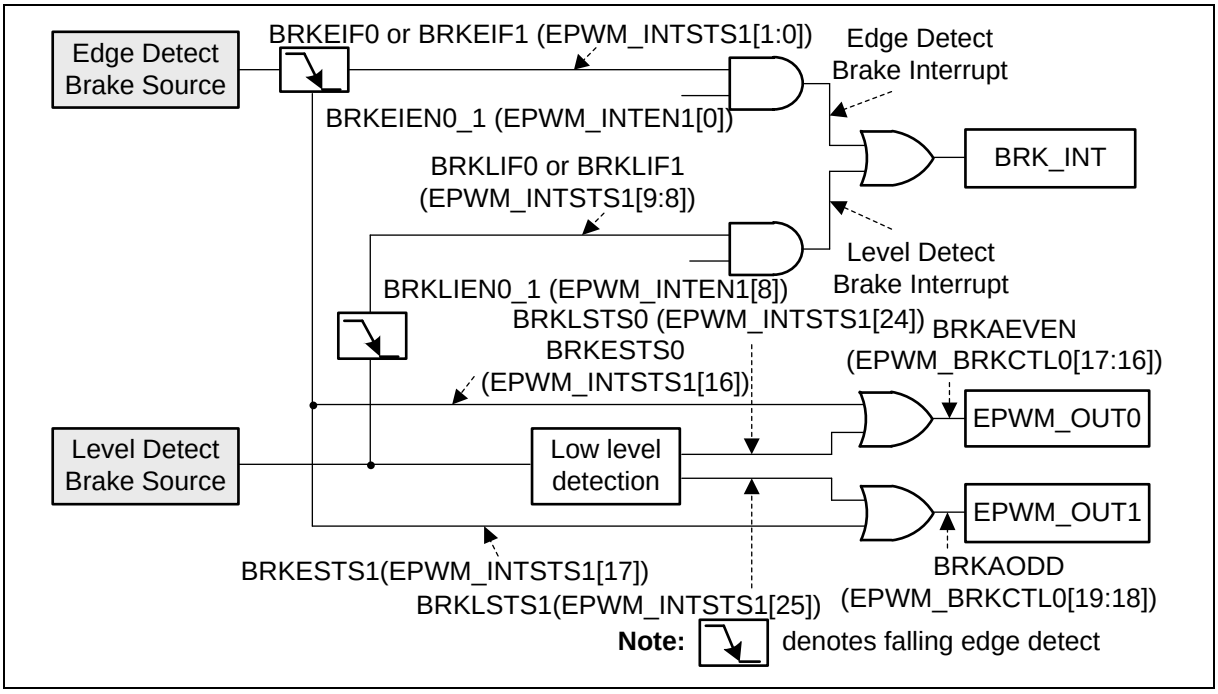


Figure 6.12-32 Brake Block Diagram for EPWMx_CH0 and EPWMx_CH1 Pair

Figure 6.12-33 illustrates the edge detector waveform for EPWMx_CH0 and EPWMx_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 EPWMx_CH0 and EPWMx_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 EPWM period. At the same moment, the EPWMx_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, EPWMx_CH1 outputs normally at the next start of the EPWM period.

As a contrast to the edge detector example, Figure 6.12-34 illustrates the level detector waveform for EPWMx_CH0 and EPWMx_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 EPWM period no matter at what states the BRKLIF0 and BRKLIF1 flags are at that moment.

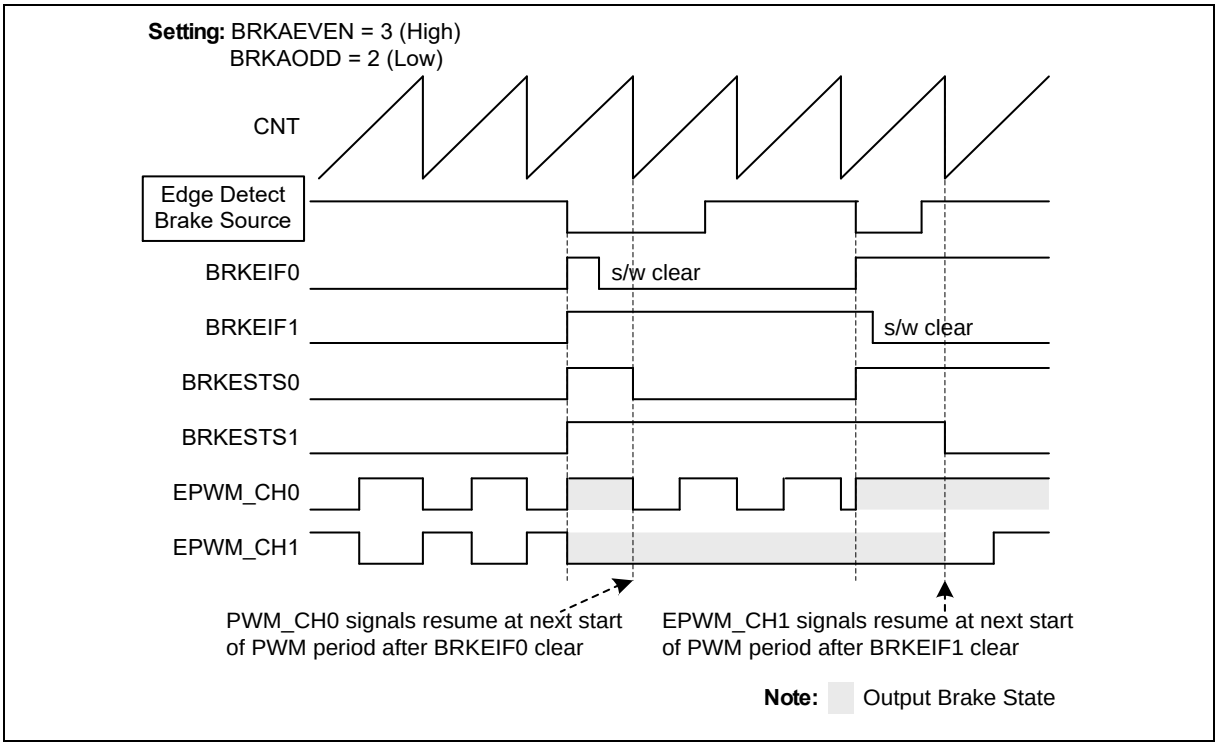


Figure 6.12-33 Edge Detector Waveform for EPWMx_CH0 and EPWMx_CH1 Pair

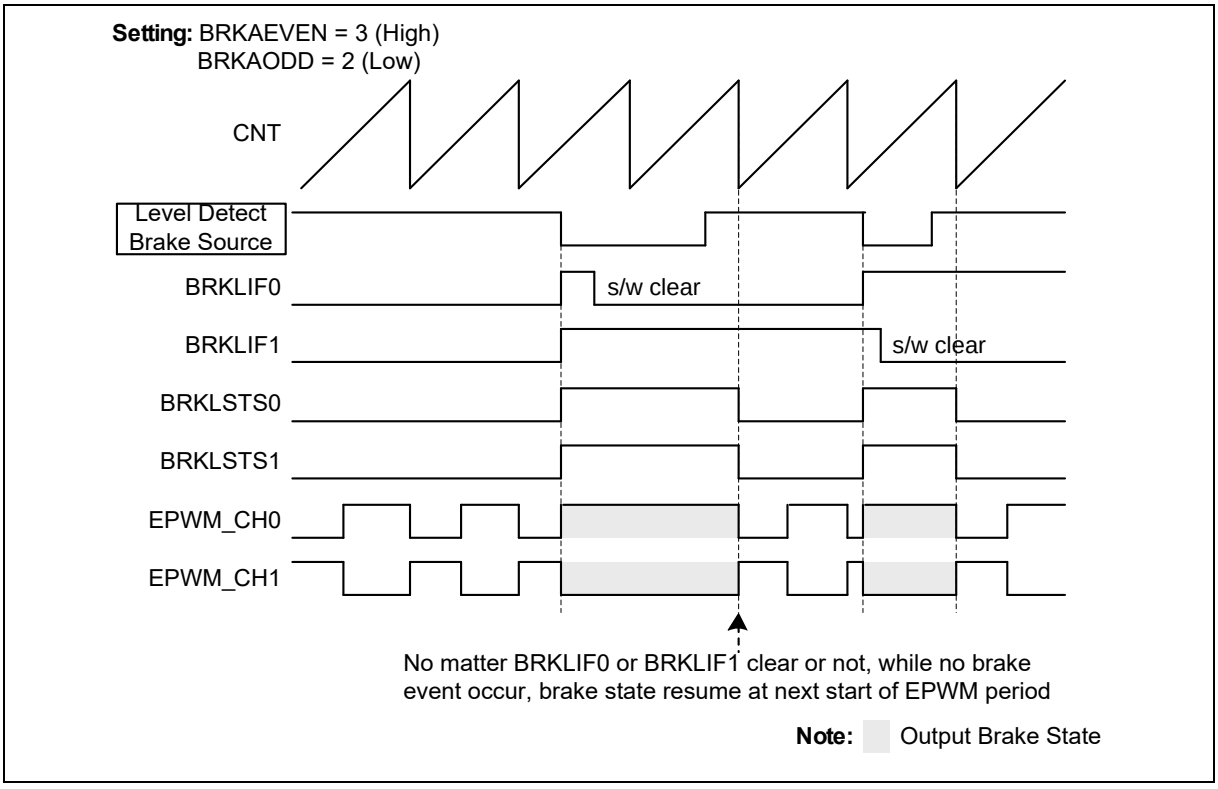


Figure 6.12-34 Level Detector Waveform for EPWMx_CH0 and EPWMx_CH1 Pair

The two kinds of detectors detect the same seven brake sources: two from external input signals, two from analog comparators(ACMP), one from EADC result monitor (EADCRM), one from system fail and

one from software triggered, that are shown in Figure 6.12-35. ACMP brake sources will be detected only when internal ACMP0_O or ACMP1_O signal from low to high.

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 6.12-36 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 EPWM brake.

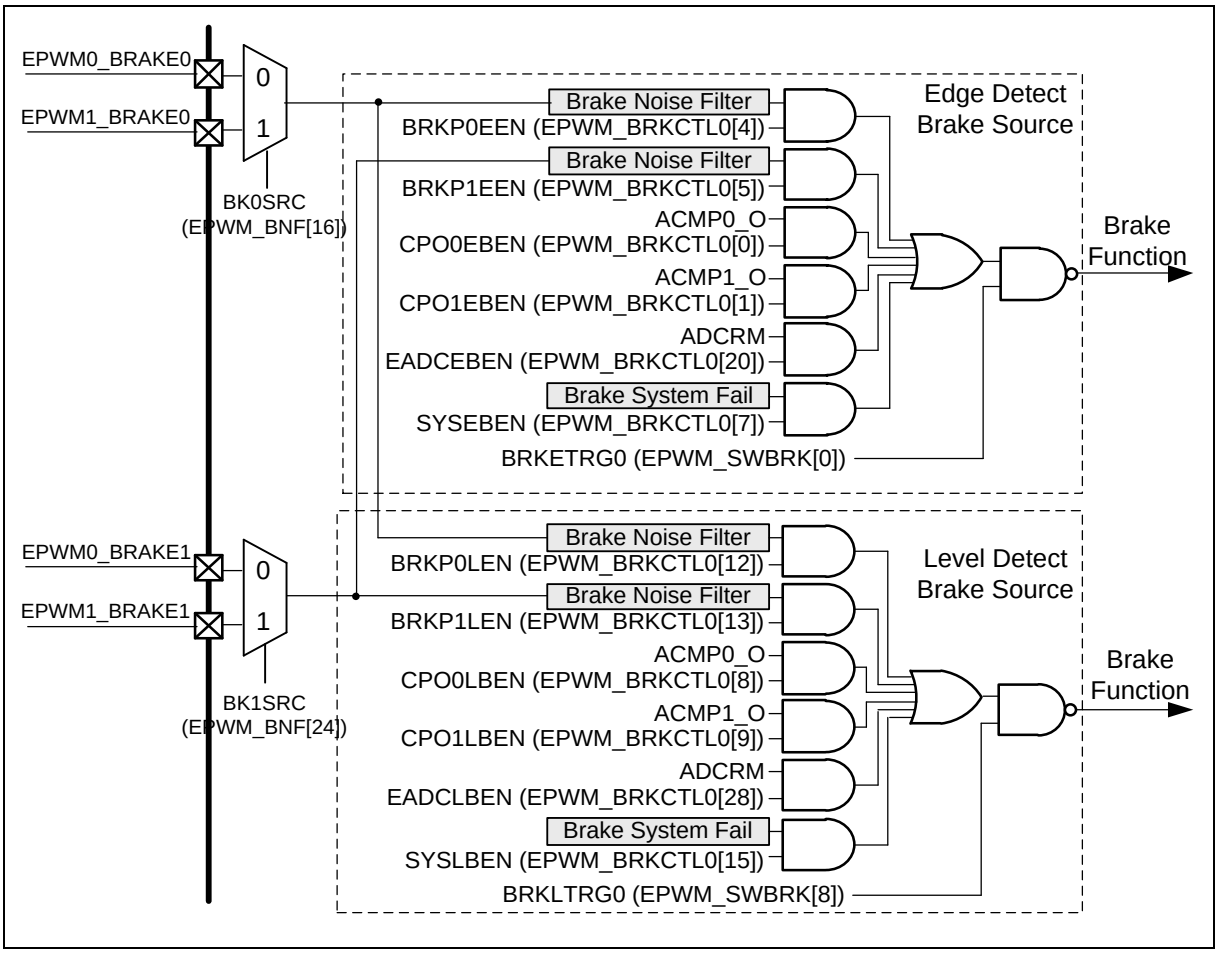


Figure 6.12-35 Brake Source Block Diagram

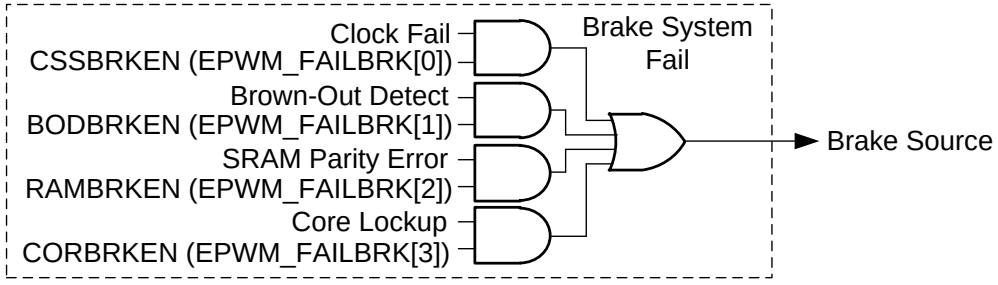


Figure 6.12-36 Brake System Fail Block Diagram

6.12.5.24 LEB Function

Leading edge blanking (LEB) function is use to blank the false trigger from brake source ACMP which may caused by EPWM output transition. Set LEBEN (EPWM_LEBCTL[0]) to enable this function. LEB source comes from EPWM_CH0, EPWM_CH2 and EPWM_CH4, use SRCENn (EPWM_LEBCTL[10:8]) as input source enable. LEB function blanking time is decided by LEBCNT (EPWM_LEBCNT[8:0]), when LEB detected trigger edge, then blanking time will count from LEBCNT+1 to 0, the counter clock base is ECLK. If a new trigger event occurs, blanking counter will reset to LEBCNT and down count again. LEB trigger edge can be rising, falling or both rising and falling edge by setting TRGTYPE (EPWM_LEBCTL[17:16]). Figure 6.12-37 shows that LEB will blanking leading edge caused by EPWM_CH0 and EPWM_CH4.

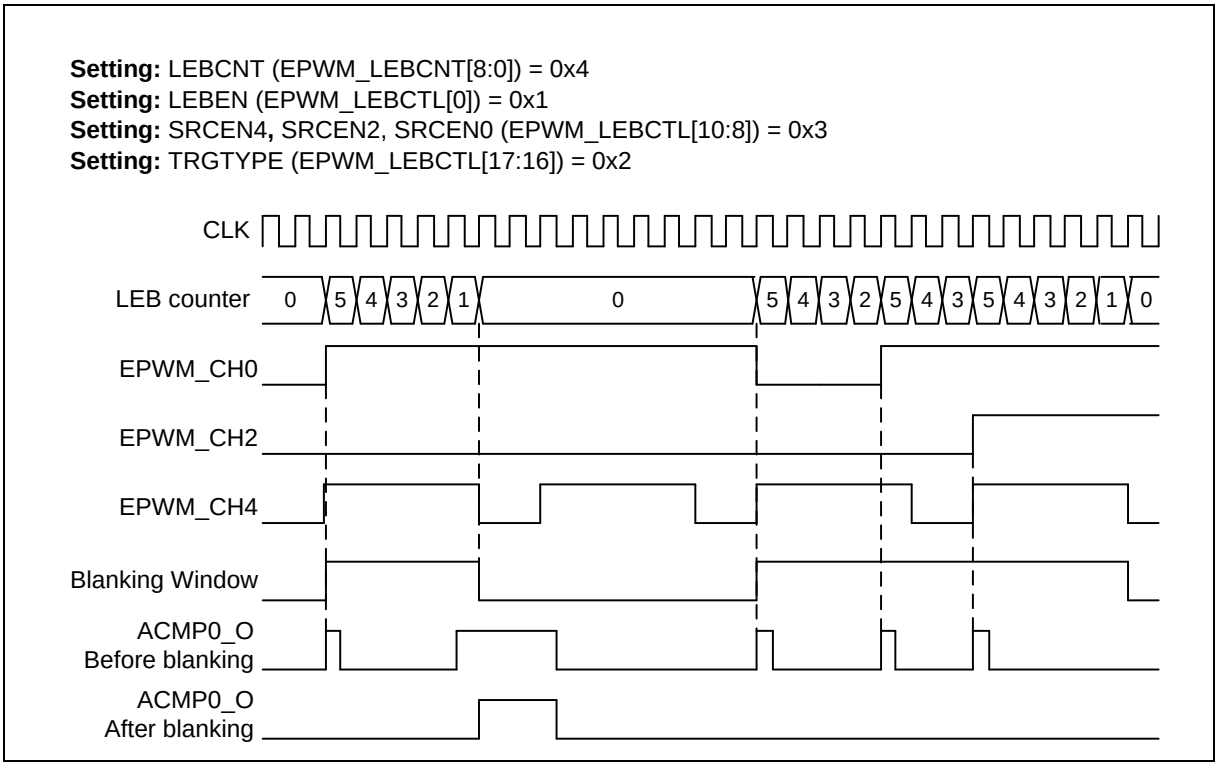
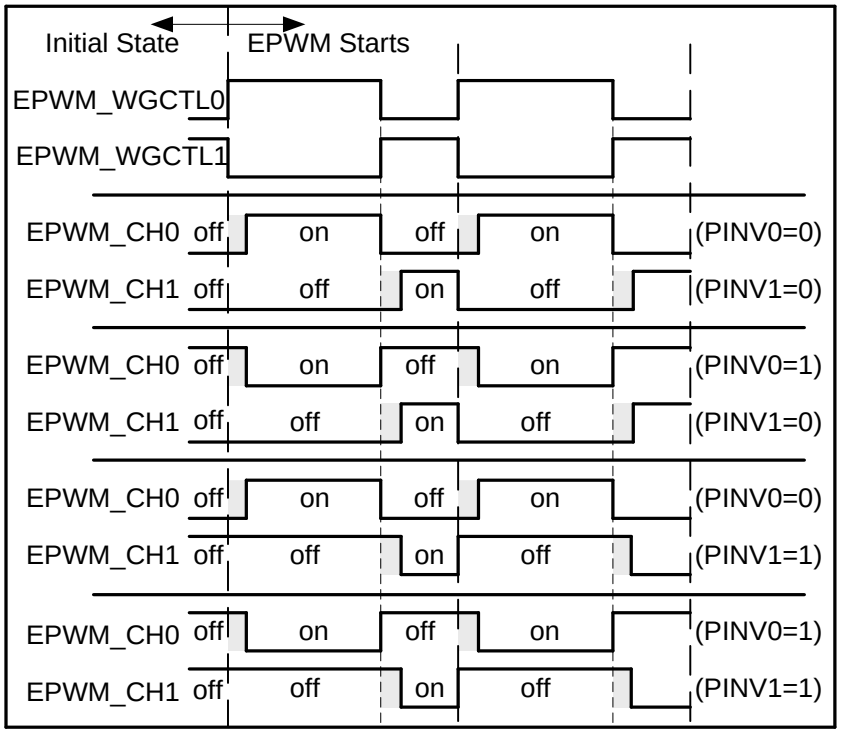


Figure 6.12-37 EPWM LEB Function Waveform

6.12.5.25 Polarity Control

Each EPWM port, from EPWM_CH0 to EPWM_CH5, has an independent polarity control module to configure the polarity of the active state of the EPWM output. By default, the EPWM output is active high. This implies the EPWM OFF state is low and ON state is high. This definition is variable through setting the EPWM Negative Polarity Control Register (EPWM_POLCTL), for each individual EPWM channel. Figure 6.12-38 shows the initial state before EPWM starting with different polarity settings.



Dead-time insertion; It is only effective in complementary mode

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

Figure 6.12-38 Initial State and Polarity Control with Rising Edge Dead-Time Insertion

6.12.5.26 EPWM Interrupt Generator

Interrupts for each EPWM show in Figure 6.12-40 and Figure 6.12-41.

The 1st EPWM interrupt (EPWM_INT) comes from EPWM complementary pair events. EPWM complementary pair shares the EPWM_INT vector in NVIC. The counter can generate the Zero point Interrupt Flag ZIFn (EPWM_INTSTS0[5:0], n=0,1..5) and the Period point Interrupt Flag PIFn (EPWM_INTSTS0[13:8], n=0,1..5). When EPWM channel n's counter equals to the comparator value stored in EPWM_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 (EPWM_INTSTS0[21:16]) is set and if matching at the opposite direction, the Down Interrupt Flag CMPDIFn (EPWM_INTSTS0[29:24]) is set. If the corresponding interrupt enable bits are set, the trigger events will generates interrupt signals.

EPWM_INT can use the EPWM_IFAn (n=0~5) register to accumulate the number of times that the interrupt flags have been triggered for each channel. By setting one of IFAEN (EPWM_IFAn[31], n=0~5) bit to 1 to enable accumulator, EPWM_INT will switch interrupt source from every event trigger interrupt to trigger interrupt once every accumulate times.

By setting the IFASEL (EPWM_IFAn[29:28], n=0~5) bits, user can select one of the 4 interrupt sources to accumulate interrupt flag times for each channel, and the number of times interrupt flags will compare with IFACNT (EPWM_IFAn[15:0], n=0~5) bits. When interrupt accumulator equals IFACNT then set IFAIFn (EPWM_AINTSTS[n], n=0~5) bits as EPWM_INT signal if user enable IFAIENn (EPWM_AINTEN[n], n=0~5) bits. Accumulator interrupt of each channel can also be as

request source of PDMA. Figure 6.12-39 is an example of channel 0 using EPWM_IFA0 register to output EPWM_INT once every IFCNT0+1 times interrupt events occurred.

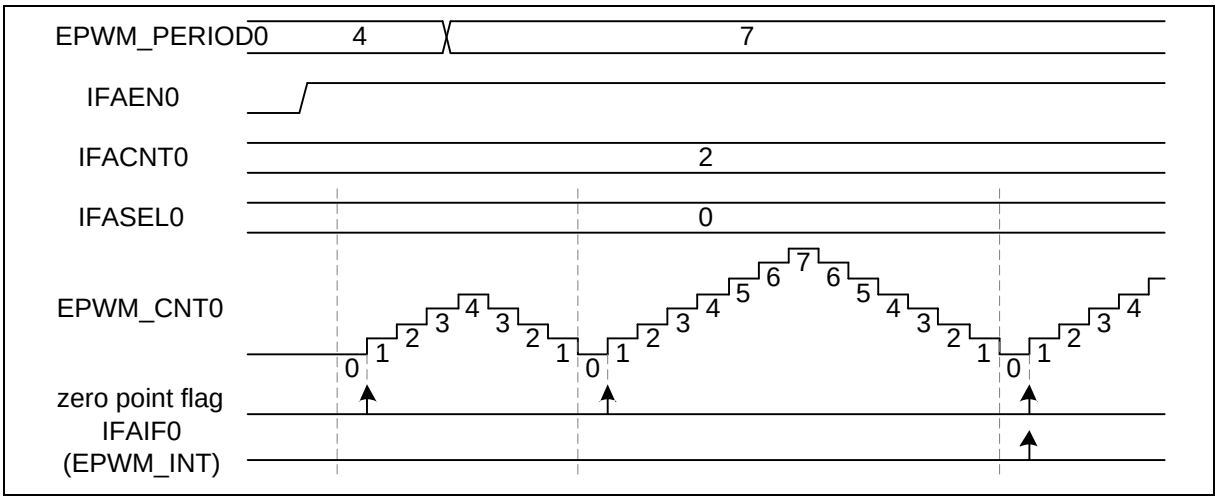


Figure 6.12-39 EPWMx_CH0 Accumulate Interrupt Waveform

The 2nd interrupt is the capture interrupt (CAP_INT). It shares the EPWM_INT vector in NVIC. The CAP_INT can be generated when the CRLIFn (EPWM_CAPIF[5:0]) flag is triggered and the Capture Rising Interrupt Enable bit CAPRIENn (EPWM_CAPIEN[5:0]) is set to 1. Or in the falling edge condition, the CFLIFn (EPWM_CAPIF[13:8]) flag can be triggered when the Capture Falling Interrupt Enable bit CAPFIENn (EPWM_CAPIEN[13:8]) is set to 1.

The 3nd interrupt is the brake interrupt (BRK_INT). The details of the BRK_INT is described in the EPWM Brake section. Figure 6.12-40 demonstrates the architecture of the EPWM interrupts.

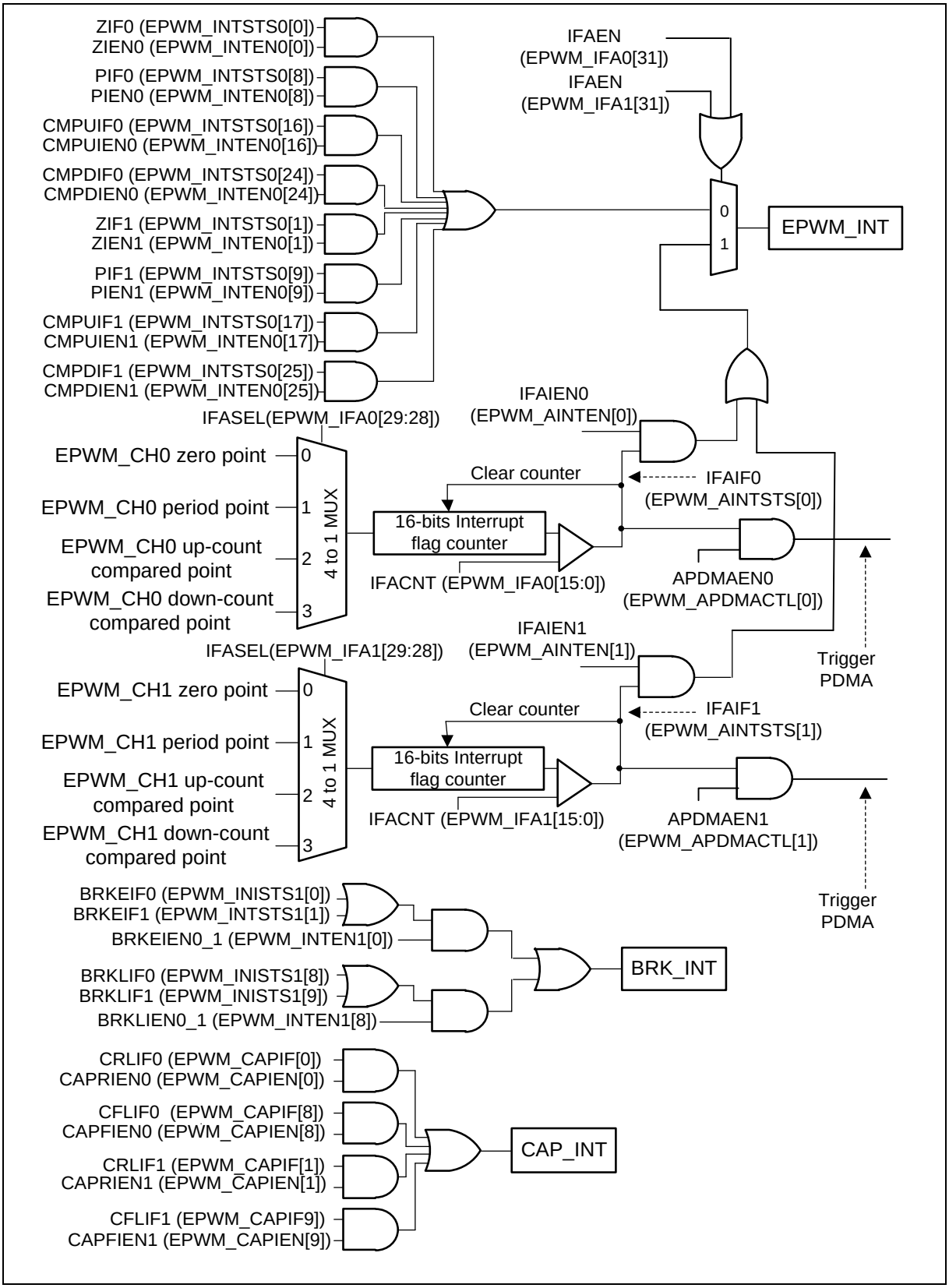


Figure 6.12-40 EPWMx_CH0 and EPWMx_CH1 Pair Interrupt Architecture Diagram

The 4nd interrupt is Fault Detect Function interrupt (FLT_INT). It shares the EPWM_INT vector in NVIC. Figure 6.12-41 shows the FLT_INT can be generate when FDIENn (EPWM_FDIEN[n], n=0~5) is set to 1 and FDIFn (EPWM_FDSTS[n], n=0~5) is set to 1 by detecting output open short.

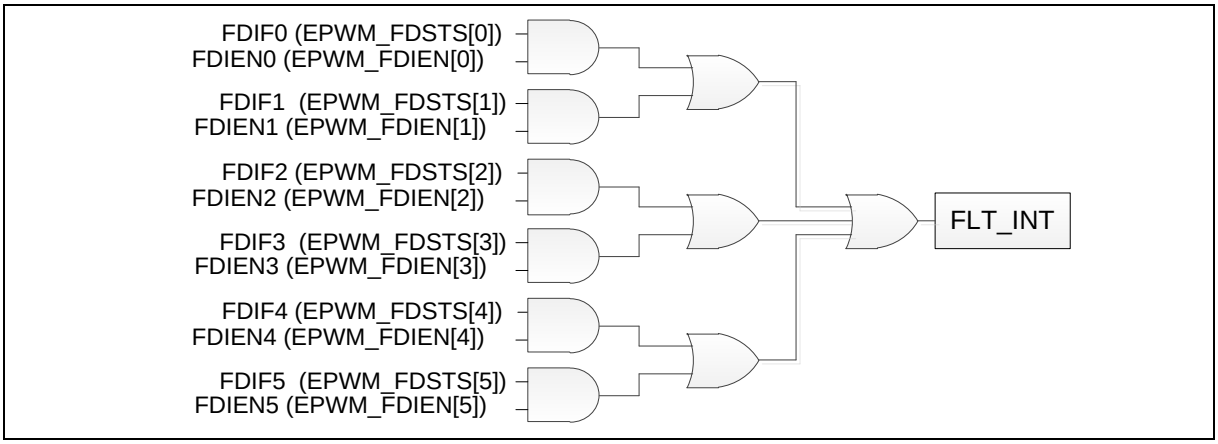


Figure 6.12-41 Fault Detect Function Interrupt Architecture Diagram

6.12.5.27 EPWM Trigger EADC/DAC Generator

EPWM can be one of the EADC conversion trigger source. Each EPWM pair channels share the same trigger source. Setting TRGSELn bit of EPWM_EADCTS0 and EPWM_EADCTS1 registers is to select the trigger sources, where TRGSELn bit is TRGSEL0, TRGSEL1, …, and TRGSEL5, which are located in EPWM_EADCTS0[3:0], EPWM_EADCTS0[11:8], EPWM_EADCTS0[19:16], EPWM_EADCTS0[27:24], EPWM_EADCTS1[3:0] and EPWM_EADCTS1[11:8], respectively. Setting TRGENn bit of EPWM_EADCTS0 and EPWM_EADCTS1 registers is to enable the trigger output to EADC, where TRGENn bit is TRGEN0, TRGEN1, …, TRGEN5, which are located in EPWM_EADCTS0[7], EPWM_EADCTS0[15], EPWM_EADCTS0[23], EPWM_EADCTS0[31], EPWM_EADCTS1[7] and EPWM_EADCTS1[15], respectively. The number n (n = 0,1, ..,5) denotes EPWM channel number.

There are 16 EPWM events can be selected as the trigger source for one pair of channels which shown in Figure 6.12-42. Figure 6.12-43 shows trigger EADC block diagram with prescaler. By setting PSCENn (EPWM_EADCPSCCTR[n], n=0~5) bit to 1, EPWM will trigger EADC when the number of trigger events is equal to EADCPSCn (n=0~5). Reading PSCNTn (n=0~5) can know how many events happened. If PSCENn (n=0~5) is 0, user also can write initial data to PSCNTn (n=0~5) and pre-scale counter will start from this data. If PSCENn (n=0~5) is set from 1 to 0, PSCNTn (n=0~5) will reset to 0. Figure 6.12-44 is the trigger EADC timing waveform in the up-down counter type.

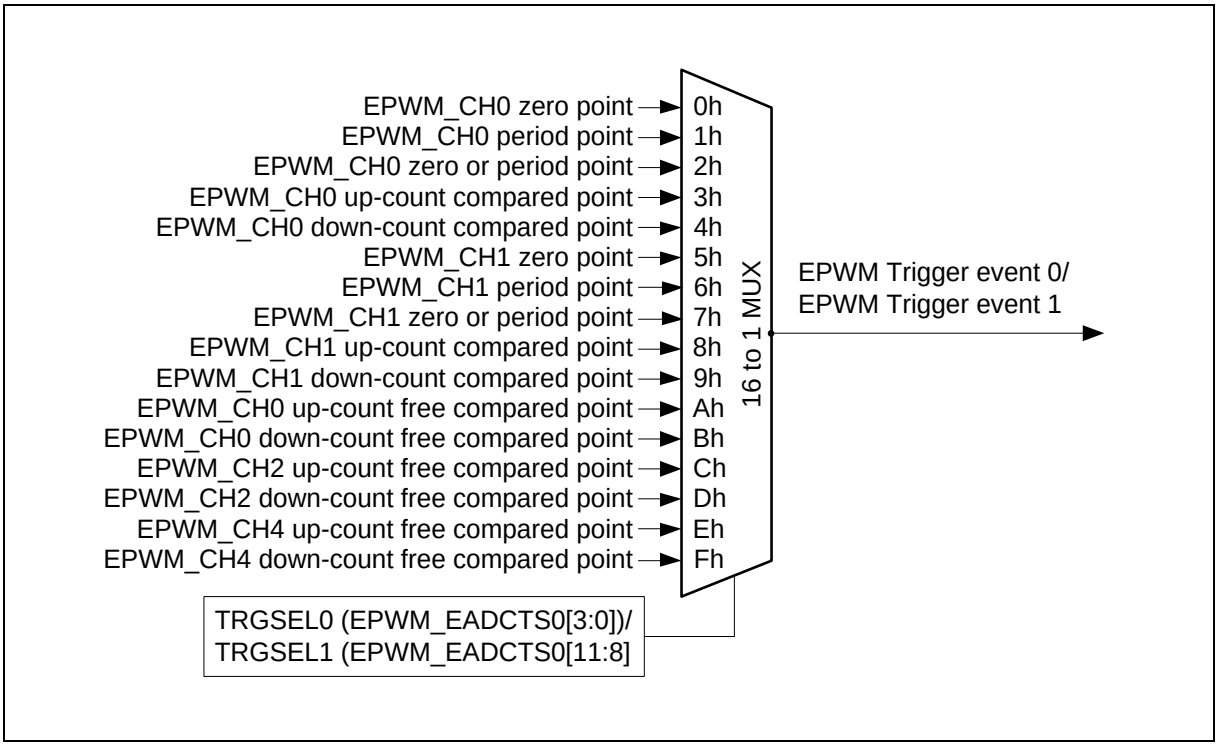


Figure 6.12-42 EPWMx_CH0 and EPWMx_CH1 Pair Trigger EADC Events

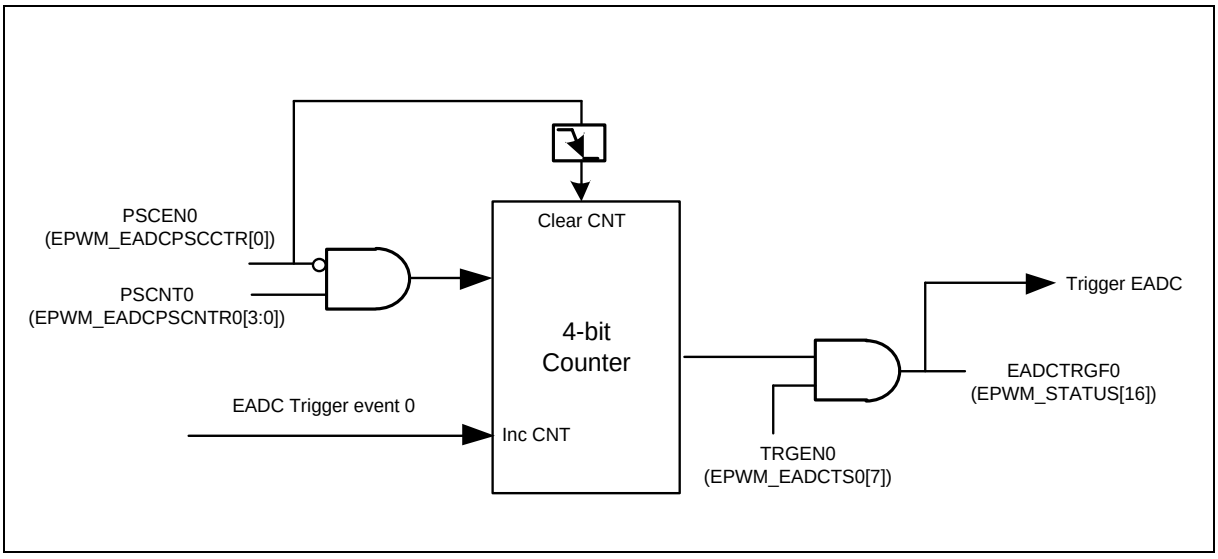


Figure 6.12-43 EPWMx_CH0 Trigger EADC Block Diagram

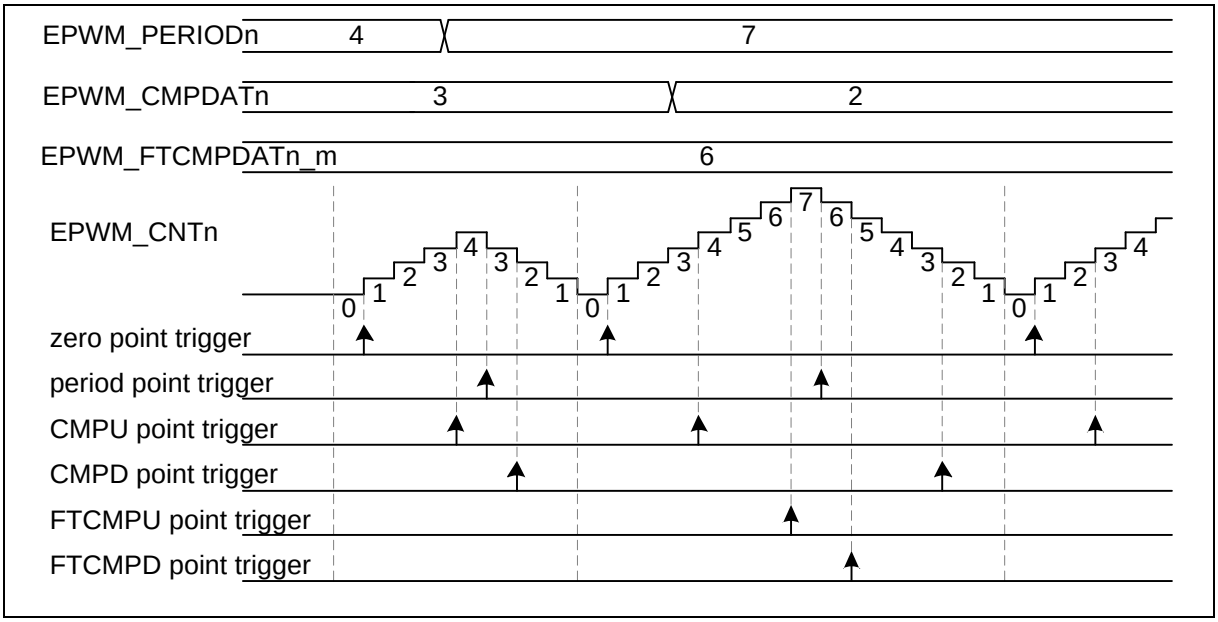


Figure 6.12-44 EPWM Trigger EADC in Up-Down Counter Type Timing Waveform

EPWM can also be used to trigger DAC conversion. Each EPWM pair channel (CH0 and CH1, CH2 and CH3, CH4 and CH5) generates a trigger signal. Using the EPWM Trigger DAC Enable Register (EPWM_DACTRGEN) can decide at which points to trigger DAC. The timing of the EPWM triggerring DAC is similar to those for triggerring EADC. However, DAC triggering function does not include the triggering events from comparison with FTCMPDAT, that is, there are no trigger points the same as FTCMPU and FTCMPD which are shown in EADC triggering.

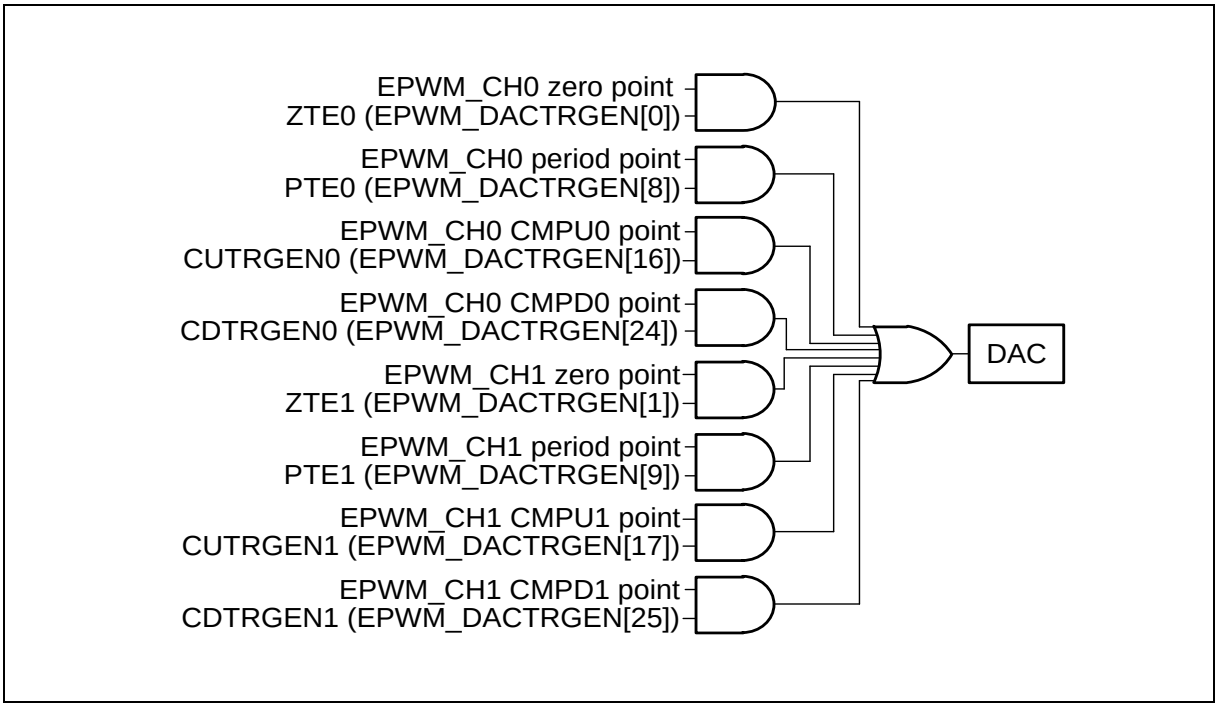


Figure 6.12-45 EPWM_CH0 and EPWM_CH1 Pair Trigger DAC Block Diagram

6.12.5.28 Capture Operation

The channels of the capture input and the EPWM output share the same pin and counter. The counter

can operating in up or down counter type. The capture input will be filtered by a 3-bit noise filter. User can enable the noise filter function by CAPNFEN (EPWM_CAPNFn[0], n=0~5) register, and noise filter sampling clock can be selected by setting CAPNFSEL bits of (EPWM_CAPNFn[6:4], n=0~5) register to fit different noise properties. Moreover, by setting the CAPNFCNT (EPWM_CAPNFn[10:8], n=0~5) bits, user can define how many sampling clock cycles a filter will recognize the effective edge of the capture input signal.

The capture function will always latch the EPWM counter to the RCAPDATn (EPWM_RCAPDATn[15:0]) bits or the FCAPDATn (EPWM_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 EPWM_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 (EPWM_CAPIEN[5:0]) bit is for the rising edge and the CAPFIENn (EPWM_CAPIEN[13:8]) bit is for the falling edge. When rising or falling latch occurs, the corresponding EPWM counter may be reloaded with the value of EPWM_PERIODn register, depending on the setting of RCRLDENn or FCRLDENn bits (where RCRLDENn and FCRLDENn are located at EPWM_CAPCTL[21:16] and EPWM_CAPCTL[29:24], respectively). Note that the corresponding GPIO pins must be configured as the capture function by enable the CAPINENn (EPWM_CAPINEN[5:0]) bits for the corresponding capture channel n. Figure 6.12-46 is the capture block diagram of channel 0.

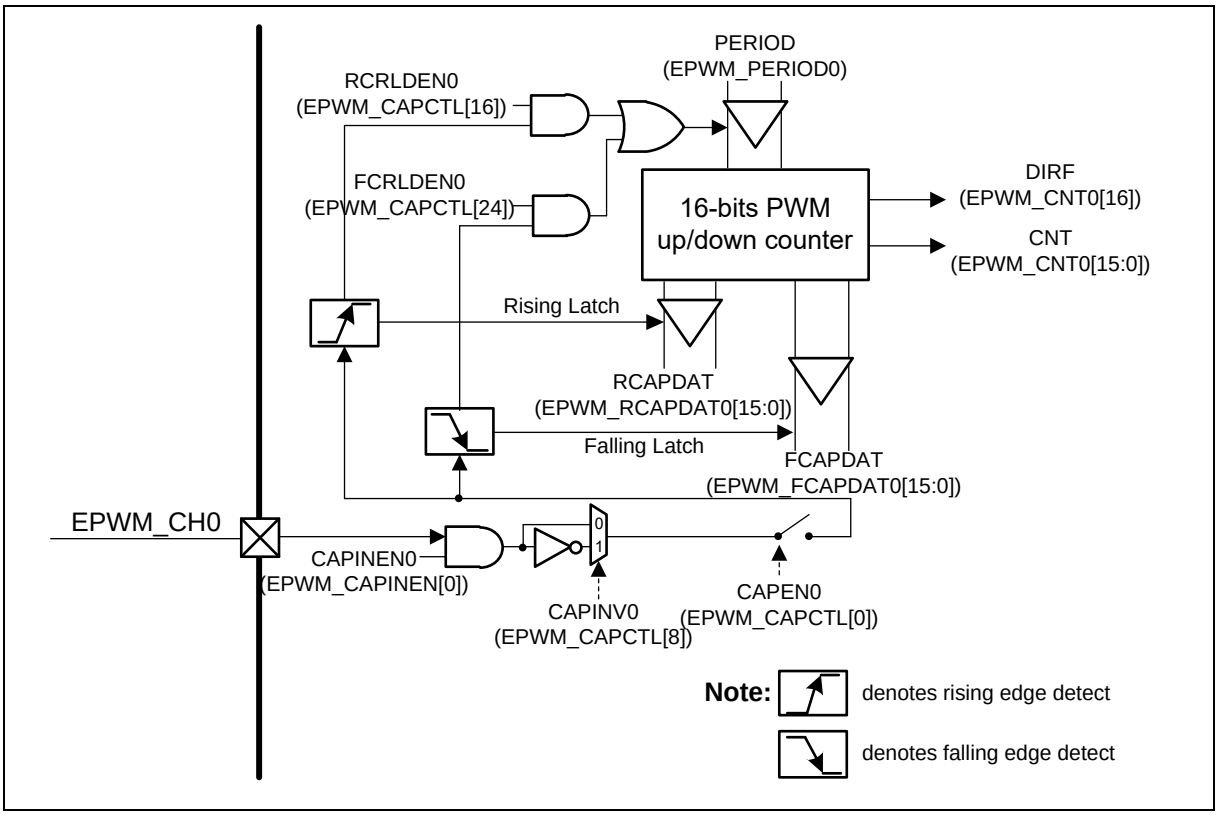


Figure 6.12-46 EPWM_CH0 Capture Block Diagram

Figure 6.12-47 illustrates the capture function timing. In this case, the capture counter is set as EPWM 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 EPWM_FCAPDATn register. When detecting the rising edge, it latches the counter value to the EPWM_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 6.12-47 also illustrates the timing example for the interrupt and interrupt flag generation. When the rising edge at channel n is detected, the corresponding CRLIFn (EPWM_CAPIF[5:0]) bit is set by hardware. Similarly, a falling edge detection at chnnel n causes the corresponding CFLIFn (EPWM_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 Over run status CRLIFOVn (EPWM_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 Over run status CFLIFOVn (EPWM_CAPSTS[13:8]).

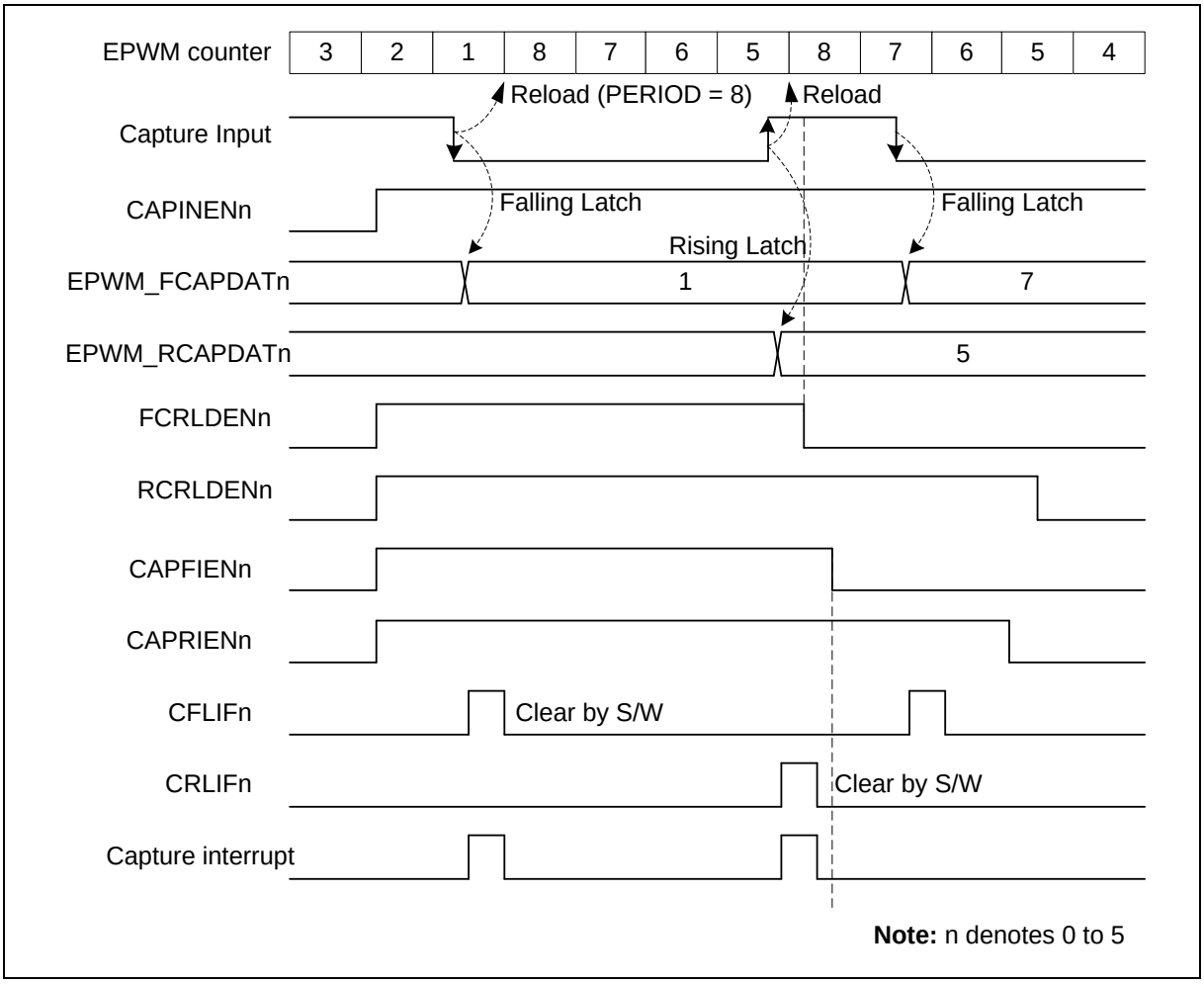


Figure 6.12-47 Capture Operation Waveform

The capture pulse width meeting the following conditions can be calculated according to the formula.

1. The capture positive or negative pulse width is shorter than a counter period.
2. The counter operates in down counter type.
3. The counter can be reloaded by both falling and rising capture events through setting FCRLDENn and RCRLDENn bits of PWM_CAPCTL register to 1.

For the negative pulse case, the channel low pulse width is calculated as (EPWM_PERIODn + 1 -

EPWM_RCAPDATn) EPWM counter time, where one EPWM counter time is (CLKPSC+1) * EPWMx_CLK clock time. In Figure 6.12-47, the low pulse width is 8+1-5 = 4 EPWM counter time.

For the positive pulse case, the channel high pulse width is calculated as (EPWM_PERIODn + 1 - EPWM_FCAPDATn) EPWM counter time, where one EPWM counter time is (CLKPSC+1) * EPWMx_CLK clock time. In Figure 6.12-47, the high pulse width is 8+1-7 = 2 EPWM counter time.

6.12.5.29 Capture PDMA Function

The EPWM module supports the PDMA transfer function when operating in the capture mode. When the corresponding PDMA enable bit CHENn_m (CHEN0_1 at EPWM_PDMACTL[0], CHEN2_3 at EPWM_PDMACTL[8] and CHEN4_5 at EPWM_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 (EPWM_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 EPWM_PDMACTL[2:1], CAPMOD2_3 at EPWM_PDMACTL[10:9] and CAPMOD4_5 at EPWM_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 EPWM_PDMACTL[3], CAPORD2_3 at EPWM_PDMACTL[11] and CAPORD4_5 at EPWM_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 (EPWM_PDMACTL[4]), CHSEL2_3 (EPWM_PDMACTL[12]) and CHSEL4_5 (EPWM_PDMACTL[20])) bit is used to decide either channel n or channel m can be serviced by the PDMA channel.

Figure 6.12-48 is capture PDMA waveform. In this case, the CHSEL0_1 (EPWM_PDMACTL[4]) bit is set to 0. Hence the PDMA will service channel 0 for the capture data transfer. CAPMOD0_1 (EPWM_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 (EPWM_PDMACTL[3]) 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 shown in Figure 6.12-48, the last assertions of the CRLIF0 and CFLIF0 signal have some overlap. The value of EPWM_RCAPDAT0 register is 11 will be loaded to EPWM_PDMACAP0_1 register to wait for transfer but not the EPWM_FCAPDAT0 value. The EPWM_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.

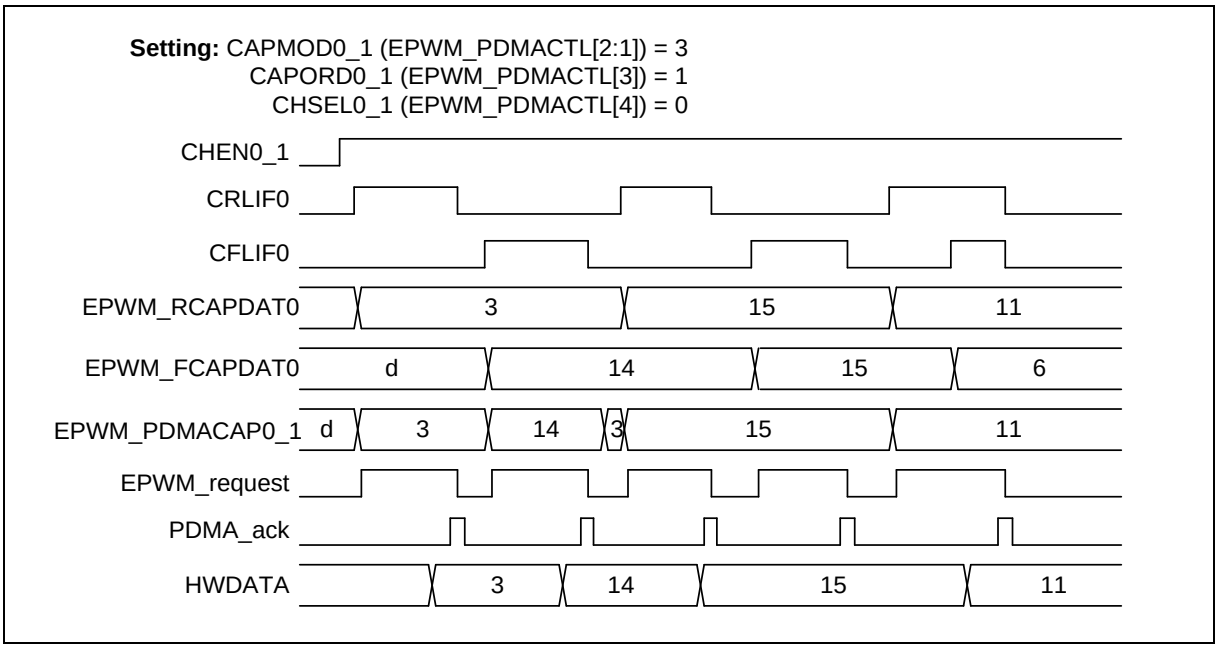


Figure 6.12-48 Capture PDMA Operation Waveform of Channel 0

6.12.5.30 Accumulator PDMA Function

The EPWM module supports the PDMA transfer function when accumulator interrupt happened. Figure 6.12-49 shows accumulator PDMA function architecture. When the corresponding PDMA enable bit APDMAENn (EPWM_APDMACTL[n], n=0~5) is set, accumulator module will send a request to PDMA controller when accumulator interrupt has happened, meaning that IFAIFn (EPWM_AINTSTS[n], n=0~5) is set 1. The PDMA controller will issue an acknowledge to accumulator after it has read memory data and send the data to the particular register (EPWM_PERIOD, etc.). So, user can use this function to change accumulator interrupt frequency.

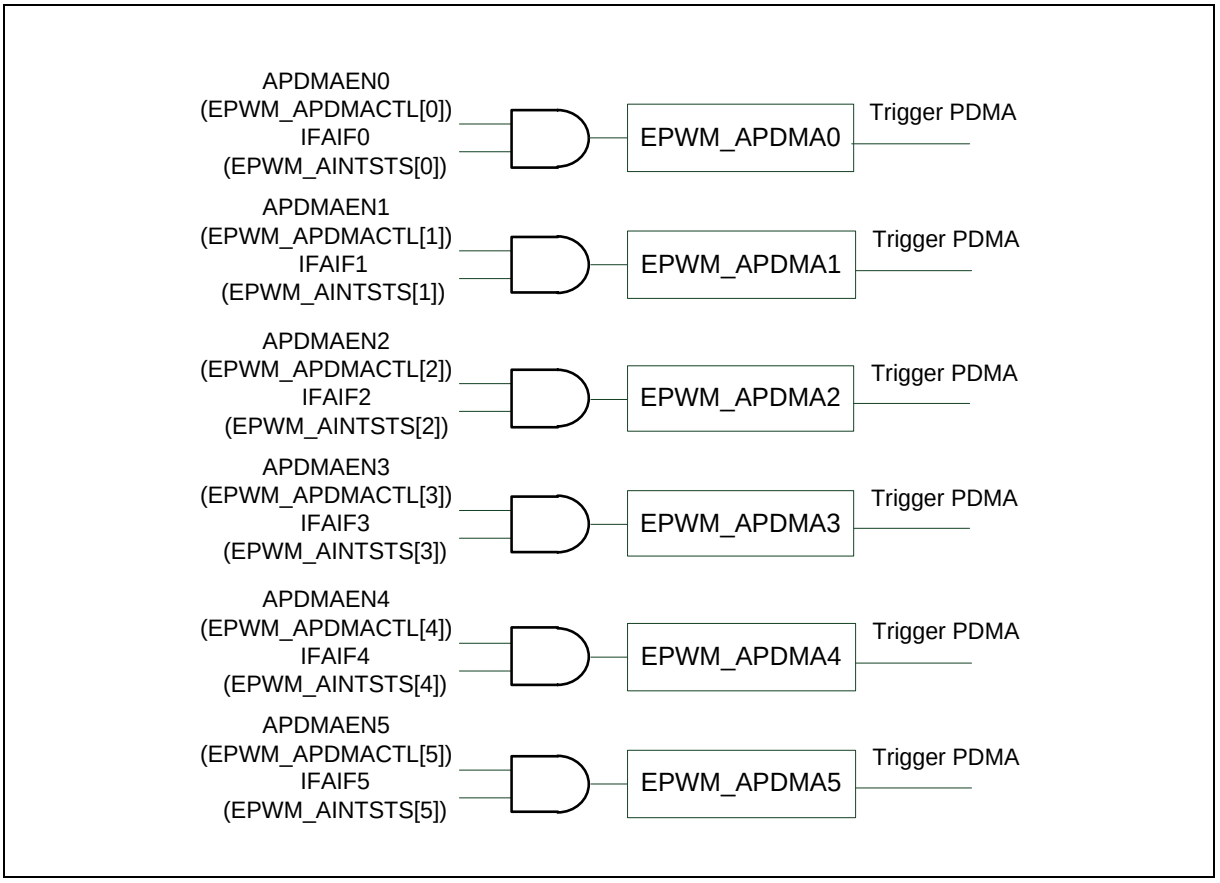


Figure 6.12-49 Accumulator PDMA Function Architecture

6.12.5.31 Accumulator Stop Mode

The EPWM module supports the Accumulator Stop Mode to stop counting when accumulator interrupt happened. Figure 6.12-50 shows Accumulator Stop Mode waveform. When the corresponding STPMOD (EPWM_IFAn[24], n=0~5) is set, accumulator module will disable CNTENn (EPWM_CNTEN[n], n=0~5) after counter finishs whole period.

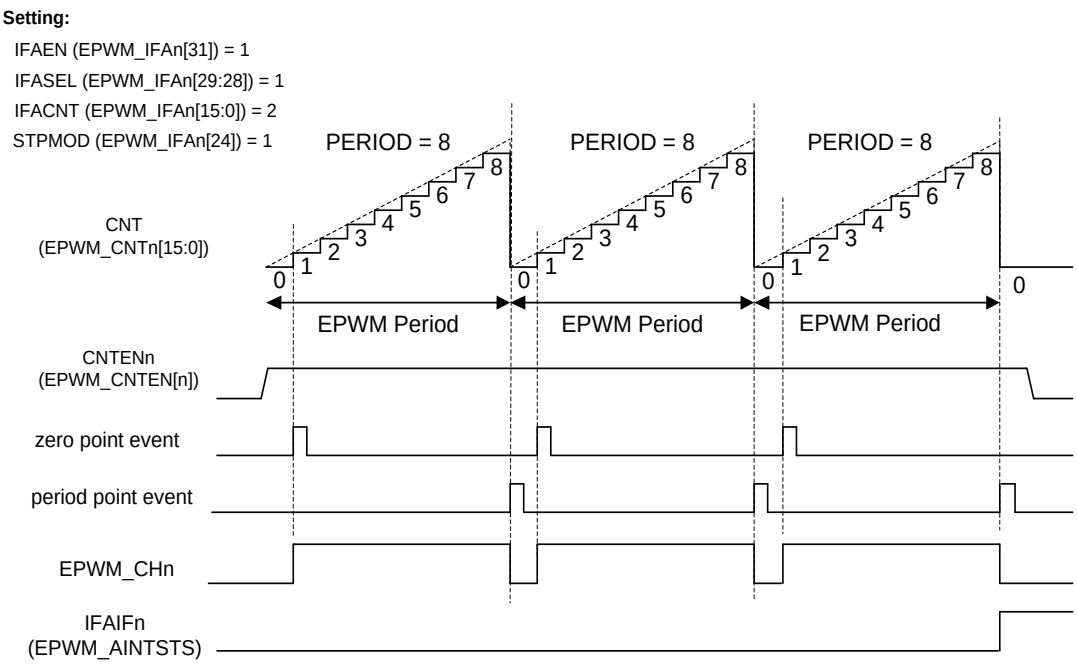


Figure 6.12-50 EPWM Accumulator Stop Mode Waveform

6.12.5.32 Fault Detect Function

The EPWM module supports Fault Detect Function to detect output short condition. Figure 6.12-51 shows fault detect function architecture. When the corresponding channel fault detect enable bit FDENn(EPWM_FDEN[n]) is set to 1, two statement will start detection. One is that EPWM output becomes from low to high or from high to low. The other is that POENn (EPWM_POEN[n], n=0~5) becomes from 0 to 1 or from 1 to 0.

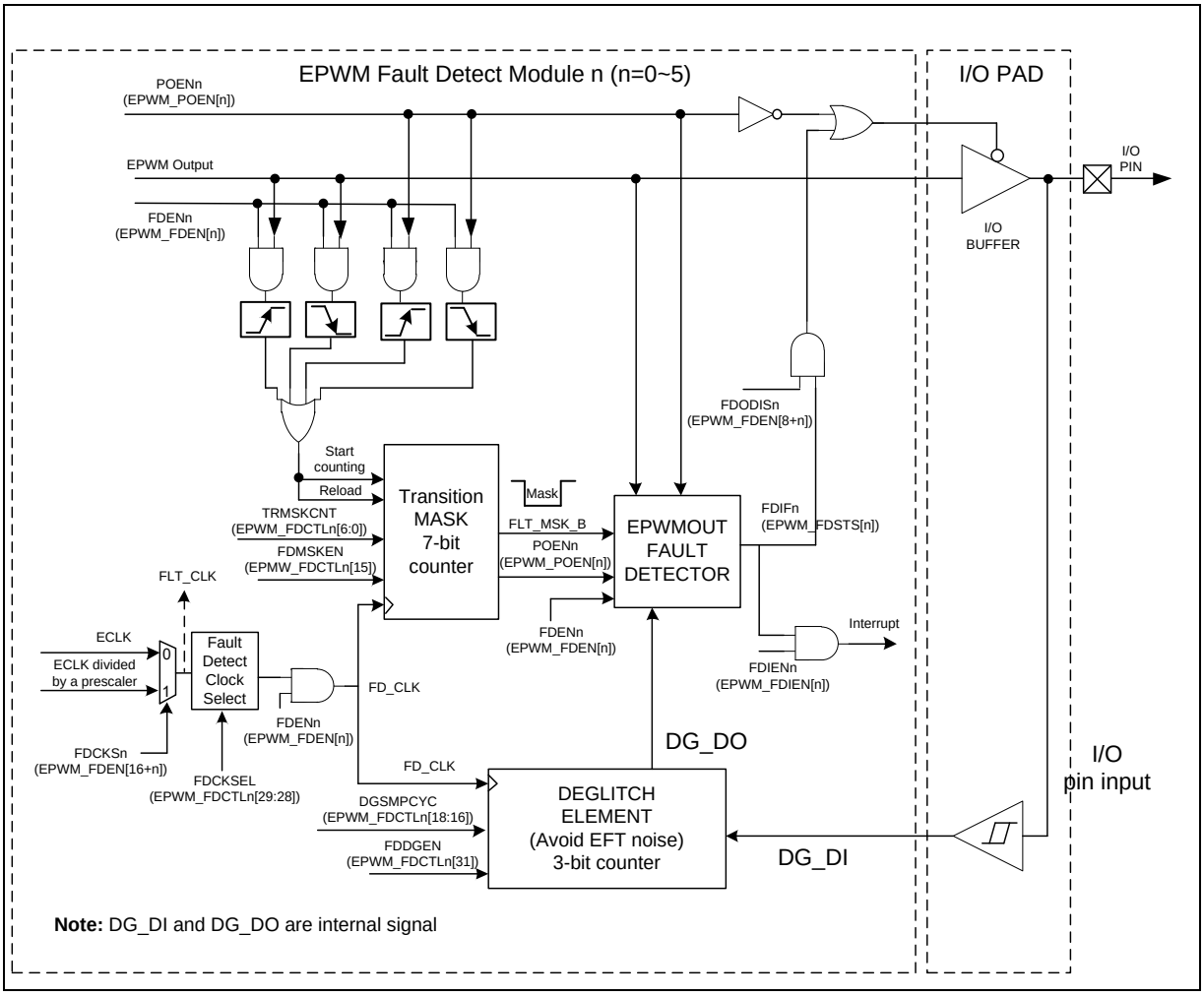


Figure 6.12-51 Fault Detect Function Architecture

Figure 6.12-52 shows Mask Function waveform example. Setting FDMSKEN (EPWM_FDCTLn[15], n=0~5) will enable the MASK Function. The detection will disable in the beginning until mask counter count from 0 to TRMSKCNT (EPWM_FDCTLn[6:0], n=0~5).

Figure 6.12-53 shows Deglitch Function waveform example. Setting FDDGEN (EPWM_FDCTLn[31], n=0~5) will enable Deglitch Function. The feedback signal will be sampled DGSMPCYC (EPWM_FDCTLn[18:16], n=0~5) +1 times and become valid input to be used for detection. FDCKSn (EPWM_FDEN[n], n=0~5) and FDCKSEL (EPWM_FDCTLn[29:28]) are clock setting for MASK and Dglitch Function. For more detail, please see the register discription.

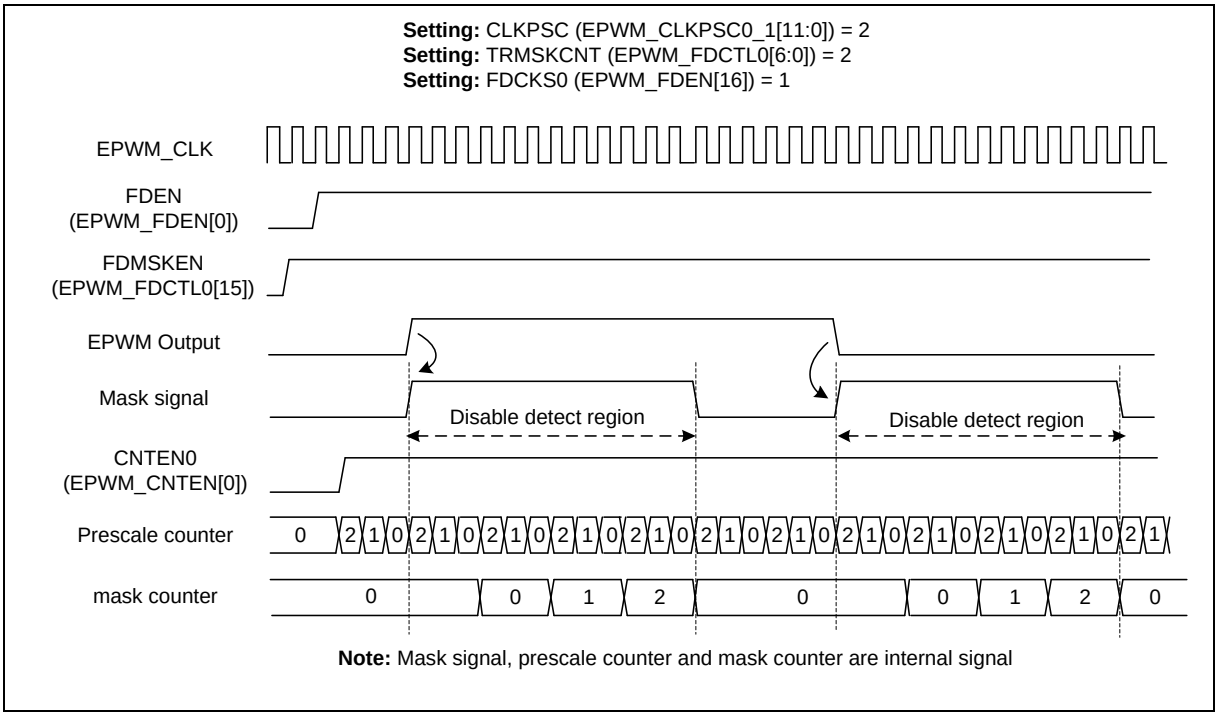


Figure 6.12-52 EPWM Mask Function Waveform

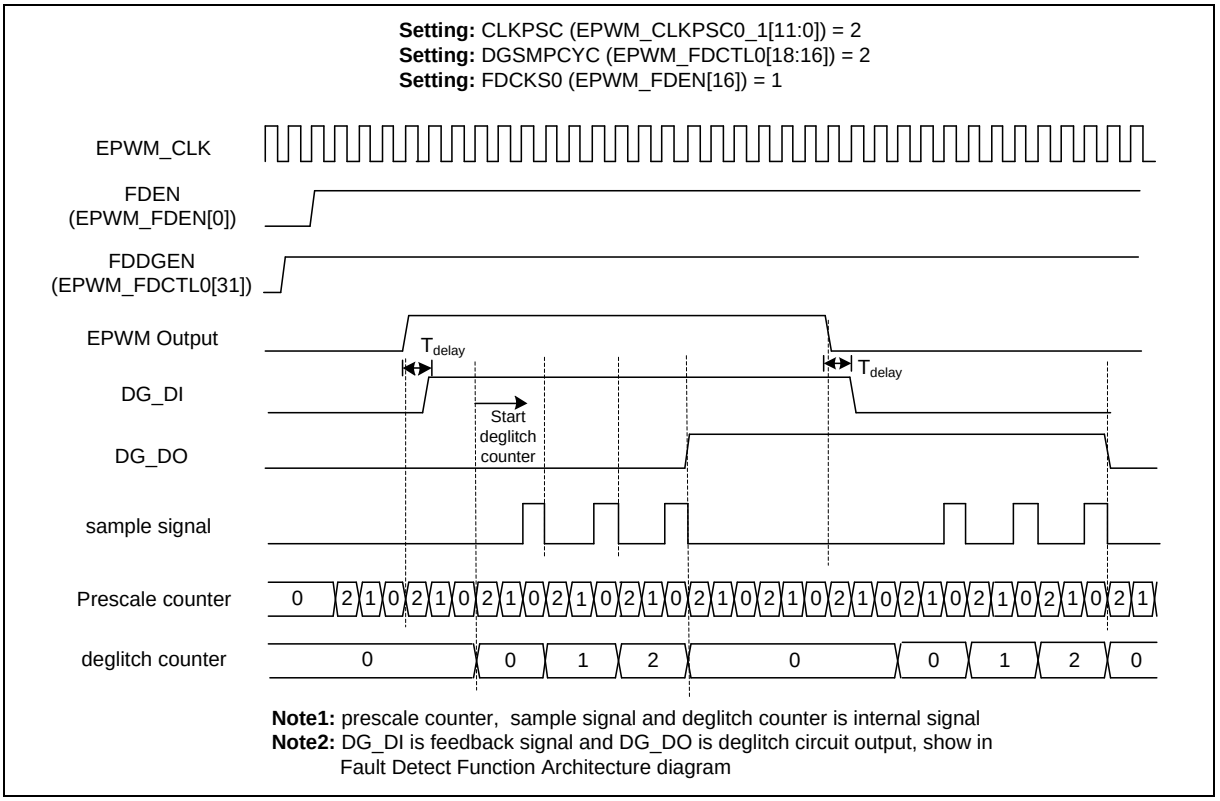


Figure 6.12-53 EPWM Deglitch Function Waveform

6.12.5.33 External Pin Event Trigger

External pin event can trigger EPWM counter. Setting EXTETEN (EPWM_EXTETCTLn[0], n=0~5) bit to 1 can enable this function. Setting CNTACTS (EPWM_EXTETCTLn[5:4], n=0~5) can choose counter action: reset, start or reset and start. Setting EXTTRGS (EPWM_EXTETCTLn[11:8], n=0~5) can select external trigger source from INT0~INT7.

Trigger event comes from detecting INT0~INT7 rising, falling or both of rising and falling. More detail setting can be found in “INT Edge Detect and Trigger Function” section of the GPIO chapter.

6.12.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
EPWM Base Address: **EPWM0_BA = 0x4005_8000** **EPWM1_BA = 0x4005_9000** **x=0,1**				
EPWM_CTL0 **x=0,1**	EPWMx_BA+0x00	R/W	EPWM Control Register 0	0x0000_0000
EPWM_CTL1 **x=0,1**	EPWMx_BA+0x04	R/W	EPWM Control Register 1	0x0000_0000
EPWM_SYNC **x=0,1**	EPWMx_BA+0x08	R/W	EPWM Synchronization Register	0x0000_0000
EPWM_SWSYNC **x=0,1**	EPWMx_BA+0x0C	R/W	EPWM Software Control Synchronization Register	0x0000_0000
EPWM_CLKSRC **x=0,1**	EPWMx_BA+0x10	R/W	EPWM Clock Source Register	0x0000_0000
EPWM_CLKPSC0_1 **x=0,1**	EPWMx_BA+0x14	R/W	EPWM Clock Prescale Register 0/1	0x0000_0000
EPWM_CLKPSC2_3 **x=0,1**	EPWMx_BA+0x18	R/W	EPWM Clock Prescale Register 2/3	0x0000_0000
EPWM_CLKPSC4_5 **x=0,1**	EPWMx_BA+0x1C	R/W	EPWM Clock Prescale Register 4/5	0x0000_0000
EPWM_CNTEN **x=0,1**	EPWMx_BA+0x20	R/W	EPWM Counter Enable Register	0x0000_0000
EPWM_CNTCLR **x=0,1**	EPWMx_BA+0x24	R/W	EPWM Clear Counter Register	0x0000_0000
EPWM_LOAD **x=0,1**	EPWMx_BA+0x28	R/W	EPWM Load Register	0x0000_0000
EPWM_PERIOD0 **x=0,1**	EPWMx_BA+0x30	R/W	EPWM Period Register 0	0x0000_0000
EPWM_PERIOD1 **x=0,1**	EPWMx_BA+0x34	R/W	EPWM Period Register 1	0x0000_0000
EPWM_PERIOD2 **x=0,1**	EPWMx_BA+0x38	R/W	EPWM Period Register 2	0x0000_0000
EPWM_PERIOD3 **x=0,1**	EPWMx_BA+0x3C	R/W	EPWM Period Register 3	0x0000_0000
EPWM_PERIOD4 **x=0,1**	EPWMx_BA+0x40	R/W	EPWM Period Register 4	0x0000_0000
EPWM_PERIOD5 **x=0,1**	EPWMx_BA+0x44	R/W	EPWM Period Register 5	0x0000_0000

EPWM_CMPDAT0 x=0,1	EPWMx_BA+0x50	R/W	EPWM Comparator Register 0	0x0000_0000
EPWM_CMPDAT1 x=0,1	EPWMx_BA+0x54	R/W	EPWM Comparator Register 1	0x0000_0000
EPWM_CMPDAT2 x=0,1	EPWMx_BA+0x58	R/W	EPWM Comparator Register 2	0x0000_0000
EPWM_CMPDAT3 x=0,1	EPWMx_BA+0x5C	R/W	EPWM Comparator Register 3	0x0000_0000
EPWM_CMPDAT4 x=0,1	EPWMx_BA+0x60	R/W	EPWM Comparator Register 4	0x0000_0000
EPWM_CMPDAT5 x=0,1	EPWMx_BA+0x64	R/W	EPWM Comparator Register 5	0x0000_0000
EPWM_DTCTL0_1 x=0,1	EPWMx_BA+0x70	R/W	EPWM Dead-time Control Register 0/1	0x0000_0000
EPWM_DTCTL2_3 x=0,1	EPWMx_BA+0x74	R/W	EPWM Dead-time Control Register 2/3	0x0000_0000
EPWM_DTCTL4_5 x=0,1	EPWMx_BA+0x78	R/W	EPWM Dead-time Control Register 4/5	0x0000_0000
EPWM_PHS0_1 x=0,1	EPWMx_BA+0x80	R/W	EPWM Counter Phase Register 0/1	0x0000_0000
EPWM_PHS2_3 x=0,1	EPWMx_BA+0x84	R/W	EPWM Counter Phase Register 2/3	0x0000_0000
EPWM_PHS4_5 x=0,1	EPWMx_BA+0x88	R/W	EPWM Counter Phase Register 4/5	0x0000_0000
EPWM_CNT0 x=0,1	EPWMx_BA+0x90	R	EPWM Counter Register 0	0x0000_0000
EPWM_CNT1 x=0,1	EPWMx_BA+0x94	R	EPWM Counter Register 1	0x0000_0000
EPWM_CNT2 x=0,1	EPWMx_BA+0x98	R	EPWM Counter Register 2	0x0000_0000
EPWM_CNT3 x=0,1	EPWMx_BA+0x9C	R	EPWM Counter Register 3	0x0000_0000
EPWM_CNT4 x=0,1	EPWMx_BA+0xA0	R	EPWM Counter Register 4	0x0000_0000
EPWM_CNT5 x=0,1	EPWMx_BA+0xA4	R	EPWM Counter Register 5	0x0000_0000
EPWM_WGCTL0 x=0,1	EPWMx_BA+0xB0	R/W	EPWM Generation Register 0	0x0000_0000
EPWM_WGCTL1 x=0,1	EPWMx_BA+0xB4	R/W	EPWM Generation Register 1	0x0000_0000
EPWM_MSKEN	EPWMx_BA+0xB8	R/W	EPWM Mask Enable Register	0x0000_0000

x=0,1				
EPWM_MSK **x=0,1**	EPWMx_BA+0xBC	R/W	EPWM Mask Data Register	0x0000_0000
EPWM_BNF **x=0,1**	EPWMx_BA+0xC0	R/W	EPWM Brake Noise Filter Register	0x0000_0000
EPWM_FAILBRK **x=0,1**	EPWMx_BA+0xC4	R/W	EPWM System Fail Brake Control Register	0x0000_0000
EPWM_BRKCTL0_1 **x=0,1**	EPWMx_BA+0xC8	R/W	EPWM Brake Edge Detect Control Register 0/1	0x0000_0000
EPWM_BRKCTL2_3 **x=0,1**	EPWMx_BA+0xCC	R/W	EPWM Brake Edge Detect Control Register 2/3	0x0000_0000
EPWM_BRKCTL4_5 **x=0,1**	EPWMx_BA+0xD0	R/W	EPWM Brake Edge Detect Control Register 4/5	0x0000_0000
EPWM_POLCTL **x=0,1**	EPWMx_BA+0xD4	R/W	EPWM Pin Polar Inverse Register	0x0000_0000
EPWM_POEN **x=0,1**	EPWMx_BA+0xD8	R/W	EPWM Output Enable Register	0x0000_0000
EPWM_SWBRK **x=0,1**	EPWMx_BA+0xDC	W	EPWM Software Brake Control Register	0x0000_0000
EPWM_INTEN0 **x=0,1**	EPWMx_BA+0xE0	R/W	EPWM Interrupt Enable Register 0	0x0000_0000
EPWM_INTEN1 **x=0,1**	EPWMx_BA+0xE4	R/W	EPWM Interrupt Enable Register 1	0x0000_0000
EPWM_INTSTS0 **x=0,1**	EPWMx_BA+0xE8	R/W	EPWM Interrupt Flag Register 0	0x0000_0000
EPWM_INTSTS1 **x=0,1**	EPWMx_BA+0xEC	R/W	EPWM Interrupt Flag Register 1	0x0000_0000
EPWM_DACTRGEN **x=0,1**	EPWMx_BA+0xF4	R/W	EPWM Trigger DAC Enable Register	0x0000_0000
EPWM_EADCTS0 **x=0,1**	EPWMx_BA+0xF8	R/W	EPWM Trigger EADC Source Select Register 0	0x0000_0000
EPWM_EADCTS1 **x=0,1**	EPWMx_BA+0xFC	R/W	EPWM Trigger EADC Source Select Register 1	0x0000_0000
EPWM_FTCMPDAT0_1 **x=0,1**	EPWMx_BA+0x100	R/W	EPWM Free Trigger Compare Register 0/1	0x0000_0000
EPWM_FTCMPDAT2_3 **x=0,1**	EPWMx_BA+0x104	R/W	EPWM Free Trigger Compare Register 2/3	0x0000_0000
EPWM_FTCMPDAT4_5 **x=0,1**	EPWMx_BA+0x108	R/W	EPWM Free Trigger Compare Register 4/5	0x0000_0000
EPWM_SSCTL **x=0,1**	EPWMx_BA+0x110	R/W	EPWM Synchronous Start Control Register	0x0000_0000

EPWM_SSTRG x=0,1	EPWMx_BA+0x114	W	EPWM Synchronous Start Trigger Register	0x0000_0000
EPWM_LEBCTL x=0,1	EPWMx_BA+0x118	R/W	EPWM Leading Edge Blanking Control Register	0x0000_0000
EPWM_LEBCNT x=0,1	EPWMx_BA+0x11C	R/W	EPWM Leading Edge Blanking Counter Register	0x0000_0000
EPWM_STATUS x=0,1	EPWMx_BA+0x120	R/W	EPWM Status Register	0x0000_0000
EPWM_IFA0 x=0,1	EPWMx_BA+0x130	R/W	EPWM Interrupt Flag Accumulator Register 0	0x0000_0000
EPWM_IFA1 x=0,1	EPWMx_BA+0x134	R/W	EPWM Interrupt Flag Accumulator Register 1	0x0000_0000
EPWM_IFA2 x=0,1	EPWMx_BA+0x138	R/W	EPWM Interrupt Flag Accumulator Register 2	0x0000_0000
EPWM_IFA3 x=0,1	EPWMx_BA+0x13C	R/W	EPWM Interrupt Flag Accumulator Register 3	0x0000_0000
EPWM_IFA4 x=0,1	EPWMx_BA+0x140	R/W	EPWM Interrupt Flag Accumulator Register 4	0x0000_0000
EPWM_IFA5 x=0,1	EPWMx_BA+0x144	R/W	EPWM Interrupt Flag Accumulator Register 5	0x0000_0000
EPWM_AINTSTS x=0,1	EPWMx_BA+0x150	R/W	EPWM Accumulator Interrupt Flag Register	0x0000_0000
EPWM_AINTEN x=0,1	EPWMx_BA+0x154	R/W	EPWM Accumulator Interrupt Enable Register	0x0000_0000
EPWM_APDMACTL x=0,1	EPWMx_BA+0x158	R/W	EPWM Accumulator PDMA Control Register	0x0000_0000
EPWM_FDEN x=0,1	EPWMx_BA+0x160	R/W	EPWM Fault Detect Enable Register	0x0000_0000
EPWM_FDCTL0 x=0,1	EPWMx_BA+0x164	R/W	EPWM Fault Detect Control Register 0	0x0000_0000
EPWM_FDCTL1 x=0,1	EPWMx_BA+0x168	R/W	EPWM Fault Detect Control Register 1	0x0000_0000
EPWM_FDCTL2 x=0,1	EPWMx_BA+0x16C	R/W	EPWM Fault Detect Control Register 2	0x0000_0000
EPWM_FDCTL3 x=0,1	EPWMx_BA+0x170	R/W	EPWM Fault Detect Control Register 3	0x0000_0000
EPWM_FDCTL4 x=0,1	EPWMx_BA+0x174	R/W	EPWM Fault Detect Control Register 4	0x0000_0000
EPWM_FDCTL5 x=0,1	EPWMx_BA+0x178	R/W	EPWM Fault Detect Control Register 5	0x0000_0000
EPWM_FDIEN	EPWMx_BA+0x17C	R/W	EPWM Fault Detect Interrupt Enable Register	0x0000_0000

x=0,1				
EPWM_FDSTS x=0,1	EPWMx_BA+0x180	R/W	EPWM Fault Detect Interrupt Flag Register	0x0000_0000
EPWM_EADCPSCCTL x=0,1	EPWMx_BA+0x184	R/W	EPWM Trigger EADC Prescale Control Register	0x0000_0000
EPWM_EADCPSC0 x=0,1	EPWMx_BA+0x188	R/W	EPWM Trigger EADC Prescale Register 0	0x0000_0000
EPWM_EADCPSC1 x=0,1	EPWMx_BA+0x18C	R/W	EPWM Trigger EADC Prescale Register 1	0x0000_0000
EPWM_EADCPSCNT0 x=0,1	EPWMx_BA+0x190	R/W	EPWM Trigger EADC Prescale Counter Register 0	0x0000_0000
EPWM_EADCPSCNT1 x=0,1	EPWMx_BA+0x194	R/W	EPWM Trigger EADC Prescale Counter Register 1	0x0000_0000
EPWM_CAPINEN x=0,1	EPWMx_BA+0x200	R/W	EPWM Capture Input Enable Register	0x0000_0000
EPWM_CAPCTL x=0,1	EPWMx_BA+0x204	R/W	EPWM Capture Control Register	0x0000_0000
EPWM_CAPSTS x=0,1	EPWMx_BA+0x208	R	EPWM Capture Status Register	0x0000_0000
EPWM_RCAPDAT0 x=0,1	EPWMx_BA+0x20C	R	EPWM Rising Capture Data Register 0	0x0000_0000
EPWM_FCAPDAT0 x=0,1	EPWMx_BA+0x210	R	EPWM Falling Capture Data Register 0	0x0000_0000
EPWM_RCAPDAT1 x=0,1	EPWMx_BA+0x214	R	EPWM Rising Capture Data Register 1	0x0000_0000
EPWM_FCAPDAT1 x=0,1	EPWMx_BA+0x218	R	EPWM Falling Capture Data Register 1	0x0000_0000
EPWM_RCAPDAT2 x=0,1	EPWMx_BA+0x21C	R	EPWM Rising Capture Data Register 2	0x0000_0000
EPWM_FCAPDAT2 x=0,1	EPWMx_BA+0x220	R	EPWM Falling Capture Data Register 2	0x0000_0000
EPWM_RCAPDAT3 x=0,1	EPWMx_BA+0x224	R	EPWM Rising Capture Data Register 3	0x0000_0000
EPWM_FCAPDAT3 x=0,1	EPWMx_BA+0x228	R	EPWM Falling Capture Data Register 3	0x0000_0000
EPWM_RCAPDAT4 x=0,1	EPWMx_BA+0x22C	R	EPWM Rising Capture Data Register 4	0x0000_0000
EPWM_FCAPDAT4 x=0,1	EPWMx_BA+0x230	R	EPWM Falling Capture Data Register 4	0x0000_0000
EPWM_RCAPDAT5 x=0,1	EPWMx_BA+0x234	R	EPWM Rising Capture Data Register 5	0x0000_0000

EPWM_FCAPDAT5 x=0,1	EPWMx_BA+0x238	R	EPWM Falling Capture Data Register 5	0x0000_0000
EPWM_PDMACTL x=0,1	EPWMx_BA+0x23C	R/W	EPWM PDMA Control Register	0x0000_0000
EPWM_PDMACAP0_1 x=0,1	EPWMx_BA+0x240	R	EPWM Capture Channel 01 PDMA Register	0x0000_0000
EPWM_PDMACAP2_3 x=0,1	EPWMx_BA+0x244	R	EPWM Capture Channel 23 PDMA Register	0x0000_0000
EPWM_PDMACAP4_5 x=0,1	EPWMx_BA+0x248	R	EPWM Capture Channel 45 PDMA Register	0x0000_0000
EPWM_CAPIEN x=0,1	EPWMx_BA+0x250	R/W	EPWM Capture Interrupt Enable Register	0x0000_0000
EPWM_CAPIF x=0,1	EPWMx_BA+0x254	R/W	EPWM Capture Interrupt Flag Register	0x0000_0000
EPWM_CAPNF0 x=0,1	EPWMx_BA+0x258	R/W	EPWM Capture Input Noise Filter Register 0	0x0000_0000
EPWM_CAPNF1 x=0,1	EPWMx_BA+0x25C	R/W	EPWM Capture Input Noise Filter Register 1	0x0000_0000
EPWM_CAPNF2 x=0,1	EPWMx_BA+0x260	R/W	EPWM Capture Input Noise Filter Register 2	0x0000_0000
EPWM_CAPNF3 x=0,1	EPWMx_BA+0x264	R/W	EPWM Capture Input Noise Filter Register 3	0x0000_0000
EPWM_CAPNF4 x=0,1	EPWMx_BA+0x268	R/W	EPWM Capture Input Noise Filter Register 4	0x0000_0000
EPWM_CAPNF5 x=0,1	EPWMx_BA+0x26C	R/W	EPWM Capture Input Noise Filter Register 5	0x0000_0000
EPWM_EXTETCTL0 x=0,1	EPWMx_BA+0x270	R/W	EPWM External Event Trigger Control Register 0	0x0000_0000
EPWM_EXTETCTL1 x=0,1	EPWMx_BA+0x274	R/W	EPWM External Event Trigger Control Register 1	0x0000_0000
EPWM_EXTETCTL2 x=0,1	EPWMx_BA+0x278	R/W	EPWM External Event Trigger Control Register 2	0x0000_0000
EPWM_EXTETCTL3 x=0,1	EPWMx_BA+0x27C	R/W	EPWM External Event Trigger Control Register 3	0x0000_0000
EPWM_EXTETCTL4 x=0,1	EPWMx_BA+0x280	R/W	EPWM External Event Trigger Control Register 4	0x0000_0000
EPWM_EXTETCTL5 x=0,1	EPWMx_BA+0x284	R/W	EPWM External Event Trigger Control Register 5	0x0000_0000
EPWM_SWEOFCTL x=0,1	EPWMx_BA+0x288	R/W	EPWM Software Event Output Force Control Register	0x0000_0000
EPWM_SWEOFTRG	EPWMx_BA+0x28C	R/W	EPWM Software Event Output Force Trigger Register	0x0000_0000

x=0,1				
EPWM_CLKPSC0 x=0,1	EPWMx_BA+0x290	R/W	EPWM Clock Prescale Register 0	0x0000_0000
EPWM_CLKPSC1 x=0,1	EPWMx_BA+0x294	R/W	EPWM Clock Prescale Register 1	0x0000_0000
EPWM_CLKPSC2 x=0,1	EPWMx_BA+0x298	R/W	EPWM Clock Prescale Register 2	0x0000_0000
EPWM_CLKPSC3 x=0,1	EPWMx_BA+0x29C	R/W	EPWM Clock Prescale Register 3	0x0000_0000
EPWM_CLKPSC4 x=0,1	EPWMx_BA+0x2A0	R/W	EPWM Clock Prescale Register 4	0x0000_0000
EPWM_CLKPSC5 x=0,1	EPWMx_BA+0x2A4	R/W	EPWM Clock Prescale Register 5	0x0000_0000
EPWM_RDTCNT0_1 x=0,1	EPWMx_BA+0x2A8	R/W	EPWM Rising Dead-time Counter Register 0/1	0x0000_0000
EPWM_RDTCNT2_3 x=0,1	EPWMx_BA+0x2AC	R/W	EPWM Rising Dead-time Counter Register 2/3	0x0000_0000
EPWM_RDTCNT4_5 x=0,1	EPWMx_BA+0x2B0	R/W	EPWM Rising Dead-time Counter Register 4/5	0x0000_0000
EPWM_FDTCNT0_1 x=0,1	EPWMx_BA+0x2B4	R/W	EPWM Falling Dead-time Counter Register 0/1	0x0000_0000
EPWM_FDTCNT2_3 x=0,1	EPWMx_BA+0x2B8	R/W	EPWM Falling Dead-time Counter Register 2/3	0x0000_0000
EPWM_FDTCNT4_5 x=0,1	EPWMx_BA+0x2BC	R/W	EPWM Falling Dead-time Counter Register 4/5	0x0000_0000
EPWM_DTCTL x=0,1	EPWMx_BA+0x2C0	R/W	EPWM Dead-time Control Register	0x0000_0000
EPWM_PBUF0 x=0,1	EPWMx_BA+0x304	R	EPWM PERIOD0 Buffer	0x0000_0000
EPWM_PBUF1 x=0,1	EPWMx_BA+0x308	R	EPWM PERIOD1 Buffer	0x0000_0000
EPWM_PBUF2 x=0,1	EPWMx_BA+0x30C	R	EPWM PERIOD2 Buffer	0x0000_0000
EPWM_PBUF3 x=0,1	EPWMx_BA+0x310	R	EPWM PERIOD3 Buffer	0x0000_0000
EPWM_PBUF4 x=0,1	EPWMx_BA+0x314	R	EPWM PERIOD4 Buffer	0x0000_0000
EPWM_PBUF5 x=0,1	EPWMx_BA+0x318	R	EPWM PERIOD5 Buffer	0x0000_0000
EPWM_CMPBUF0 x=0,1	EPWMx_BA+0x31C	R	EPWM CMPDAT0 Buffer	0x0000_0000

EPWM_CMPBUF1 x=0,1	EPWMx_BA+0x320	R	EPWM CMPDAT1 Buffer	0x0000_0000
EPWM_CMPBUF2 x=0,1	EPWMx_BA+0x324	R	EPWM CMPDAT2 Buffer	0x0000_0000
EPWM_CMPBUF3 x=0,1	EPWMx_BA+0x328	R	EPWM CMPDAT3 Buffer	0x0000_0000
EPWM_CMPBUF4 x=0,1	EPWMx_BA+0x32C	R	EPWM CMPDAT4 Buffer	0x0000_0000
EPWM_CMPBUF5 x=0,1	EPWMx_BA+0x330	R	EPWM CMPDAT5 Buffer	0x0000_0000
EPWM_CPSCBUF0_1 x=0,1	EPWMx_BA+0x334	R	EPWM CLKPSC0_1 Buffer	0x0000_0000
EPWM_CPSCBUF2_3 x=0,1	EPWMx_BA+0x338	R	EPWM CLKPSC2_3 Buffer	0x0000_0000
EPWM_CPSCBUF4_5 x=0,1	EPWMx_BA+0x33C	R	EPWM CLKPSC4_5 Buffer	0x0000_0000
EPWM_FTCBUF0_1 x=0,1	EPWMx_BA+0x340	R	EPWM FTCMPDAT0_1 Buffer	0x0000_0000
EPWM_FTCBUF2_3 x=0,1	EPWMx_BA+0x344	R	EPWM FTCMPDAT2_3 Buffer	0x0000_0000
EPWM_FTCBUF4_5 x=0,1	EPWMx_BA+0x348	R	EPWM FTCMPDAT4_5 Buffer	0x0000_0000
EPWM_FTCI x=0,1	EPWMx_BA+0x34C	R/W	EPWM FTCMPDAT Indicator Register	0x0000_0000
EPWM_CPSCBUF0 x=0,1	EPWMx_BA+0x350	R	EPWM CLKPSC0 Buffer	0x0000_0000
EPWM_CPSCBUF1 x=0,1	EPWMx_BA+0x354	R	EPWM CLKPSC1 Buffer	0x0000_0000
EPWM_CPSCBUF2 x=0,1	EPWMx_BA+0x358	R	EPWM CLKPSC2 Buffer	0x0000_0000
EPWM_CPSCBUF3 x=0,1	EPWMx_BA+0x35C	R	EPWM CLKPSC3 Buffer	0x0000_0000
EPWM_CPSCBUF4 x=0,1	EPWMx_BA+0x360	R	EPWM CLKPSC4 Buffer	0x0000_0000
EPWM_CPSCBUF5 x=0,1	EPWMx_BA+0x364	R	EPWM CLKPSC5 Buffer	0x0000_0000
EPWM_IFACNT0 x=0,1	EPWMx_BA+0x368	R	EPWM Interrupt Flag Accumulator Counter 0	0x0000_0000
EPWM_IFACNT1 x=0,1	EPWMx_BA+0x36C	R	EPWM Interrupt Flag Accumulator Counter 1	0x0000_0000
EPWM_IFACNT2	EPWMx_BA+0x370	R	EPWM Interrupt Flag Accumulator Counter 2	0x0000_0000

x=0,1				
EPWM_IFACNT3 x=0,1	EPWMx_BA+0x374	R	EPWM Interrupt Flag Accumulator Counter 3	0x0000_0000
EPWM_IFACNT4 x=0,1	EPWMx_BA+0x378	R	EPWM Interrupt Flag Accumulator Counter 4	0x0000_0000
EPWM_IFACNT5 x=0,1	EPWMx_BA+0x37C	R	EPWM Interrupt Flag Accumulator Counter 5	0x0000_0000

6.12.7 Register Description

EPWM Control Register 0 (EPWM_CTL0)

Register	Offset	R/W	Description	Reset Value
EPWM_CTL0	EPWMx_BA+0x00	R/W	EPWM 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 Bit (Write Protect)** 0 = ICE debug mode acknowledgement effects EPWM output. EPWM pin will be forced as tri-state while ICE debug mode acknowledged. 1 = ICE debug mode acknowledgement disabled. EPWM pin will keep output no matter ICE debug mode acknowledged or not. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[30]	**DBGHALT**	**ICE Debug Mode Counter Halt (Write Protect)** If counter halt is enabled, EPWM all counters will keep current value until exit ICE debug mode. 0 = ICE debug mode counter halt Disabled. 1 = ICE debug mode counter halt Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[29:26]	**Reserved**	Reserved.
[24]	**GROUPEN**	**Group Function Enable Bit** 0 = The output waveform of each EPWM channel are independent. 1 = Unify the EPWM_CH2 and EPWM_CH4 to output the same waveform as EPWM_CH0 and unify the EPWM_CH3 and EPWM_CH5 to output the same waveform as EPWM_CH1.
[23:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**IMMLDENn**	**Immediately Load Enable Bits** 0 = PERIOD will load to PBUF at the end point of each period. CMP will load to CMPBUF at the end point or center point of each period by setting CTRLD bit. 1 = PERIOD/CMP will load to PBUF and CMPBUF immediately when software update PERIOD/CMP. **Note:** If IMMLDENn is enabled, WINLDENn and CTRLDn will be invalid.
[15:14]	**Reserved**	Reserved.
[8+n] n=0,1..5	**WINLDENn**	**Window Load Enable Bits** 0 = PERIOD will load to PBUF at the end point of each period. CMP 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. CMP 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 EPWM_LOAD register and cleared by hardware after load success.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**CTRLDn**	**Center Re-load** In up-down counter type, PERIOD will load to PBUF at the end point of each period. CMP will load to CMPBUF at the center point of a period.

EPWM Control Register 1 (EPWM_CTL1)

Register	Offset	R/W	Description	Reset Value
EPWM_CTL1	EPWMx_BA+0x04	R/W	EPWM 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.
[24+n/2] n=0,2,4	**OUTMODEn**	**EPWM Output Mode** Each bit n controls the output mode of corresponding EPWM channel n. 0 = EPWM independent mode. 1 = EPWM complementary mode. **Note:** When operating in group function, these bits must all set to the same mode.
[23:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**CNTMODEn**	**EPWM Counter Mode** 0 = Auto-reload mode. 1 = One-shot mode.
[15:12]	**Reserved**	Reserved.
[2n+1:2n] n=0,1..5	**CNTTYPEn**	**EPWM Counter Behavior Type** 00 = Up counter type (supported in capture mode). 01 = Down count type (supported in capture mode). 10 = Up-down counter type. 11 = Reserved.

EPWM Synchronization Register (EPWM_SYNC)

Register	Offset	R/W	Description	Reset Value
EPWM_SYNC	EPWMx_BA+0x08	R/W	EPWM 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.
[24+n/2] n=0,2,4	**PHSDIRn**	**EPWM Phase Direction Control** 0 = Control EPWM counter count decrement after synchronizing. 1 = Control EPWM 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**	**EPWM0_SYNC_IN Noise Filter Enable Bits** 0 = Noise filter of input pin EPWM0_SYNC_IN Disabled. 1 = Noise filter of input pin EPWM0_SYNC_IN Enabled.
[15:14]	**Reserved**	Reserved.
[9+n:8+n] n=0,2,4	**SINSRCn**	**EPWM0_SYNC_IN Source Selection** 00 = Synchronize source from SYNC_IN or SWSYNC. 01 = Counter equal to 0.

		10 = Counter equal to EPWM_CMPDATm, m denotes 1, 3, 5. 11 = SYNC_OUT will not be generated.
[7:3]	**Reserved**	Reserved.
[n/2] n=0,2,4	**PHSENn**	**SYNC Phase Enable Bits** 0 = EPWM counter disabled to load PHS value. 1 = EPWM counter enabled to load PHS value.

EPWM Software Control Synchronization Register (EPWM_SWSYNC)

Register	Offset	R/W	Description	Reset Value
EPWM_SWSYNC	EPWMx_BA+0x0C	R/W	EPWM 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.
[n/2] n=0,2,4	**SWSYNCn**	**Software SYNC Function** When SINSRCn (EPWM_SYNC[13:8]) is selected to 0, SYNC_OUT source comes from SYNC_IN or this bit.

EPWM Clock Source Register (EPWM_CLKSRC)

Register	Offset	R/W	Description	Reset Value
EPWM_CLKSRC	EPWMx_BA+0x10	R/W	EPWM Clock Source Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved					**ECLKSRC4**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**ECLKSRC2**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**ECLKSRC0**		

Bits	Description	
[31:19]	**Reserved**	Reserved.
[18:16]	**ECLKSRC4**	**EPWM_CH45 External Clock Source Select** 000 = EPWMx_CLK, x denotes 0 or 1. 001 = TIMER0 overflow. 010 = TIMER1 overflow. 011 = TIMER2 overflow. 100 = TIMER3 overflow. Others = Reserved.
[15:11]	**Reserved**	Reserved.
[10:8]	**ECLKSRC2**	**EPWM_CH23 External Clock Source Select** 000 = EPWMx_CLK, x denotes 0 or 1. 001 = TIMER0 overflow. 010 = TIMER1 overflow. 011 = TIMER2 overflow. 100 = TIMER3 overflow. Others = Reserved.
[7:3]	**Reserved**	Reserved.
[2:0]	**ECLKSRC0**	**EPWM_CH01 External Clock Source Select** 000 = EPWMx_CLK, x denotes 0 or 1. 001 = TIMER0 overflow. 010 = TIMER1 overflow. 011 = TIMER2 overflow. 100 = TIMER3 overflow. Others = Reserved.

EPWM Clock Prescale Register 0_1, 2_3, 4_5 (EPWM_CLKPSC0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
EPWM_CLKPSC0_1	EPWMx_BA+0x14	R/W	EPWM Clock Prescale Register 0/1	0x0000_0000
EPWM_CLKPSC2_3	EPWMx_BA+0x18	R/W	EPWM Clock Prescale Register 2/3	0x0000_0000
EPWM_CLKPSC4_5	EPWMx_BA+0x1C	R/W	EPWM Clock Prescale Register 4/5	0x0000_0000

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

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

EPWM Counter Enable Register (EPWM_CNTEN)

Register	Offset	R/W	Description	Reset Value
EPWM_CNTEN	EPWMx_BA+0x20	R/W	EPWM 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.
[n] n=0,1..5	CNTENn	**EPWM Counter Enable Bits** 0 = EPWM Counter and clock prescaler stop running. 1 = EPWM Counter and clock prescaler start running. **Note:** In complementary mode, not only even channels, but also odd channels need to be enabled at the same time.

EPWM Clear Counter Register (EPWM_CNTCLR)

Register	Offset	R/W	Description	Reset Value
EPWM_CNTCLR	EPWMx_BA+0x24	R/W	EPWM 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.
[n] n=0,1..5	CNTCLRn	**Clear EPWM Counter Control Bit** It is automatically cleared by hardware. Each bit n controls the corresponding EPWM channel n. 0 = No effect. 1 = Clear 16-bit EPWM counter to 0000H.

EPWM Load Register (EPWM_LOAD)

Register	Offset	R/W	Description	Reset Value
EPWM_LOAD	EPWMx_BA+0x28	R/W	EPWM 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.
[n] n=0,1..5	**LOADn**	**Re-load EPWM Comparator Register (EPWM_CMPDATn) Control Bit** This bit is software write, hardware clear when current EPWM 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, WINLDENn(EPWM_CTL0[13:8]) = 1.

EPWM Period Register 0~5 (EPWM_PERIOD0~5)

Register	Offset	R/W	Description	Reset Value
EPWM_PERIOD0	EPWMx_BA+0x30	R/W	EPWM Period Register 0	0x0000_0000
EPWM_PERIOD1	EPWMx_BA+0x34	R/W	EPWM Period Register 1	0x0000_0000
EPWM_PERIOD2	EPWMx_BA+0x38	R/W	EPWM Period Register 2	0x0000_0000
EPWM_PERIOD3	EPWMx_BA+0x3C	R/W	EPWM Period Register 3	0x0000_0000
EPWM_PERIOD4	EPWMx_BA+0x40	R/W	EPWM Period Register 4	0x0000_0000
EPWM_PERIOD5	EPWMx_BA+0x44	R/W	EPWM 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.
[15:0]	**PERIOD**	**EPWM Period Register** Up-Count mode: In this mode, EPWM counter counts from 0 to PERIOD, and restarts from 0. EPWM period time = (PERIOD+1) * (CLKPSC+1) * EPWM_CLK . Down-Count mode: In this mode, EPWM counter counts from PERIOD to 0, and restarts from PERIOD. EPWM period time = (PERIOD+1) * (CLKPSC+1) * EPWM_CLK . Up-Down-Count mode: In this mode, EPWM counter counts from 0 to PERIOD, then decrements to 0 and repeats again. EPWM period time = 2 * PERIOD * (CLKPSC+1) * EPWM_CLK.

EPWM Comparator Register 0~5 (EPWM_CMPDAT0~5)

Register	Offset	R/W	Description	Reset Value
EPWM_CMPDAT0	EPWMx_BA+0x50	R/W	EPWM Comparator Register 0	0x0000_0000
EPWM_CMPDAT1	EPWMx_BA+0x54	R/W	EPWM Comparator Register 1	0x0000_0000
EPWM_CMPDAT2	EPWMx_BA+0x58	R/W	EPWM Comparator Register 2	0x0000_0000
EPWM_CMPDAT3	EPWMx_BA+0x5C	R/W	EPWM Comparator Register 3	0x0000_0000
EPWM_CMPDAT4	EPWMx_BA+0x60	R/W	EPWM Comparator Register 4	0x0000_0000
EPWM_CMPDAT5	EPWMx_BA+0x64	R/W	EPWM Comparator Register 5	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CMP**	**EPWM Comparator Register** CMP is used to compare with CNT (EPWM_CNTn[15:0]) bits to generate EPWM waveform, interrupt and trigger EADC/DAC. In independent mode, EPWM_CMPDATn, n=0,1..5 denote as 6 independent EPWM_CH0~5 compared point. In complementary mode, EPWM_CMPDAT0, EPWM_CMPDAT2, EPWM_CMPDAT4 denote as first compared point, and EPWM_CMPDAT1, EPWM_CMPDAT3, EPWM_CMPDAT5 denote as second compared point for the corresponding 3 complementary pairs EPWM_CH0 and EPWM_CH1, EPWM_CH2 and EPWM_CH3, EPWM_CH4 and EPWM_CH5.

EPWM Dead-time Control Register 0_1, 2_3, 4_5 (EPWM_DTCTL0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
EPWM_DTCTL0_1	EPWMx_BA+0x70	R/W	EPWM Dead-time Control Register 0/1	0x0000_0000
EPWM_DTCTL2_3	EPWMx_BA+0x74	R/W	EPWM Dead-time Control Register 2/3	0x0000_0000
EPWM_DTCTL4_5	EPWMx_BA+0x78	R/W	EPWM Dead-time Control Register 4/5	0x0000_0000

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

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**DTCKSEL**	**Dead-time Clock Select (Write Protect)** 0 = Dead-time clock source from EPWM_CLK. 1 = Dead-time clock source from prescaler output. **Note:** This bit is write protected. Refer to REGWRPROT register.
[23:17]	**Reserved**	Reserved.
[16]	**DTEN**	**Enable Dead-time Insertion for EPWM Pair (EPWM_CH0, EPWM_CH1) (EPWM_CH2, EPWM_CH3) (EPWM_CH4, EPWM_CH5) (Write Protect)** Dead-time insertion is only active when this pair of complementary EPWM is enabled. If dead- time insertion is inactive, the outputs of pin pair are complementary without any delay. 0 = Dead-time insertion Disabled on the pin pair. 1 = Dead-time insertion Enabled on the pin pair. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[15:12]	**Reserved**	Reserved.
[11:0]	**DTCNT**	**Dead-time Counter (Write Protect)** The dead-time can be calculated from the following formula: DTCKSEL=0: Dead-time = (DTCNT[11:0]+1) * EPWM_CLK period. DTCKSEL=1: Dead-time = (DTCNT[11:0]+1) * EPWM_CLK period * (CLKPSC+1). **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

EPWM Counter Phase Register 0_1, 2_3, 4_5 (EPWM_PHS0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
EPWM_PHS0_1	EPWMx_BA+0x80	R/W	EPWM Counter Phase Register 0/1	0x0000_0000
EPWM_PHS2_3	EPWMx_BA+0x84	R/W	EPWM Counter Phase Register 2/3	0x0000_0000
EPWM_PHS4_5	EPWMx_BA+0x88	R/W	EPWM 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.
[15:0]	**PHS**	**EPWM Synchronous Start Phase Bits** PHS determines the EPWM synchronous start phase value. These bits only use in synchronous function.

EPWM Counter Register 0~5 (EPWM_CNT0~5)

Register	Offset	R/W	Description	Reset Value
EPWM_CNT0	EPWMx_BA+0x90	R	EPWM Counter Register 0	0x0000_0000
EPWM_CNT1	EPWMx_BA+0x94	R	EPWM Counter Register 1	0x0000_0000
EPWM_CNT2	EPWMx_BA+0x98	R	EPWM Counter Register 2	0x0000_0000
EPWM_CNT3	EPWMx_BA+0x9C	R	EPWM Counter Register 3	0x0000_0000
EPWM_CNT4	EPWMx_BA+0xA0	R	EPWM Counter Register 4	0x0000_0000
EPWM_CNT5	EPWMx_BA+0xA4	R	EPWM 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.
[16]	**DIRF**	**EPWM Direction Indicator Flag (Read Only)** 0 = Counter is counting down. 1 = Counter is counting up.
[15:0]	**CNT**	**EPWM Data Register (Read Only)** User can monitor CNT to know the current value in 16-bit period counter.

EPWM Generation Register 0 (EPWM_WGCTL0)

Register	Offset	R/W	Description	Reset Value
EPWM_WGCTL0	EPWMx_BA+0xB0	R/W	EPWM 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.
[17+2n:16+2n] n=0,1..5	**PRDPCTLn**	**EPWM Period (Center) Point Control** EPWM can control output level when EPWM counter counts to (PERIODn+1). 00 = Do nothing. 01 = EPWM period (center) point output Low. 10 = EPWM period (center) point output High. 11 = EPWM period (center) point output Toggle. **Note:** This bit is center point control when EPWM counter operating in up-down counter type.
[15:12]	**Reserved**	Reserved.
[1+2n:2n] n=0,1..5	**ZPCTLn**	**EPWM Zero Point Control** EPWM can control output level when EPWM counter counts to 0. 00 = Do nothing. 01 = EPWM zero point output Low. 10 = EPWM zero point output High. 11 = EPWM zero point output Toggle.

EPWM Generation Register 1 (EPWM_WGCTL1)

Register	Offset	R/W	Description	Reset Value
EPWM_WGCTL1	EPWMx_BA+0xB4	R/W	EPWM 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.
[17+2n:16+2n] n=0,1..5	**CMPDCTLn**	**EPWM Compare Down Point Control** EPWM can control output level when EPWM counter counts down to CMP. 00 = Do nothing. 01 = EPWM compare down point output Low. 10 = EPWM compare down point output High. 11 = EPWM compare down point output Toggle. **Note:** In complementary mode, CMPDCTL1, 3, 5 is used as another CMPDCTL for channel 0, 2, 4.
[15:12]	**Reserved**	Reserved.
[1+2n:2n] n=0,1..5	**CMPUCTLn**	**EPWM Compare Up Point Control** EPWM can control output level when EPWM counter counts up to CMP. 00 = Do nothing. 01 = EPWM compare up point output Low. 10 = EPWM compare up point output High. 11 = EPWM compare up point output Toggle. **Note:** In complementary mode, CMPUCTL1, 3, 5 is used as another CMPUCTL for channel 0, 2, 4.

EPWM Mask Enable Register (EPWM_MSKEN)

Register	Offset	R/W	Description	Reset Value
EPWM_MSKEN	EPWMx_BA+0xB8	R/W	EPWM 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.
[n] n=0,1..5	MSKENn	**EPWM Mask Enable Bits** The EPWM output signal will be masked when this bit is enabled. The corresponding EPWM channel n will output MSKDATn (EPWM_MSK[5:0]) data. 0 = EPWM output signal is non-masked. 1 = EPWM output signal is masked and output MSKDATn data.

EPWM Mask DATA Register (EPWM_MSK)

Register	Offset	R/W	Description	Reset Value
EPWM_MSK	EPWMx_BA+0xBC	R/W	EPWM 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.
[n] n=0,1..5	MSKDATn	**EPWM Mask Data Bit** This data bit control the state of EPWMn output pin, if corresponding mask function is enabled. 0 = Output logic low to EPWM channel n. 1 = Output logic high to EPWM channel n.

EPWM Brake Noise Filter Register (EPWM_BNF)

Register	Offset	R/W	Description	Reset Value
EPWM_BNF	EPWMx_BA+0xC0	R/W	EPWM Brake Noise Filter Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							BK1SRC
23	22	21	20	19	18	17	16
Reserved							BK0SRC
15	14	13	12	11	10	9	8
BRK1PINV	BRK1FCNT			BRK1NFSEL			BRK1NFEN
7	6	5	4	3	2	1	0
BRK0PINV	BRK0FCNT			BRK0NFSEL			BRK0NFEN

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**BK1SRC**	**Brake 1 Pin Source Select** For EPWM0 setting: 0 = Brake 1 pin source come from EPWM0_BRAKE1. 1 = Brake 1 pin source come from EPWM1_BRAKE1. For EPWM1 setting: 0 = Brake 1 pin source come from EPWM1_BRAKE1. 1 = Brake 1 pin source come from EPWM0_BRAKE1.
[23:17]	**Reserved**	Reserved.
[16]	**BK0SRC**	**Brake 0 Pin Source Select** For EPWM0 setting: 0 = Brake 0 pin source come from EPWM0_BRAKE0. 1 = Brake 0 pin source come from EPWM1_BRAKE0. For EPWM1 setting: 0 = Brake 0 pin source come from EPWM1_BRAKE0. 1 = Brake 0 pin source come from EPWM0_BRAKE0.
[15]	**BRK1PINV**	**Brake 1 Pin Inverse** 0 **=** brake pin event will be detected if EPWMx_BRAKE1 pin status transfer from low to high in edge-detect, or pin status is high in level-detect. 1 = brake pin event will be detected if EPWMx_BRAKE1 pin status transfer from high to low in edge-detect, or pin status is low in level-detect.
[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 = PCLK. 001 = Filter clock = PCLK/2.

		010 = Filter clock = PCLK/4. 011 = Filter clock = PCLK/8. 100 = Filter clock = PCLK/16. 101 = Filter clock = PCLK/32. 110 = Filter clock = PCLK/64. 111 = Filter clock = PCLK/128.
[8]	**BRK1NFEN**	**EPWM Brake 1 Noise Filter Enable Bit** 0 = Noise filter of EPWM Brake 1 Disabled. 1 = Noise filter of EPWM Brake 1 Enabled.
[7]	**BRK0PINV**	**Brake 0 Pin Inverse** 0 **=** brake pin event will be detected if EPWMx_BRAKE0 pin status transfer from low to high in edge-detect, or pin status is high in level-detect. 1 = brake pin event will be detected if EPWMx_BRAKE0 pin status transfer from high to low in edge-detect, or pin status is low in level-detect.
[6:4]	**BRK0FCNT**	**Brake 0 Edge Detector Filter Count** The register bits control the Brake0 filter counter to count from 0 to BRK0FCNT.
[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**	**EPWM Brake 0 Noise Filter Enable Bit** 0 = Noise filter of EPWM Brake 0 Disabled. 1 = Noise filter of EPWM Brake 0 Enabled.

EPWM System Fail Brake Control Register (EPWM_FAILBRK)

Register	Offset	R/W	Description	Reset Value
EPWM_FAILBRK	EPWMx_BA+0xC4	R/W	EPWM 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.
[3]	**CORBRKEN**	**Core Lockup Detection Trigger EPWM 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 EPWM 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 EPWM 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 EPWM Brake Function 0 Enable Bit** 0 = Brake Function triggered by CSS detection Disabled. 1 = Brake Function triggered by CSS detection Enabled.

EPWM Brake Edge Detect Control Register 0_1, 2_3, 4_5 (EPWM_BRKCTL0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
EPWM_BRKCTL0_1	EPWMx_BA+0xC8	R/W	EPWM Brake Edge Detect Control Register 0/1	0x0000_0000
EPWM_BRKCTL2_3	EPWMx_BA+0xCC	R/W	EPWM Brake Edge Detect Control Register 2/3	0x0000_0000
EPWM_BRKCTL4_5	EPWMx_BA+0xD0	R/W	EPWM Brake Edge Detect Control Register 4/5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			EADC0LBEN	Reserved			
23	22	21	20	19	18	17	16
Reserved			EADC0EBEN	BRKAODD		BRKAEVEN	
15	14	13	12	11	10	9	8
SYSLBEN	Reserved	BRKP1LEN	BRKP0LEN	Reserved		CPO1LBEN	CPO0LBEN
7	6	5	4	3	2	1	0
SYSEBEN	Reserved	BRKP1EEN	BRKP0EEN	Reserved		CPO1EBEN	CPO0EBEN

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28]	**EADC0LBEN**	**Enable EADC0 Result Monitor (EADC0RM) As Level-detect Brake Source (Write Protect)** 0 = EADC0RM as level-detect brake source Disabled. 1 = EADC0RM as level-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[27:21]	**Reserved**	Reserved.
[20]	**EADC0EBEN**	**Enable EADC0 Result Monitor (EADC0RM) As Edge-detect Brake Source (Write Protect)** 0 = EADC0RM as edge-detect brake source Disabled. 1 = EADC0RM as edge-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[19:18]	**BRKAODD**	**EPWM Brake Action Select for Odd Channel (Write Protect)** 00 = EPWMx brake event will not affect odd channels output. 01 = EPWM odd channel output tri-state when EPWMx brake event happened. 10 = EPWM odd channel output low level when EPWMx brake event happened. 11 = EPWM odd channel output high level when EPWMx brake event happened. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[17:16]	**BRKAEVEN**	**EPWM Brake Action Select for Even Channel (Write Protect)** 00 = EPWMx brake event will not affect even channels output. 01 = EPWM even channel output tri-state when EPWMx brake event happened. 10 = EPWM even channel output low level when EPWMx brake event happened. 11 = EPWM even channel output high level when EPWMx brake event happened. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[15]	**SYSLBEN**	**Enable System Fail As Level-detect Brake Source (Write Protect)**

		0 = System Fail condition as level-detect brake source Disabled. 1 = System Fail condition as level-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[14]	**Reserved**	Reserved.
[13]	**BRKP1LEN**	**Enable BKP1 Pin As Level-detect Brake Source (Write Protect)** 0 = EPWMx_BRAKE1 pin as level-detect brake source Disabled. 1 = EPWMx_BRAKE1 pin as level-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[12]	**BRKP0LEN**	**Enable BKP0 Pin As Level-detect Brake Source (Write Protect)** 0 = EPWMx_BRAKE0 pin as level-detect brake source Disabled. 1 = EPWMx_BRAKE0 pin as level-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[11:10]	**Reserved**	Reserved.
[9]	**CPO1LBEN**	**Enable ACMP1_O Digital Output As Level-detect Brake Source (Write Protect)** 0 = ACMP1_O as level-detect brake source Disabled. 1 = ACMP1_O as level-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[8]	**CPO0LBEN**	**Enable ACMP0_O Digital Output As Level-detect Brake Source (Write Protect)** 0 = ACMP0_O as level-detect brake source Disabled. 1 = ACMP0_O as level-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[7]	**SYSEBEN**	**Enable System Fail As Edge-detect Brake Source (Write Protect)** 0 = System Fail condition as edge-detect brake source Disabled. 1 = System Fail condition as edge-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[6]	**Reserved**	Reserved.
[5]	**BRKP1EEN**	**Enable EPWMx_BRAKE1 Pin As Edge-detect Brake Source (Write Protect)** 0 = EPWMx_BRAKE1 pin as edge-detect brake source Disabled. 1 = EPWMx_BRAKE1 pin as edge-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[4]	**BRKP0EEN**	**Enable EPWMx_BRAKE0 Pin As Edge-detect Brake Source (Write Protect)** 0 = EPWMx_BRAKE0 pin as edge-detect brake source Disabled. 1 = EPWMx_BRAKE0 pin as edge-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[3:2]	**Reserved**	Reserved.
[1]	**CPO1EBEN**	**Enable ACMP1_O Digital Output As Edge-detect Brake Source (Write Protect)** 0 = ACMP1_O as edge-detect brake source Disabled. 1 = ACMP1_O as edge-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[0]	**CPO0EBEN**	**Enable ACMP0_O Digital Output As Edge-detect Brake Source (Write Protect)** 0 = ACMP0_O as edge-detect brake source Disabled. 1 = ACMP0_O as edge-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

EPWM Pin Polar Inverse Control (EPWM_POLCTL)

Register	Offset	R/W	Description	Reset Value
EPWM_POLCTL	EPWMx_BA+0xD4	R/W	EPWM 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.
[n] n=0,1..5	**PINVn**	**EPWM PIN Polar Inverse Control** The register controls polarity state of EPWMx_CHn output pin. 0 = EPWMx_CHn output pin polar inverse Disabled. 1 = EPWMx_CHn output pin polar inverse Enabled.

EPWM Output Enable Register (EPWM_POEN)

Register	Offset	R/W	Description	Reset Value
EPWM_POEN	EPWMx_BA+0xD8	R/W	EPWM 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.
[n] n=0,1..5	**POENn**	**EPWM Pin Output Enable Bits** 0 = EPWMx_CHn pin at tri-state. 1 = EPWMx_CHn pin in output mode.

EPWM Software Brake Control Register (EPWM_SWBRK)

Register	Offset	R/W	Description	Reset Value
EPWM_SWBRK	EPWMx_BA+0xDC	W	EPWM 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.
[8+n/2] n=0,2,4	**BRKLTRGn**	**EPWM Level Brake Software Trigger (Write Only) (Write Protect)** Write 1 to this bit will trigger level brake, and set BRKLIFn to 1 in EPWM_INTSTS1 register. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[7:3]	**Reserved**	Reserved.
[n/2] n=0,2,4	**BRKETRGn**	**EPWM Edge Brake Software Trigger (Write Only) (Write Protect)** Write 1 to this bit will trigger edge brake, and set BRKEIFn to 1 in EPWM_INTSTS1 register. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

EPWM Interrupt Enable Register 0 (EPWM_INTEN0)

Register	Offset	R/W	Description	Reset Value
EPWM_INTEN0	EPWMx_BA+0xE0	R/W	EPWM 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.
[24+n] n=0,1..5	**CMPDIENn**	**EPWM 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 is used as another CMPDIEN for channel 0, 2, 4.
[23:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**CMPUIENn**	**EPWM 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 is used as another CMPUIEN for channel 0, 2, 4.
[15:14]	**Reserved**	Reserved.
[8+n] n=0,1..5	**PIENn**	**EPWM Period Point Interrupt Enable Bits** 0 = Period point interrupt Disabled. 1 = Period point interrupt Enabled. **Note 1:** When up-down counter type period point means center point. **Note 2:** Odd channels will read always 0 at complementary mode.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**ZIENn**	**EPWM Zero Point Interrupt Enable Bits** 0 = Zero point interrupt Disabled. 1 = Zero point interrupt Enabled. **Note:** Odd channels will read always 0 at complementary mode.

EPWM Interrupt Enable Register 1 (EPWM_INTEN1)

Register	Offset	R/W	Description	Reset Value
EPWM_INTEN1	EPWMx_BA+0xE4	R/W	EPWM Interrupt Enable Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					BRKLIEN4_5	BRKLIEN2_3	BRKLIEN0_1
7	6	5	4	3	2	1	0
Reserved					BRKEIEN4_5	BRKEIEN2_3	BRKEIEN0_1

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10]	**BRKLIEN4_5**	**EPWM Level-detect Brake Interrupt Enable for Channel4/5 (Write Protect)** 0 = Level-detect Brake interrupt for channel4/5 Disabled. 1 = Level-detect Brake interrupt for channel4/5 Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[9]	**BRKLIEN2_3**	**EPWM Level-detect Brake Interrupt Enable for Channel2/3 (Write Protect)** 0 = Level-detect Brake interrupt for channel2/3 Disabled. 1 = Level-detect Brake interrupt for channel2/3 Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[8]	**BRKLIEN0_1**	**EPWM Level-detect Brake Interrupt Enable for Channel0/1 (Write Protect)** 0 = Level-detect Brake interrupt for channel0/1 Disabled. 1 = Level-detect Brake interrupt for channel0/1 Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[7:3]	**Reserved**	Reserved.
[2]	**BRKEIEN4_5**	**EPWM Edge-detect Brake Interrupt Enable for Channel4/5 (Write Protect)** 0 = Edge-detect Brake interrupt for channel4/5 Disabled. 1 = Edge-detect Brake interrupt for channel4/5 Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[1]	**BRKEIEN2_3**	**EPWM Edge-detect Brake Interrupt Enable for Channel2/3 (Write Protect)** 0 = Edge-detect Brake interrupt for channel2/3 Disabled. 1 = Edge-detect Brake interrupt for channel2/3 Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[0]	**BRKEIEN0_1**	**EPWM Edge-detect Brake Interrupt Enable for Channel0/1 (Write Protect)** 0 = Edge-detect Brake interrupt for channel0/1 Disabled. 1 = Edge-detect Brake interrupt for channel0/1 Enabled.

		Note: This bit is write protected. Refer to SYS_REGLCTL register.

EPWM Interrupt Flag Register 0 (EPWM_INTSTS0)

Register	Offset	R/W	Description	Reset Value
EPWM_INTSTS0	EPWMx_BA+0xE8	R/W	EPWM 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.
[24+n] n=0,1..5	**CMPDIFn**	**EPWM Compare Down Count Interrupt Flag** Flag is set by hardware when EPWM counter down count and reaches EPWM_CMPDATn, software can clear this bit by writing 1 to it. **Note:** In complementary mode, CMPDIF1, 3, 5 is used as another CMPDIF for channel 0, 2, 4. **Note:** In complementary mode, CMPDIF1, 3, 5 will be set when the corresponding counter enable bits are set at the same time with even channels
[23:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**CMPUIFn**	**EPWM Compare Up Count Interrupt Flag** Flag is set by hardware when EPWM counter up count and reaches EPWM_CMPDATn, software can clear this bit by writing 1 to it. **Note:** In complementary mode, CMPUIF1, 3, 5 is used as another CMPUIF for channel 0, 2, 4. **Note:** In complementary mode, CMPUIF1, 3, 5 will be set when the corresponding counter enable bits are set at the same time with even channels
[15:14]	**Reserved**	Reserved.
[8+n] n=0,1..5	**PIFn**	**EPWM Period Point Interrupt Flag** This bit is set by hardware when EPWM counter reaches EPWM_PERIODn. **Note:** This bit can be cleared to 0 by software writing 1.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**ZIFn**	**EPWM Zero Point Interrupt Flag** This bit is set by hardware when EPWM counter reaches 0. **Note:** This bit can be cleared to 0 by software writing 1

EPWM Interrupt Flag Register 1 (EPWM_INTSTS1)

Register	Offset	R/W	Description	Reset Value
EPWM_INTSTS1	EPWMx_BA+0xEC	R/W	EPWM 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.
[24+n] n=0,1..5	**BRKLSTSn**	**EPWM Channel N Level-detect Brake Status (Read Only)** 0 = EPWM channel n level-detect brake state is released. 1 = When EPWM channel n level-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the EPWM channel n at brake state. **Note:** This bit is read only and auto cleared by hardware. When enabled brake source return to high level, EPWM will release brake state until current EPWM period finished. The EPWM waveform will start output from next full EPWM period.
[23:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**BRKESTSn**	**EPWM Channel N Edge-detect Brake Status (Read Only)** 0 = EPWM channel n edge-detect brake state is released. 1 = When EPWM channel n edge-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the EPWM channel n at brake state. **Note:** This bit is read only and auto cleared by hardware. When edge-detect brake interrupt flag is cleared, EPWM will release brake state until current EPWM period finished. The EPWM waveform will start output from next full EPWM period.
[15:14]	**Reserved**	Reserved.
[8+n] n=0,1..5	**BRKLIFn**	**EPWM Channel N Level-detect Brake Interrupt Flag (Write Protect)** 0 = EPWM channel n level-detect brake event do not happened. 1 = When EPWM channel n level-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**BRKEIFn**	**EPWM Channel N Edge-detect Brake Interrupt Flag (Write Protect)** 0 = EPWM channel n edge-detect brake event do not happened. 1 = When EPWM channel n edge-detect brake event happened, this bit is set to 1, writing 1 to clear. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

EPWM Trigger DAC Enable Register (EPWM_DACTRGEN)

Register	Offset	R/W	Description	Reset Value
EPWM_DACTRGEN	EPWMx_BA+0xF4	R/W	EPWM Trigger DAC Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		CDTRGEN5	CDTRGEN4	CDTRGEN3	CDTRGEN2	CDTRGEN1	CDTRGEN0
23	22	21	20	19	18	17	16
Reserved		CUTRGEN5	CUTRGEN4	CUTRGEN3	CUTRGEN2	CUTRGEN1	CUTRGEN0
15	14	13	12	11	10	9	8
Reserved		PTE5	PTE4	PTE3	PTE2	PTE1	PTE0
7	6	5	4	3	2	1	0
Reserved		ZTE5	ZTE4	ZTE3	ZTE2	ZTE1	ZTE0

Bits	Description	
[31:30]	**Reserved**	Reserved.
[24+n] n=0,1..5	**CDTRGEn**	**EPWM Compare Down Count Point Trigger DAC Enable Bits** EPWM can trigger DAC to start action when EPWM counter down count to CMP if this bit is set to1. 0 = EPWM Compare Down count point trigger DAC function Disabled. 1 = EPWM Compare Down count point trigger DAC function Enabled. **Note 1:** This bit should keep at 0 when EPWM counter operating in up counter type. **Note 2:** In complementary mode, CDTRGEN1, 3, 5 is used as another CDTRGEN for channel 0, 2, 4.
[23:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**CUTRGEn**	**EPWM Compare Up Count Point Trigger DAC Enable Bits** EPWM can trigger DAC to start action when EPWM counter counts up to CMP if this bit is set to1. 0 = EPWM Compare Up point trigger DAC function Disabled. 1 = EPWM Compare Up point trigger DAC function Enabled. **Note 1:** This bit should keep at 0 when EPWM counter operating in down counter type. **Note 2:** In complementary mode, CUTRGEN1, 3, 5 is used as another CUTRGEN for channel 0, 2, 4.
[15:14]	**Reserved**	Reserved.
[8+n] n=0,1..5	**PTEn**	**EPWM Period Point Trigger DAC Enable Bits** EPWM can trigger DAC to start action when EPWM counter counts up to (PERIODn+1) if this bit is set to1. 0 = EPWM period point trigger DAC function Disabled. 1 = EPWM period point trigger DAC function Enabled.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**ZTEn**	**EPWM Zero Point Trigger DAC Enable Bits** EPWM can trigger EADC/DAC/DMA to start action when EPWM counter down count to zero if this bit is set to1. 0 = EPWM period point trigger DAC function Disabled. 1 = EPWM period point trigger DAC function Enabled.

EPWM Trigger EADC Source Select Register 0 (EPWM_EADCTS0)

Register	Offset	R/W	Description	Reset Value
EPWM_EADCTS0	EPWMx_BA+0xF8	R/W	EPWM 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**	**EPWM_CH3 Trigger EADC Enable Bit** 0 = EPWM_CH3 Trigger EADC function Disabled. 1 = EPWM_CH3 Trigger EADC function Enabled.
[30:28]	**Reserved**	Reserved.
[27:24]	**TRGSEL3**	**EPWM_CH3 Trigger EADC Source Select** 0000 = EPWM_CH2 zero point. 0001 = EPWM_CH2 period point. 0010 = EPWM_CH2 zero or period point. 0011 = EPWM_CH2 up-count compared point. 0100 = EPWM_CH2 down-count compared point. 0101 = EPWM_CH3 zero point. 0110 = EPWM_CH3 period point. 0111 = EPWM_CH3 zero or period point. 1000 = EPWM_CH3 up-count compared point. 1001 = EPWM_CH3 down-count compared point. 1010 = EPWM_CH0 up-count free trigger compared point. 1011 = EPWM_CH0 down-count free trigger compared point. 1100 = EPWM_CH2 up-count free trigger compared point. 1101 = EPWM_CH2 down-count free trigger compared point. 1110 = EPWM_CH4 up-count free trigger compared point. 1111 = EPWM_CH4 down-count free trigger compared point.
[23]	**TRGEN2**	**EPWM_CH2 Trigger EADC Enable Bit** 0 = EPWM_CH2 Trigger EADC function Disabled. 1 = EPWM_CH2 Trigger EADC function Enabled.
[22:20]	**Reserved**	Reserved.
[19:16]	**TRGSEL2**	**EPWM_CH2 Trigger EADC Source Select**

		0000 = EPWM_CH2 zero point. 0001 = EPWM_CH2 period point. 0010 = EPWM_CH2 zero or period point. 0011 = EPWM_CH2 up-count compared point. 0100 = EPWM_CH2 down-count compared point. 0101 = EPWM_CH3 zero point. 0110 = EPWM_CH3 period point. 0111 = EPWM_CH3 zero or period point. 1000 = EPWM_CH3 up-count compared point. 1001 = EPWM_CH3 down-count compared point. 1010 = EPWM_CH0 up-count free trigger compared point. 1011 = EPWM_CH0 down-count free trigger compared point. 1100 = EPWM_CH2 up-count free trigger compared point. 1101 = EPWM_CH2 down-count free trigger compared point. 1110 = EPWM_CH4 up-count free trigger compared point. 1111 = EPWM_CH4 down-count free trigger compared point.
[15]	**TRGEN1**	**EPWM_CH1 Trigger EADC Enable Bit** 0 = EPWM_CH1 Trigger EADC function Disabled. 1 = EPWM_CH1 Trigger EADC function Enabled.
[14:12]	**Reserved**	Reserved.
[11:8]	**TRGSEL1**	**EPWM_CH1 Trigger EADC Source Select** 0000 = EPWM_CH0 zero point. 0001 = EPWM_CH0 period point. 0010 = EPWM_CH0 zero or period point. 0011 = EPWM_CH0 up-count compared point. 0100 = EPWM_CH0 down-count compared point. 0101 = EPWM_CH1 zero point. 0110 = EPWM_CH1 period point. 0111 = EPWM_CH1 zero or period point. 1000 = EPWM_CH1 up-count compared point. 1001 = EPWM_CH1 down-count compared point. 1010 = EPWM_CH0 up-count free trigger compared point. 1011 = EPWM_CH0 down-count free trigger compared point. 1100 = EPWM_CH2 up-count free trigger compared point. 1101 = EPWM_CH2 down-count free trigger compared point. 1110 = EPWM_CH4 up-count free trigger compared point. 1111 = EPWM_CH4 down-count free trigger compared point.
[7]	**TRGEN0**	**EPWM_CH0 Trigger EADC Enable Bit** 0 = EPWM_CH0 Trigger EADC function Disabled. 1 = EPWM_CH0 Trigger EADC function Enabled.
[6:4]	**Reserved**	Reserved.
[3:0]	**TRGSEL0**	**EPWM_CH0 Trigger EADC Source Select** 0000 = EPWM_CH0 zero point. 0001 = EPWM_CH0 period point. 0010 = EPWM_CH0 zero or period point. 0011 = EPWM_CH0 up-count compared point. 0100 = EPWM_CH0 down-count compared point.

		0101 = EPWM_CH1 zero point. 0110 = EPWM_CH1 period point. 0111 = EPWM_CH1 zero or period point. 1000 = EPWM_CH1 up-count compared point. 1001 = EPWM_CH1 down-count compared point. 1010 = EPWM_CH0 up-count free trigger compared point. 1011 = EPWM_CH0 down-count free trigger compared point. 1100 = EPWM_CH2 up-count free trigger compared point. 1101 = EPWM_CH2 down-count free trigger compared point. 1110 = EPWM_CH4 up-count free trigger compared point. 1111 = EPWM_CH4 down-count free trigger compared point.

EPWM Trigger EADC Source Select Register 1 (EPWM_EADCTS1)

Register	Offset	R/W	Description	Reset Value
EPWM_EADCTS1	EPWMx_BA+0xFC	R/W	EPWM 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.
[15]	**TRGEN5**	**EPWM_CH5 Trigger EADC Enable Bit** 0 = EPWM_CH5 Trigger EADC function Disabled. 1 = EPWM_CH5 Trigger EADC function Enabled.
[14:12]	**Reserved**	Reserved.
[11:8]	**TRGSEL5**	**EPWM_CH5 Trigger EADC Source Select** 0000 = EPWM_CH4 zero point. 0001 = EPWM_CH4 period point. 0010 = EPWM_CH4 zero or period point. 0011 = EPWM_CH4 up-count compared point. 0100 = EPWM_CH4 down-count compared point. 0101 = EPWM_CH5 zero point. 0110 = EPWM_CH5 period point. 0111 = EPWM_CH5 zero or period point. 1000 = EPWM_CH5 up-count compared point. 1001 = EPWM_CH5 down-count compared point. 1010 = EPWM_CH0 up-count free trigger compared point. 1011 = EPWM_CH0 down-count free trigger compared point. 1100 = EPWM_CH2 up-count free trigger compared point. 1101 = EPWM_CH2 down-count free trigger compared point. 1110 = EPWM_CH4 up-count free trigger compared point. 1111 = EPWM_CH4 down-count free trigger compared point.
[7]	**TRGEN4**	**EPWM_CH4 Trigger EADC Enable Bit** 0 = EPWM_CH4 Trigger EADC function Disabled. 1 = EPWM_CH4 Trigger EADC function Enabled.
[6:4]	**Reserved**	Reserved.

[3:0]	**TRGSEL4**	**EPWM_CH4 Trigger EADC Source Select** 0000 = EPWM_CH4 zero point. 0001 = EPWM_CH4 period point. 0010 = EPWM_CH4 zero or period point. 0011 = EPWM_CH4 up-count compared point. 0100 = EPWM_CH4 down-count compared point. 0101 = EPWM_CH5 zero point. 0110 = EPWM_CH5 period point. 0111 = EPWM_CH5 zero or period point. 1000 = EPWM_CH5 up-count compared point. 1001 = EPWM_CH5 down-count compared point. 1010 = EPWM_CH0 up-count free trigger compared point. 1011 = EPWM_CH0 down-count free trigger compared point. 1100 = EPWM_CH2 up-count free trigger compared point. 1101 = EPWM_CH2 down-count free trigger compared point. 1110 = EPWM_CH4 up-count free trigger compared point. 1111 = EPWM_CH4 down-count free trigger compared point.

EPWM Free Trigger Compare Register 0_1, 2_3, 4_5 (EPWM_FTCMPDAT0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
EPWM_FTCMPDAT0_1	EPWMx_BA+0x100	R/W	EPWM Free Trigger Compare Register 0/1	0x0000_0000
EPWM_FTCMPDAT2_3	EPWMx_BA+0x104	R/W	EPWM Free Trigger Compare Register 2/3	0x0000_0000
EPWM_FTCMPDAT4_5	EPWMx_BA+0x108	R/W	EPWM 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.
[15:0]	**FTCMP**	**EPWM Free Trigger Compare Register** FTCMP use to compare with even CNT (EPWM_CNTm[15:0], m=0,2,4) to trigger EADC. EPWM_FTCMPDAT0_1, EPWM_FTCMPDAT2_3, EPWM_FTCMPDAT4_5 corresponding complementary pairs EPWM_CH0and EPWM_CH1, EPWM_CH2 and EPWM_CH3, EPWM_CH4 and EPWM_CH5.

EPWM Synchronous Start Control Register (EPWM_SSCTL)

Register	Offset	R/W	Description	Reset Value
EPWM_SSCTL	EPWMx_BA+0x110	R/W	EPWM 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.
[9:8]	**SSRC**	**EPWM Synchronous Start Source Select Bits** 00 = Synchronous start source come from EPWM0. 01 = Synchronous start source come from EPWM1. 10 = Synchronous start source come from BPWM0. 11 = Synchronous start source come from BPWM1.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**SSENn**	**EPWM Synchronous Start Function Enable Bits** When synchronous start function is enabled, the EPWM counter enable register (EPWM_CNTEN) can be enabled by writing EPWM synchronous start trigger bit (CNTSEN). 0 = EPWM synchronous start function Disabled. 1 = EPWM synchronous start function Enabled.

EPWM Synchronous Start Trigger Register (EPWM_SSTRG)

Register	Offset	R/W	Description	Reset Value
EPWM_SSTRG	EPWMx_BA+0x114	W	EPWM Synchronous Start Trigger Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**CNTSEN**	**EPWM Counter Synchronous Start Enable (Write Only)** PMW counter synchronous enable function is used to make selected EPWM channels (include EPWM0_CHx and EPWM1_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 EPWM channel counter synchronous start function is enabled.

EPWM Leading Edge Blanking Control Register (EPWM_LEBCTL)

Register	Offset	R/W	Description	Reset Value
EPWM_LEBCTL	EPWMx_BA+0x118	R/W	EPWM Leading Edge Blanking Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						TRGTYPE	
15	14	13	12	11	10	9	8
Reserved					SRCEN4	SRCEN2	SRCEN0
7	6	5	4	3	2	1	0
Reserved							LEBEN

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17:16]	**TRGTYPE**	**EPWM Leading Edge Blanking Trigger Type** 0 = When detect leading edge blanking source rising edge, blanking counter start counting. 1 = When detect leading edge blanking source falling edge, blanking counter start counting. 2 = When detect leading edge blanking source rising or falling edge, blanking counter start counting. 3 = Reserved.
[15:11]	**Reserved**	Reserved.
[10]	**SRCEN4**	**EPWM Leading Edge Blanking Source From EPWM_CH4 Enable Bit** 0 = EPWM Leading Edge Blanking Source from EPWM_CH4 Disabled. 1 = EPWM Leading Edge Blanking Source from EPWM_CH4 Enabled.
[9]	**SRCEN2**	**EPWM Leading Edge Blanking Source From EPWM_CH2 Enable Bit** 0 = EPWM Leading Edge Blanking Source from EPWM_CH2 Disabled. 1 = EPWM Leading Edge Blanking Source from EPWM_CH2 Enabled.
[8]	**SRCEN0**	**EPWM Leading Edge Blanking Source From EPWM_CH0 Enable Bit** 0 = EPWM Leading Edge Blanking Source from EPWM_CH0 Disabled. 1 = EPWM Leading Edge Blanking Source from EPWM_CH0 Enabled.
[7:1]	**Reserved**	Reserved.
[0]	**LEBEN**	**EPWM Leading Edge Blanking Enable Bit** 0 = EPWM Leading Edge Blanking Disabled. 1 = EPWM Leading Edge Blanking Enabled.

EPWM Leading Edge Blanking Counter Register (EPWM_LEBCNT)

Register	Offset	R/W	Description	Reset Value
EPWM_LEBCNT	EPWMx_BA+0x11C	R/W	EPWM Leading Edge Blanking 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							**LEBCNT**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LEBCNT							

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8:0]	**LEBCNT**	**EPWM Leading Edge Blanking Counter** This counter value decides leading edge blanking window size. Blanking window size = LEBCNT+1, and LEB counter clock base is ECLK.

EPWM Status Register (EPWM_STATUS)

Register	Offset	R/W	Description	Reset Value
EPWM_STATUS	EPWMx_BA+0x120	R/W	EPWM Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							DACTRGF
23	22	21	20	19	18	17	16
Reserved		EADCTRGF5	EADCTRGF4	EADCTRGF3	EADCTRGF2	EADCTRGF1	EADCTRGF0
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	
[31:25]	**Reserved**	Reserved.
[24]	**DACTRGF**	**DAC Start of Conversion Flag** 0 = No DAC start of conversion trigger event has occurred. 1 = A DAC start of conversion trigger event has occurred. **Note:** This bit can be cleared by software writing 1.
[23:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**EADCTRGFn**	**EADC Start of Conversion Flag** 0 = No EADC start of conversion trigger event has occurred. 1 = An EADC start of conversion trigger event has occurred. **Note:** This bit can be cleared by software writing 1.
[15:11]	**Reserved**	Reserved.
[8+n/2] n=0,2,4	**SYNCINFn**	**Input Synchronization Latched Flag** 0 = No SYNC_IN event has occurred. 1 = A SYNC_IN event has occurred. **Note:** This bit can be cleared by software writing 1.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**CNTMAXFn**	**Time-base Counter Equal to 0xFFFF Latched Flag** 0 = The time-base counter never reached its maximum value 0xFFFF. 1 = The time-base counter reached its maximum value. **Note:** This bit can be cleared by software writing 1.

EPWM Interrupt Flag Accumulator Register (EPWM_IFAn)

Register	Offset	R/W	Description	Reset Value
EPWM_IFA0	EPWMx_BA+0x130	R/W	EPWM Interrupt Flag Accumulator Register 0	0x0000_0000
EPWM_IFA1	EPWMx_BA+0x134	R/W	EPWM Interrupt Flag Accumulator Register 1	0x0000_0000
EPWM_IFA2	EPWMx_BA+0x138	R/W	EPWM Interrupt Flag Accumulator Register 2	0x0000_0000
EPWM_IFA3	EPWMx_BA+0x13C	R/W	EPWM Interrupt Flag Accumulator Register 3	0x0000_0000
EPWM_IFA4	EPWMx_BA+0x140	R/W	EPWM Interrupt Flag Accumulator Register 4	0x0000_0000
EPWM_IFA5	EPWMx_BA+0x144	R/W	EPWM Interrupt Flag Accumulator Register 5	0x0000_0000

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

Bits	Description	
[31]	**IFAEN**	**EPWM_CHn Interrupt Flag Accumulator Enable Bit** 0 = EPWM_CHn interrupt flag accumulator Disabled. 1 = EPWM_CHn interrupt flag accumulator Enabled. **Note:** Disabling this bit will reset related EPWM_IFACNT
[30]	**Reserved**	Reserved.
[29:28]	**IFASEL**	**EPWM_CHn Interrupt Flag Accumulator Source Select** 00 = EPWM_CHn zero point. 01 = EPWM_CHn period in channel n. 10 = EPWM_CHn up-count compared point. 11 = EPWM_CHn down-count compared point.
[27:25]	**Reserved**	Reserved.
[24]	**STPMOD**	**EPWM_CHn Accumulator Stop Mode Enable Bit** 0 = EPWM_CHn Stop Mode Disabled. 1 = EPWM_CHn Stop Mode Enabled.
[23:16]	**Reserved**	Reserved.
[15:0]	**IFACNT**	**EPWM_CHn Interrupt Flag Counter** The register sets the count number which defines (IFACNT+1) times of EPWM_CHn period occurs to set bit IFAIFn to request the EPWM period interrupt.EPWM flag will be set in every IFACNT[15:0] times of EPWM

		period.

EPWM Accumulator Interrupt Flag Register (EPWM_AINTSTS)

Register	Offset	R/W	Description	Reset Value
EPWM_AINTSTS	EPWMx_BA+0x150	R/W	EPWM Accumulator 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							
7	6	5	4	3	2	1	0
Reserved		IFAIF5	IFAIF4	IFAIF3	IFAIF2	IFAIF1	IFAIF0

Bits	Description	
[31:6]	Reserved	Reserved.
[n] n=0,1..5	IFAIFn	**EPWM_CHn Interrupt Flag Accumulator Interrupt Flag** Flag is set by hardware when condition match IFASEL in EPWM_IFAn register, software can clear this bit by writing 1 to it.

EPWM Accumulator Interrupt Enable Register (EPWM_AINTEN)

Register	Offset	R/W	Description	Reset Value
EPWM_AINTEN	EPWMx_BA+0x154	R/W	EPWM Accumulator 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		**IFAIEN5**	**IFAIEN4**	**IFAIEN3**	**IFAIEN2**	**IFAIEN1**	**IFAIEN0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**IFAIENn**	**EPWM_CHn Interrupt Flag Accumulator Interrupt Enable Bits** 0 = Interrupt Flag accumulator interrupt Disabled. 1 = Interrupt Flag accumulator interrupt Enabled.

EPWM Accumulator PDMA Control Register (EPWM_APDMACTL)

Register	Offset	R/W	Description	Reset Value
EPWM_APDMACTL	EPWMx_BA+0x158	R/W	EPWM Accumulator 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		APDMAEN5	APDMAEN4	APDMAEN3	APDMAEN2	APDMAEN1	APDMAEN0

Bits	Description	
[31:6]	Reserved	Reserved.
[n] n=0,1..5	APDMAENn	**Channel n Accumulator PDMA Enable Bits** 0 = Channel n PDMA function Disabled. 1 = Channel n PDMA function Enabled for the channel n to trigger PDMA to transfer memory data to register.

EPWM Fault Detect Enable Register (EPWM_FDEN)

Register	Offset	R/W	Description	Reset Value
EPWM_FDEN	EPWMx_BA+0x160	R/W	EPWM Fault Detect Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved		FDCKS5	FDCKS4	FDCKS3	FDCKS2	FDCKS1	FDCKS0
15	14	13	12	11	10	9	8
Reserved		FDODIS5	FDODIS4	FDODIS3	FDODIS2	FDODIS1	FDODIS0
7	6	5	4	3	2	1	0
Reserved		FDEN5	FDEN4	FDEN3	FDEN2	FDEN1	FDEN0

Bits	Description	
[31:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**FDCKSn**	**EPWM Channel n Fault Detect Clock Source Select Bits** 0 = EPWMx_CLK, x denotes 0 or 1. 1 = EPWMx_CLK divide by prescaler, x denotes 0 or 1.
[8+n] n=0,1..5	**FDODISn**	**EPWM Channel n Output Fault Detect Disable Bits** 0 = EPWM detect fault and output Enabled. 1 = EPWM detect fault and output Disabled.
[n] n=0,1..5	**FDENn**	**EPWM Fault Detect Function Enable Bits** 0 = Fault detect function Disabled. 1 = Fault detect function Enabled.

EPWM Fault Detect Control Register 0~5 (EPWM_FDCTL0~5)

Register	Offset	R/W	Description	Reset Value
EPWM_FDCTL0	EPWMx_BA+0x164	R/W	EPWM Fault Detect Control Register 0	0x0000_0000
EPWM_FDCTL1	EPWMx_BA+0x168	R/W	EPWM Fault Detect Control Register 1	0x0000_0000
EPWM_FDCTL2	EPWMx_BA+0x16C	R/W	EPWM Fault Detect Control Register 2	0x0000_0000
EPWM_FDCTL3	EPWMx_BA+0x170	R/W	EPWM Fault Detect Control Register 3	0x0000_0000
EPWM_FDCTL4	EPWMx_BA+0x174	R/W	EPWM Fault Detect Control Register 4	0x0000_0000
EPWM_FDCTL5	EPWMx_BA+0x178	R/W	EPWM Fault Detect Control Register 5	0x0000_0000

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

Bits	Description	
[31]	**FDDGEN**	**Fault Detect Deglitch Enable Bit** 0 = Fault detect deglitch function Disabled. 1 = Fault detect deglitch function Enabled.
[30]	**Reserved**	Reserved.
[29:28]	**FDCKSEL**	**EPWM Channel Fault Detect Clock Select** 00 = FLT_CLK/1. 01 = FLT_CLK/2. 10 = FLT_CLK/4. 11 = FLT_CLK/8. **Note:** FLT_CLK is FDCKSn (EPWM_FDENn[16+n], n=0,1..5) selected clock.
[27:19]	**Reserved**	Reserved.
[18:16]	**DGSMPCYC**	**Deglitch Sampling Cycle** FDCKS is set to 0: Sampling detect signal each EPWMx_CLK * (2^FDCKSEL) period and detect DGSMPCYC+1 times FDCKS is set to 1: Sampling detect signal each EPWMx_CLK * CLKPSC * (2^FDCKSEL) period and detect DGSMPCYC+1 times **Note:**

		CLKPSC (EPWM_CLKPSCn[11:0]) is 0: TRMSKCNT >= DGSMPCYC + 2 FDCKS is 1 and CLKPSC (EPWM_CLKPSCn[11:0]) is 1: TRMSKCNT >= DGSMPCYC + 1 FDCKS is 1 and CLKPSC (EPWM_CLKPSCn[11:0]) is 2: TRMSKCNT >= DGSMPCYC
[15]	**FDMSKEN**	**Fault Detect Mask Enable Bit** 0 = Fault detect mask function Disabled. 1 = Fault detect mask function Enabled.
[14:7]	**Reserved**	Reserved.
[6:0]	**TRMSKCNT**	**Transition Mask Counter** The fault detect result will be masked before counter count from 0 to TRMSKCNT. FDCKS is set to 0: Mask time is EPWMx_CLK * (2^FDCKSEL) * (TRMSKCNT+2) FDCKS is set to 1: Mask time EPWMx_CLK * CLKPSC * (2^FDCKSEL) * (TRMSKCNT+2) **Note:** CLKPSC (EPWM_CLKPSCn[11:0]) is 0: TRMSKCNT >= DGSMPCYC + 2 FDCKS is 1 and CLKPSC (EPWM_CLKPSCn[11:0]) is 1: TRMSKCNT >= DGSMPCYC + 1 FDCKS is 1 and CLKPSC (EPWM_CLKPSCn[11:0]) is 2: TRMSKCNT >= DGSMPCYC

EPWM Fault Detect Interrupt Enable Register (EPWM_FDIEN)

Register	Offset	R/W	Description	Reset Value
EPWM_FDIEN	EPWMx_BA+0x17C	R/W	EPWM Fault Detect 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		**FDIEN5**	**FDIEN4**	**FDIEN3**	**FDIEN2**	**FDIEN1**	**FDIEN0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n]	**FDIENn**	**EPWM Channel n Fault Detect Interrupt Enable Bit** 0 = EPWM Channel n Fault Detect Interrupt Disabled. 1 = EPWM Channel n Fault Detect Interrupt Enabled.

EPWM Fault Detect Interrupt Flag Register (EPWM_FDSTS)

Register	Offset	R/W	Description	Reset Value
EPWM_FDSTS	EPWMx_BA+0x180	R/W	EPWM Fault Detect 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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**FDIF5**	**FDIF4**	**FDIF3**	**FDIF2**	**FDIF1**	**FDIF0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**FDIFn**	**EPWM Channel n Fault Detect Interrupt Flag Bit** Fault Detect Interrupt Flag will be set when EPWM output short. Software can clear this bit by writing 1 to it.

EPWM Trigger EADC Prescale Control Register (EPWM_EADCPSCCTL)

Register	Offset	R/W	Description	Reset Value
EPWM_EADCPSCCTL	EPWMx_BA+0x184	R/W	EPWM Trigger EADC Prescale 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		PSCEN5	PSCEN4	PSCEN3	PSCEN2	PSCEN1	PSCEN0

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**PSCENn**	**EPWM Trigger EADC Pre-scale Function Enable Bits** 0 = EPWM Trigger EADC Pre-scale function Disabled. 1 = EPWM Trigger EADC Pre-scale function Enabled.

EPWM Trigger EADC Prescale Register 0 (EPWM_EADCPSC0)

Register	Offset	R/W	Description	Reset Value
EPWM_EADCPSC0	EPWMx_BA+0x188	R/W	EPWM Trigger EADC Prescale Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				EADCPSC3			
23	22	21	20	19	18	17	16
Reserved				EADCPSC2			
15	14	13	12	11	10	9	8
Reserved				EADCPSC1			
7	6	5	4	3	2	1	0
Reserved				EADCPSC0			

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:24]	**EADCPSC3**	**EPWM Channel 3 Trigger EADC Prescale** The register sets the count number which defines (EADCPSC3+1) times of EPWM_CH3 trigger EADC event occurs to trigger EADC and set trigger EADC flag bit EADCTRGF3.
[23:20]	**Reserved**	Reserved.
[19:16]	**EADCPSC2**	**EPWM Channel 2 Trigger EADC Prescale** The register sets the count number which defines (EADCPSC2+1) times of EPWM_CH2 trigger EADC event occurs to trigger EADC and set trigger EADC flag bit EADCTRGF2.
[15:12]	**Reserved**	Reserved.
[11:8]	**EADCPSC1**	**EPWM Channel 1 Trigger EADC Prescale** The register sets the count number which defines (EADCPSC1+1) times of EPWM_CH1 trigger EADC event occurs to trigger EADC and set trigger EADC flag bit EADCTRGF1.
[7:4]	**Reserved**	Reserved.
[3:0]	**EADCPSC0**	**EPWM Channel 0 Trigger EADC Prescale** The register sets the count number which defines (EADCPSC0+1) times of EPWM_CH0 trigger EADC event occurs to trigger EADC and set trigger EADC flag bit EADCTRGF0.

EPWM Trigger EADC Prescale Register 1 (EPWM_EADCPSC1)

Register	Offset	R/W	Description	Reset Value
EPWM_EADCPSC1	EPWMx_BA+0x18C	R/W	EPWM Trigger EADC Prescale 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				EADCPSC5			
7	6	5	4	3	2	1	0
Reserved				EADCPSC4			

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:8]	**EADCPSC5**	**EPWM Channel 5 Trigger EADC Prescale** The register sets the count number which defines (EADCPSC5+1) times of EPWM_CH5 trigger EADC event occurs to trigger EADC and set trigger EADC flag bit EADCTRGF5.
[7:4]	**Reserved**	Reserved.
[3:0]	**EADCPSC4**	**EPWM Channel 4 Trigger EADC Prescale** The register sets the count number which defines (EADCPSC4+1) times of EPWM_CH4 trigger EADC event occurs to trigger EADC and set trigger EADC flag bit EADCTRGF4.

EPWM Trigger EADC Prescale Counter Register 0 (EPWM_EADCPSCNT0)

Register	Offset	R/W	Description	Reset Value
EPWM_EADCPSCNT0	EPWMx_BA+0x190	R/W	EPWM Trigger EADC Prescale Counter Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				PSCNT3			
23	22	21	20	19	18	17	16
Reserved				PSCNT2			
15	14	13	12	11	10	9	8
Reserved				PSCNT1			
7	6	5	4	3	2	1	0
Reserved				PSCNT0			

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:24]	**PSCNT3**	**EPWM Trigger EADC Prescale Counter 3** User can monitor PSCNT3 to know the current value in 4-bit trigger EADC prescale counter. **Note 1:** user can write only when PSCEN3 is 0. **Note 2:** Write data limitation: PSCNT3 < EADCPSC3.
[23:20]	**Reserved**	Reserved.
[19:16]	**PSCNT2**	**EPWM Trigger EADC Prescale Counter 2** User can monitor PSCNT2 to know the current value in 4-bit trigger EADC prescale counter. **Note 1:** user can write only when PSCEN2 is 0. **Note 2:** Write data limitation: PSCNT2 < EADCPSC2.
[15:12]	**Reserved**	Reserved.
[11:8]	**PSCNT1**	**EPWM Trigger EADC Prescale Counter 1** User can monitor PSCNT1 to know the current value in 4-bit trigger EADC prescale counter. **Note 1:** user can write only when PSCEN1 is 0. **Note 2:** Write data limitation: PSCNT1 < EADCPSC1.
[7:4]	**Reserved**	Reserved.
[3:0]	**PSCNT0**	**EPWM Trigger EADC Prescale Counter 0** User can monitor PSCNT0 to know the current value in 4-bit trigger EADC prescale counter. **Note 1:** user can write only when PSCEN0 is 0. **Note 2:** Write data limitation: PSCNT0 < EADCPSC0.

EPWM Trigger EADC Prescale Counter Register 1 (EPWM_EADCPSCNT1)

Register	Offset	R/W	Description	Reset Value
EPWM_EADCPSCNT1	EPWMx_BA+0x194	R/W	EPWM Trigger EADC Prescale Counter 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				PSCNT5			
7	6	5	4	3	2	1	0
Reserved				PSCNT4			

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:8]	**PSCNT5**	**EPWM Trigger EADC Prescale Counter 5** User can monitor PSCNT5 to know the current value in 4-bit trigger EADC prescale counter. **Note 1:** user can write only when PSCEN5 is 0. **Note 2:** Write data limitation: PSCNT5 < EADCPSC5.
[7:4]	**Reserved**	Reserved.
[3:0]	**PSCNT4**	**EPWM Trigger EADC Prescale Counter 4** User can monitor PSCNT4 to know the current value in 4-bit trigger EADC prescale counter. **Note 1:** user can write only when PSCEN4 is 0. **Note 2:** Write data limitation: PSCNT4 < EADCPSC4.

EPWM Capture Input Enable Register (EPWM_CAPINEN)

Register	Offset	R/W	Description	Reset Value
EPWM_CAPINEN	EPWMx_BA+0x200	R/W	EPWM 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.
[n] n=0,1..5	**CAPINENn**	**Capture Input Enable Bits** 0 = EPWM Channel capture input path Disabled. The input of EPWM channel capture function is always regarded as 0. 1 = EPWM Channel capture input path Enabled. The input of EPWM channel capture function comes from correlative multifunction pin.

EPWM Capture Control Register (EPWM_CAPCTL)

Register	Offset	R/W	Description	Reset Value
EPWM_CAPCTL	EPWMx_BA+0x204	R/W	EPWM 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.
[24+n] n=0,1..5	**FCRLDENn**	**Falling Capture Reload Enable Bits** 0 = Falling capture reload counter Disabled. 1 = Falling capture reload counter Enabled.
[23:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**RCRLDENn**	**Rising Capture Reload Enable Bits** 0 = Rising capture reload counter Disabled. 1 = Rising capture reload counter Enabled.
[15:14]	**Reserved**	Reserved.
[8+n] n=0,1..5	**CAPINVn**	**Capture Inverter Enable Bits** 0 = Capture source inverter Disabled. 1 = Capture source inverter Enabled. Reverse the input signal from GPIO.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**CAPENn**	**Capture Function Enable Bits** 0 = Capture function Disabled. EPWM_RCAPDATn/EPWM_FCAPDATn register will not be updated. 1 = Capture function Enabled. Capture latched the EPWM counter value when detected rising or falling edge of input signal and saved to RCAPDAT (Rising latch) and FCAPDAT (Falling latch).

EPWM Capture Status Register (EPWM_CAPSTS)

Register	Offset	R/W	Description	Reset Value
EPWM_CAPSTS	EPWMx_BA+0x208	R	EPWM 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.
[8+n] n=0,1..5	**CFLIFOVn**	**Capture Falling Latch Interrupt Flag Overrun Status (Read Only)** This flag indicates if falling latch happened when the corresponding CFLIFn(EPWM_CAPIF[8+n]) is 1. **Note:** This bit will be cleared automatically when user clear corresponding CFLIFn(EPWM_CAPIF[8+n]).
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**CRLIFOVn**	**Capture Rising Latch Interrupt Flag Overrun Status (Read Only)** This flag indicates if rising latch happened when the corresponding CRLIFn(EPWM_CAPIF[n]) is 1. **Note:** This bit will be cleared automatically when user clear corresponding CRLIFn(EPWM_CAPIF[n]).

EPWM Rising Capture Data Register 0~5 (EPWM_RCAPDAT 0~5)

Register	Offset	R/W	Description	Reset Value
EPWM_RCAPDAT0	EPWMx_BA+0x20C	R	EPWM Rising Capture Data Register 0	0x0000_0000
EPWM_RCAPDAT1	EPWMx_BA+0x214	R	EPWM Rising Capture Data Register 1	0x0000_0000
EPWM_RCAPDAT2	EPWMx_BA+0x21C	R	EPWM Rising Capture Data Register 2	0x0000_0000
EPWM_RCAPDAT3	EPWMx_BA+0x224	R	EPWM Rising Capture Data Register 3	0x0000_0000
EPWM_RCAPDAT4	EPWMx_BA+0x22C	R	EPWM Rising Capture Data Register 4	0x0000_0000
EPWM_RCAPDAT5	EPWMx_BA+0x234	R	EPWM Rising Capture Data Register 5	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**RCAPDAT**	**EPWM Rising Capture Data Register (Read Only)** When rising capture condition happened, the EPWM counter value will be saved in this register.

EPWM Falling Capture Data Register 0~5 (EPWM_FCAPDAT 0~5)

Register	Offset	R/W	Description	Reset Value
EPWM_FCAPDAT0	EPWMx_BA+0x210	R	EPWM Falling Capture Data Register 0	0x0000_0000
EPWM_FCAPDAT1	EPWMx_BA+0x218	R	EPWM Falling Capture Data Register 1	0x0000_0000
EPWM_FCAPDAT2	EPWMx_BA+0x220	R	EPWM Falling Capture Data Register 2	0x0000_0000
EPWM_FCAPDAT3	EPWMx_BA+0x228	R	EPWM Falling Capture Data Register 3	0x0000_0000
EPWM_FCAPDAT4	EPWMx_BA+0x230	R	EPWM Falling Capture Data Register 4	0x0000_0000
EPWM_FCAPDAT5	EPWMx_BA+0x238	R	EPWM Falling Capture Data Register 5	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**FCAPDAT**	**EPWM Falling Capture Data Register (Read Only)** When falling capture condition happened, the EPWM counter value will be saved in this register.

EPWM PDMA Control Register (EPWM_PDMACTL)

Register	Offset	R/W	Description	Reset Value
EPWM_PDMACTL	EPWMx_BA+0x23C	R/W	EPWM PDMA Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			CHSEL4_5	CAPORD4_5	CAPMOD4_5		CHEN4_5
15	14	13	12	11	10	9	8
Reserved			CHSEL2_3	CAPORD2_3	CAPMOD2_3		CHEN2_3
7	6	5	4	3	2	1	0
Reserved			CHSEL0_1	CAPORD0_1	CAPMOD0_1		CHEN0_1

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20]	**CHSEL4_5**	**Select Channel 4/5 to Do PDMA Transfer** 0 = Channel4. 1 = Channel5.
[19]	**CAPORD4_5**	**Capture Channel 4/5 Rising/Falling Order** Set this bit to determine whether the EPWM_RCAPDAT4/5 or EPWM_FCAPDAT4/5 is the first captured data transferred to memory through PDMA when CAPMOD4_5 =11. 0 = EPWM_FCAPDAT4/5 is the first captured data to memory. 1 = EPWM_RCAPDAT4/5 is the first captured data to memory.
[18:17]	**CAPMOD4_5**	**Select EPWM_RCAPDAT4/5 or EPWM_FCAPDAT4/5 to Do PDMA Transfer** 00 = Reserved. 01 = EPWM_RCAPDAT4/5. 10 = EPWM_FCAPDAT4/5. 11 = Both EPWM_RCAPDAT4/5 and EPWM_FCAPDAT4/5.
[16]	**CHEN4_5**	**Channel 4/5 PDMA Enable Bit** 0 = Channel 4/5 PDMA function Disabled. 1 = Channel 4/5 PDMA function Enabled for the channel 4/5 captured data and transfer to memory.
[15:13]	**Reserved**	Reserved.
[12]	**CHSEL2_3**	**Select Channel 2/3 to Do PDMA Transfer** 0 = Channel2. 1 = Channel3.
[11]	**CAPORD2_3**	**Capture Channel 2/3 Rising/Falling Order** Set this bit to determine whether the EPWM_RCAPDAT2/3 or EPWM_FCAPDAT2/3 is the first captured data transferred to memory through PDMA when CAPMOD2_3 =11. 0 = EPWM_FCAPDAT2/3 is the first captured data to memory. 1 = EPWM_RCAPDAT2/3 is the first captured data to memory.

[10:9]	**CAPMOD2_3**	**Select EPWM_RCAPDAT2/3 or EPWM_FCAODAT2/3 to Do PDMA Transfer** 00 = Reserved. 01 = EPWM_RCAPDAT2/3. 10 = EPWM_FCAPDAT2/3. 11 = Both EPWM_RCAPDAT2/3 and EPWM_FCAPDAT2/3.
[8]	**CHEN2_3**	**Channel 2/3 PDMA Enable Bit** 0 = Channel 2/3 PDMA function Disabled. 1 = Channel 2/3 PDMA function Enabled for the channel 2/3 captured data and transfer to memory.
[7:5]	**Reserved**	Reserved.
[4]	**CHSEL0_1**	**Select Channel 0/1 to Do PDMA Transfer** 0 = Channel0. 1 = Channel1.
[3]	**CAPORD0_1**	**Capture Channel 0/1 Rising/Falling Order** Set this bit to determine whether the EPWM_RCAPDAT0/1 or EPWM_FCAPDAT0/1 is the first captured data transferred to memory through PDMA when CAPMOD0_1 =11. 0 = EPWM_FCAPDAT0/1 is the first captured data to memory. 1 = EPWM_RCAPDAT0/1 is the first captured data to memory.
[2:1]	**CAPMOD0_1**	**Select EPWM_RCAPDAT0/1 or EPWM_FCAPDAT0/1 to Do PDMA Transfer** 00 = Reserved. 01 = EPWM_RCAPDAT0/1. 10 = EPWM_FCAPDAT0/1. 11 = Both EPWM_RCAPDAT0/1 and EPWM_FCAPDAT0/1.
[0]	**CHEN0_1**	**Channel 0/1 PDMA Enable Bit** 0 = Channel 0/1 PDMA function Disabled. 1 = Channel 0/1 PDMA function Enabled for the channel 0/1 captured data and transfer to memory.

EPWM Capture Channel 0_1, 2_3, 4_5 PDMA Register (EPWM_PDMACAP 0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
EPWM_PDMACAP0_1	EPWMx_BA+0x240	R	EPWM Capture Channel 01 PDMA Register	0x0000_0000
EPWM_PDMACAP2_3	EPWMx_BA+0x244	R	EPWM Capture Channel 23 PDMA Register	0x0000_0000
EPWM_PDMACAP4_5	EPWMx_BA+0x248	R	EPWM Capture Channel 45 PDMA Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CAPBUF**	**EPWM Capture PDMA Register (Read Only)** This register is used as a buffer to transfer EPWM capture rising or falling data to memory by PDMA.

EPWM Capture Interrupt Enable Register (EPWM_CAPIEN)

Register	Offset	R/W	Description	Reset Value
EPWM_CAPIEN	EPWMx_BA+0x250	R/W	EPWM 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.
[8+n] n=0,1..5	**CAPFIENn**	**EPWM Capture Falling Latch Interrupt Enable Bits** 0 = Capture falling edge latch interrupt Disabled. 1 = Capture falling edge latch interrupt Enabled.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**CAPRIENn**	**EPWM Capture Rising Latch Interrupt Enable Bits** 0 = Capture rising edge latch interrupt Disabled. 1 = Capture rising edge latch interrupt Enabled.

EPWM Capture Interrupt Flag Register (EPWM_CAPIF)

Register	Offset	R/W	Description	Reset Value
EPWM_CAPIF	EPWMx_BA+0x254	R/W	EPWM 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.
[8+n] n=0,1..5	**CFLIFn**	**EPWM Capture Falling Latch Interrupt Flag** 0 = No capture falling latch condition happened. 1 = Capture falling latch condition happened, this flag will be set to high. **Note 1:** When Capture with PDMA operating, EPWM_CAPIF corresponding channel CFLIFn will be cleared by hardware after PDMA transfer data. **Note 2:** This bit is cleared by writing 1 to it.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**CRLIFn**	**EPWM Capture Rising Latch Interrupt Flag** 0 = No capture rising latch condition happened. 1 = Capture rising latch condition happened, this flag will be set to high. **Note 1:** When Capture with PDMA operating, EPWM_CAPIF corresponding channel CRLIFn will be cleared by hardware after PDMA transfer data. **Note 2:** This bit is cleared by writing 1 to it.

EPWM Capture Input Noise Filter Register (EPWM_CAPNF0~5)

Register	Offset	R/W	Description	Reset Value
EPWM_CAPNF0	EPWMx_BA+0x258	R/W	EPWM Capture Input Noise Filter Register 0	0x0000_0000
EPWM_CAPNF1	EPWMx_BA+0x25C	R/W	EPWM Capture Input Noise Filter Register 1	0x0000_0000
EPWM_CAPNF2	EPWMx_BA+0x260	R/W	EPWM Capture Input Noise Filter Register 2	0x0000_0000
EPWM_CAPNF3	EPWMx_BA+0x264	R/W	EPWM Capture Input Noise Filter Register 3	0x0000_0000
EPWM_CAPNF4	EPWMx_BA+0x268	R/W	EPWM Capture Input Noise Filter Register 4	0x0000_0000
EPWM_CAPNF5	EPWMx_BA+0x26C	R/W	EPWM Capture Input Noise Filter 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
Reserved					CAPNFCNT		
7	6	5	4	3	2	1	0
Reserved	CAPNFSEL			Reserved			CAPNFEN

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10:8]	**CAPNFCNT**	**Capture Edge Detector Noise Filter Count** The register bits control the capture filter counter to count from 0 to CAPNFCNT.
[6:4]	**CAPNFSEL**	**Capture Edge Detector Noise Filter Clock Selection** 000 = Filter clock = PCLK. 001 = Filter clock = PCLK/2. 010 = Filter clock = PCLK/4. 011 = Filter clock = PCLK/8. 100 = Filter clock = PCLK/16. 101 = Filter clock = PCLK/32. 110 = Filter clock = PCLK/64. 111 = Filter clock = PCLK/128.
[3:1]	**Reserved**	Reserved.
[0]	**CAPNFEN**	**Capture Noise Filter Enable** 0 = Capture Noise Filter function Disabled. 1 = Capture Noise Filter function Enabled.

EPWM External Event Trigger Control Register 0~5 (EPWMx_EXTETCTL0~5)

Register	Offset	R/W	Description	Reset Value
EPWM_EXTETCTL0	EPWMx_BA+0x270	R/W	EPWM External Event Trigger Control Register 0	0x0000_0000
EPWM_EXTETCTL1	EPWMx_BA+0x274	R/W	EPWM External Event Trigger Control Register 1	0x0000_0000
EPWM_EXTETCTL2	EPWMx_BA+0x278	R/W	EPWM External Event Trigger Control Register 2	0x0000_0000
EPWM_EXTETCTL3	EPWMx_BA+0x27C	R/W	EPWM External Event Trigger Control Register 3	0x0000_0000
EPWM_EXTETCTL4	EPWMx_BA+0x280	R/W	EPWM External Event Trigger Control Register 4	0x0000_0000
EPWM_EXTETCTL5	EPWMx_BA+0x284	R/W	EPWM External Event Trigger Control 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
Reserved				EXTTRGS			
7	6	5	4	3	2	1	0
Reserved		CNTACTS		Reserved			EXTETEN

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:8]	**EXTTRGS**	**External Trigger Selection** 0000 = INT0 0001 = INT1 0010 = INT2 0011 = INT3 0100 = INT4 0101 = INT5 0110 = INT6 0111 = INT7 Other = Resrved
[5:4]	**CNTACTS**	**Counter Action Selection** 00 = Counter reset 01 = Counter start 10 = Counter reset and start 11 = Reseved
[3:1]	**Reserved**	Reserved.
[0]	**EXETEN**	**External Event Trigger Enable Bit** 0 = External Event Trigger function Disabled.

		1 = External Event Trigger function Enabled.

EPWM Software Event Output Force Control Register (EPWMx_SWEOFCTL)

Register	Offset	R/W	Description	Reset Value
EPWM_SWEOFCTL	EPWMx_BA+0x288	R/W	EPWM Software Event Output Force 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				**OUTACTS5**		**OUTACTS4**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OUTACTS3		**OUTACTS2**		**OUTACTS1**		**OUTACTS0**	

Bits	Description	
[31:12]	**Reserved**	Reserved.
[2n+1:2n] n=0,1..5	**OUTACTSn**	**Output Action Selection** 00 = Do nothing 01 = EPWM output Low. 10 = EPWM output High. 11 = EPWM output Toggle.

EPWM Software Event Output Force Trigger Register (EPWMx_SWEOFTRG)

Register	Offset	R/W	Description	Reset Value
EPWM_SWEOFTRG	EPWMx_BA+0x28C	R/W	EPWM Software Event Output Force 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		**SWETRG5**	**SWETRG4**	**SWETRG3**	**SWETRG2**	**SWETRG1**	**SWETRG0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**SWETRGn**	**Software Event Trigger** Write 1 to this bit will change EPWM output status according to OUTACTSn in EPWMx_SWEOFCTL setting. **Note:** This bit will auto cleared by hardware.

EPWM Clock Prescale Register 0~5 (EPWM_CLKPSC0~5)

Register	Offset	R/W	Description	Reset Value
EPWM_CLKPSC0	EPWMx_BA+0x290	R/W	EPWM Clock Prescale Register 0	0x0000_0000
EPWM_CLKPSC1	EPWMx_BA+0x294	R/W	EPWM Clock Prescale Register 1	0x0000_0000
EPWM_CLKPSC2	EPWMx_BA+0x298	R/W	EPWM Clock Prescale Register 2	0x0000_0000
EPWM_CLKPSC3	EPWMx_BA+0x29C	R/W	EPWM Clock Prescale Register 3	0x0000_0000
EPWM_CLKPSC4	EPWMx_BA+0x2A0	R/W	EPWM Clock Prescale Register 4	0x0000_0000
EPWM_CLKPSC5	EPWMx_BA+0x2A4	R/W	EPWM Clock Prescale 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
Reserved				CLKPSC			
7	6	5	4	3	2	1	0
CLKPSC							

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

EPWM Rising Dead-time Counter Register 0_1, 2_3, 4_5 (EPWM_RDTCNT0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
EPWM_RDTCNT0_1	EPWMx_BA+0x2A8	R/W	EPWM Rising Dead-time Counter Register 0/1	0x0000_0000
EPWM_RDTCNT2_3	EPWMx_BA+0x2AC	R/W	EPWM Rising Dead-time Counter Register 2/3	0x0000_0000
EPWM_RDTCNT4_5	EPWMx_BA+0x2B0	R/W	EPWM Rising Dead-time Counter 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				RDTCNT			
7	6	5	4	3	2	1	0
RDTCNT							

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:0]	**RDTCNT**	**Rising Dead-time Counter (Write Protect)** The Rising dead-time can be calculated from the following formula: RDTCKSEL=0: Rising Dead-time = (RDTCNT[11:0]+1) * EPWM_CLK period. RDTCKSEL=1: Rising Dead-time = (RDTCNT[11:0]+1) * EPWM_CLK period * (CLKPSC+1). **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

EPWM Falling Dead-time Counter Register 0_1, 2_3, 4_5 (EPWM_FDTCNT0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
EPWM_FDTCNT0_1	EPWMx_BA+0x2B4	R/W	EPWM Falling Dead-time Counter Register 0/1	0x0000_0000
EPWM_FDTCNT2_3	EPWMx_BA+0x2B8	R/W	EPWM Falling Dead-time Counter Register 2/3	0x0000_0000
EPWM_FDTCNT4_5	EPWMx_BA+0x2BC	R/W	EPWM Falling Dead-time Counter 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				FDTCNT			
7	6	5	4	3	2	1	0
FDTCNT							

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:0]	**FDTCNT**	**Falling Dead-time Counter (Write Protect)** The dead-time can be calculated from the following formula: FDTCKSEL=0: Falling Dead-time = (FDTCNT[11:0]+1) * EPWM_CLK period. FDTCKSEL=1: Falling Dead-time = (FDTCNT[11:0]+1) * EPWM_CLK period * (CLKPSC+1). **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

EPWM Dead-time Control Register (EPWM_DTCTL)

Register	Offset	R/W	Description	Reset Value
EPWM_DTCTL	EPWMx_BA+0x2C0	R/W	EPWM Dead-time Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					DTCKSEL4	DTCKSEL2	DTCKSEL0
15	14	13	12	11	10	9	8
Reserved					FDTEN4	FDTEN2	FDTEN0
7	6	5	4	3	2	1	0
Reserved					RDTEN4	RDTEN2	RDTEN0

Bits	Description	
[31:17]	**Reserved**	Reserved.
[n/2+16] n=0,2,4	**DTCKSELn**	**Dead-time Clock Select for EPWM Pair (EPWM_CH(n/2), EPWM_CH(n/2+1)) (Write Protect)** 0 = Dead-time clock source from EPWM_CLK. 1 = Dead-time clock source from prescaler output. **Note:** This bit is write protected. Refer to REGWRPROT register.
[15:11]	**Reserved**	Reserved.
[n/2+8] n=0,2,4	**FDTENn**	**Enable Falling Dead-time Insertion for EPWM Pair (EPWM_CH(n/2), EPWM_CH(n/2+1)) (Write Protect)** Falling Dead-time insertion is only active when this pair of complementary EPWM is enabled. If falling dead- time insertion is inactive, the outputs of pin pair are complementary without any delay. 0 = Falling Dead-time insertion Disabled on the pin pair. 1 = Falling Dead-time insertion Enabled on the pin pair. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[7:3]	**Reserved**	Reserved.
[n/2] n=0,2,4	**RDTENn**	**Enable Rising Dead-time Insertion for EPWM Pair (EPWM_CH(n/2), EPWM_CH(n/2+1)) (Write Protect)** Rising Dead-time insertion is only active when this pair of complementary EPWM is enabled. If rising dead- time insertion is inactive, the outputs of pin pair are complementary without any delay. 0 = Rising Dead-time insertion Disabled on the pin pair. 1 = Rising Dead-time insertion Enabled on the pin pair. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

EPWM Period Register Buffer 0~5 (EPWM_PBUF0~5)

Register	Offset	R/W	Description	Reset Value
EPWM_PBUF0	EPWMx_BA+0x304	R	EPWM PERIOD0 Buffer	0x0000_0000
EPWM_PBUF1	EPWMx_BA+0x308	R	EPWM PERIOD1 Buffer	0x0000_0000
EPWM_PBUF2	EPWMx_BA+0x30C	R	EPWM PERIOD2 Buffer	0x0000_0000
EPWM_PBUF3	EPWMx_BA+0x310	R	EPWM PERIOD3 Buffer	0x0000_0000
EPWM_PBUF4	EPWMx_BA+0x314	R	EPWM PERIOD4 Buffer	0x0000_0000
EPWM_PBUF5	EPWMx_BA+0x318	R	EPWM 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.
[15:0]	**PBUF**	**EPWM Period Register Buffer (Read Only)** Used as PERIOD active register.

EPWM Comparator Register Buffer 0~5 (EPWM_CMPBUF0~5)

Register	Offset	R/W	Description	Reset Value
EPWM_CMPBUF0	EPWMx_BA+0x31C	R	EPWM CMPDAT0 Buffer	0x0000_0000
EPWM_CMPBUF1	EPWMx_BA+0x320	R	EPWM CMPDAT1 Buffer	0x0000_0000
EPWM_CMPBUF2	EPWMx_BA+0x324	R	EPWM CMPDAT2 Buffer	0x0000_0000
EPWM_CMPBUF3	EPWMx_BA+0x328	R	EPWM CMPDAT3 Buffer	0x0000_0000
EPWM_CMPBUF4	EPWMx_BA+0x32C	R	EPWM CMPDAT4 Buffer	0x0000_0000
EPWM_CMPBUF5	EPWMx_BA+0x330	R	EPWM CMPDAT5 Buffer	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CMPBUF**	**EPWM Comparator Register Buffer (Read Only)** Used as CMP active register.

EPWM CLKPSC Buffer 0_1, 2_3, 4_5 (EPWM_CPSCBUF0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
EPWM_CPSCBUF0_1	EPWMx_BA+0x334	R	EPWM CLKPSC0_1 Buffer	0x0000_0000
EPWM_CPSCBUF2_3	EPWMx_BA+0x338	R	EPWM CLKPSC2_3 Buffer	0x0000_0000
EPWM_CPSCBUF4_5	EPWMx_BA+0x33C	R	EPWM 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.
[11:0]	**CPSCBUF**	**EPWM Counter Clock Prescale Buffer** Used as EPWM counter clock pre-scale active register.

EPWM_FTCMPDAT Buffer (EPWM_FTCBUF0_1,2_3,4_5)

Register	Offset	R/W	Description	Reset Value
EPWM_FTCBUF0_1	EPWMx_BA+0x340	R	EPWM FTCMPDAT0_1 Buffer	0x0000_0000
EPWM_FTCBUF2_3	EPWMx_BA+0x344	R	EPWM FTCMPDAT2_3 Buffer	0x0000_0000
EPWM_FTCBUF4_5	EPWMx_BA+0x348	R	EPWM 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.
[15:0]	**FTCMPBUF**	**EPWM FTCMPDAT Buffer (Read Only)** Used as FTCMP active buffer.

EPWM FTCMPDAT Indicator Register (EPWM_FTCI)

Register	Offset	R/W	Description	Reset Value
EPWM_FTCI	EPWMx_BA+0x34C	R/W	EPWM 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.
[8+n/2] n=0,2,4	**FTCMDn**	**EPWM FTCMPDAT Down Indicator** Indicator is set by hardware when EPWM counter down count and reaches EPWM_FTCMPDATn, software can clear this bit by writing 1 to it.
[7:3]	**Reserved**	Reserved.
[n/2] n=0,2,4	**FTCMUn**	**EPWM FTCMPDAT Up Indicator** Indicator is set by hardware when EPWM counter up count and reaches EPWM_FTCMPDATn, software can clear this bit by writing 1 to it.

EPWM CLKPSC Buffer 0~5 (EPWM_CPSCBUF0~5)

Register	Offset	R/W	Description	Reset Value
EPWM_CPSCBUF0	EPWMx_BA+0x350	R	EPWM CLKPSC0 Buffer	0x0000_0000
EPWM_CPSCBUF1	EPWMx_BA+0x354	R	EPWM CLKPSC1 Buffer	0x0000_0000
EPWM_CPSCBUF2	EPWMx_BA+0x358	R	EPWM CLKPSC2 Buffer	0x0000_0000
EPWM_CPSCBUF3	EPWMx_BA+0x35C	R	EPWM CLKPSC3 Buffer	0x0000_0000
EPWM_CPSCBUF4	EPWMx_BA+0x360	R	EPWM CLKPSC4 Buffer	0x0000_0000
EPWM_CPSCBUF5	EPWMx_BA+0x364	R	EPWM CLKPSC5 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.
[11:0]	**CPSCBUF**	**EPWM Counter Clock Prescale Buffer** Used as EPWM counter clock pre-scale active register.

EPWM Interrupt Flag Accumulator Counter 0~5 (EPWM_IFACNT0~5)

Register	Offset	R/W	Description	Reset Value
EPWM_IFACNT0	EPWMx_BA+0x368	R	EPWM Interrupt Flag Accumulator Counter 0	0x0000_0000
EPWM_IFACNT1	EPWMx_BA+0x36C	R	EPWM Interrupt Flag Accumulator Counter 1	0x0000_0000
EPWM_IFACNT2	EPWMx_BA+0x370	R	EPWM Interrupt Flag Accumulator Counter 2	0x0000_0000
EPWM_IFACNT3	EPWMx_BA+0x374	R	EPWM Interrupt Flag Accumulator Counter 3	0x0000_0000
EPWM_IFACNT4	EPWMx_BA+0x378	R	EPWM Interrupt Flag Accumulator Counter 4	0x0000_0000
EPWM_IFACNT5	EPWMx_BA+0x37C	R	EPWM Interrupt Flag Accumulator Counter 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**
ACUCNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ACUCNT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**ACUCNT**	**Accumulator Counter (Read Only)** This value indicates how many interrupt are accumulated when using interrupt flag accumulator function.

6.13 Basic PWM Generator and Capture Timer (BPWM)

6.13.1 Overview

The chip provides two BPWM generators — BPWM0 and BPWM1 as shown in Figure 6.13-1. Each BPWM supports 6 channels of BPWM output or input capture. There is a 12-bit prescaler to support flexible clock to the 16-bit BPWM counter with 16-bit comparator. The BPWM counter supports up, down and up-down counter types, all 6 channels share one counter. BPWM uses the comparator compared with counter to generate events. These events are used to generate BPWM pulse, interrupt and trigger signal for EADC0 to start conversion. For BPWM output control unit, it supports polarity output, independent pin mask and tri-state output enable.

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

6.13.2 Features

6.13.2.1 BPWM Function Features

- Supports maximum clock frequency up to maximum PLL frequency.
- Supports up to two BPWM modules; each module provides 6 output channels
- Supports independent mode for BPWM output/Capture input channel
- Supports 12-bit prescalar from 1 to 4096
- Supports 16-bit resolution BPWM counter; each module provides 1 BPWM counter
  - Up, down and up/down counter operation type
- Supports mask function and tri-state enable for each BPWM pin
- Supports interrupt in the following events:
  - BPWM counter matches 0, period value or compared value
- Supports trigger EADC0 in the following events:
  - BPWM counter matches 0, period value or compared value

6.13.2.2 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

	M471
Trigger numbers for EADC	1

Table 6.13-1 BPWM Features Comparison Table

6.13.3 Block Diagram

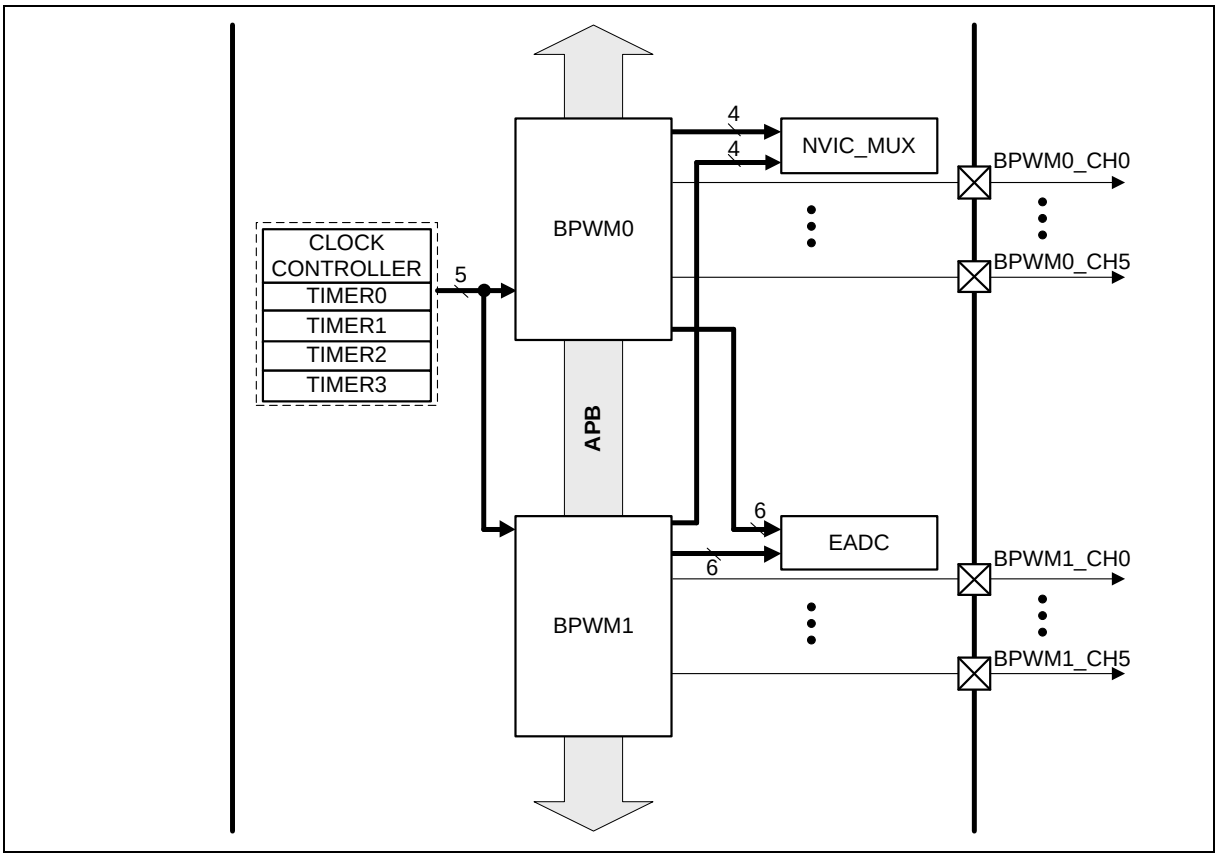


Figure 6.13-1 BPWM Generator Overview Block Diagram

Each BPWM generator has only one clock source inputs and can be selected from BPWM Clock or four TIMER trigger BPWM outputs as shown in Figure 6.13-2 by ECLKSRC0 (BPWM_CLKSRC[2:0]) for BPWM_CLK0. In general case, BPWM0 Clock must be selected from PCLK0 by setting BPWM0SEL (CLK_CLKSEL2[8]) to 1 and BPWM1 Clock from PCLK1 by setting BPWM1SEL (CLK_CLKSEL2[9]) to 1.

When operating in maximum HCLK clock frequency as shown in Figure 6.13-3, and Table 6.13-2. BPWM0 and BPWM1 clock must be selected to HCLK clock by setting BPWM0SEL (CLK_CLKSEL2[8]) and BPWM1SEL (CLK_CLKSEL2[9]) to 0.

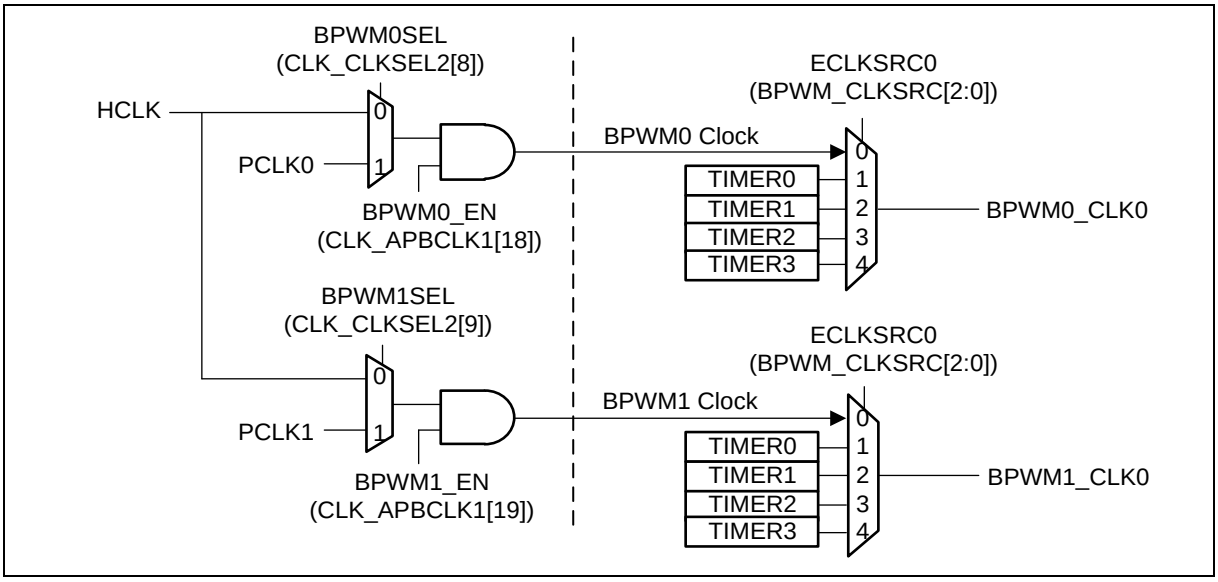


Figure 6.13-2 BPWM Clock Source Control

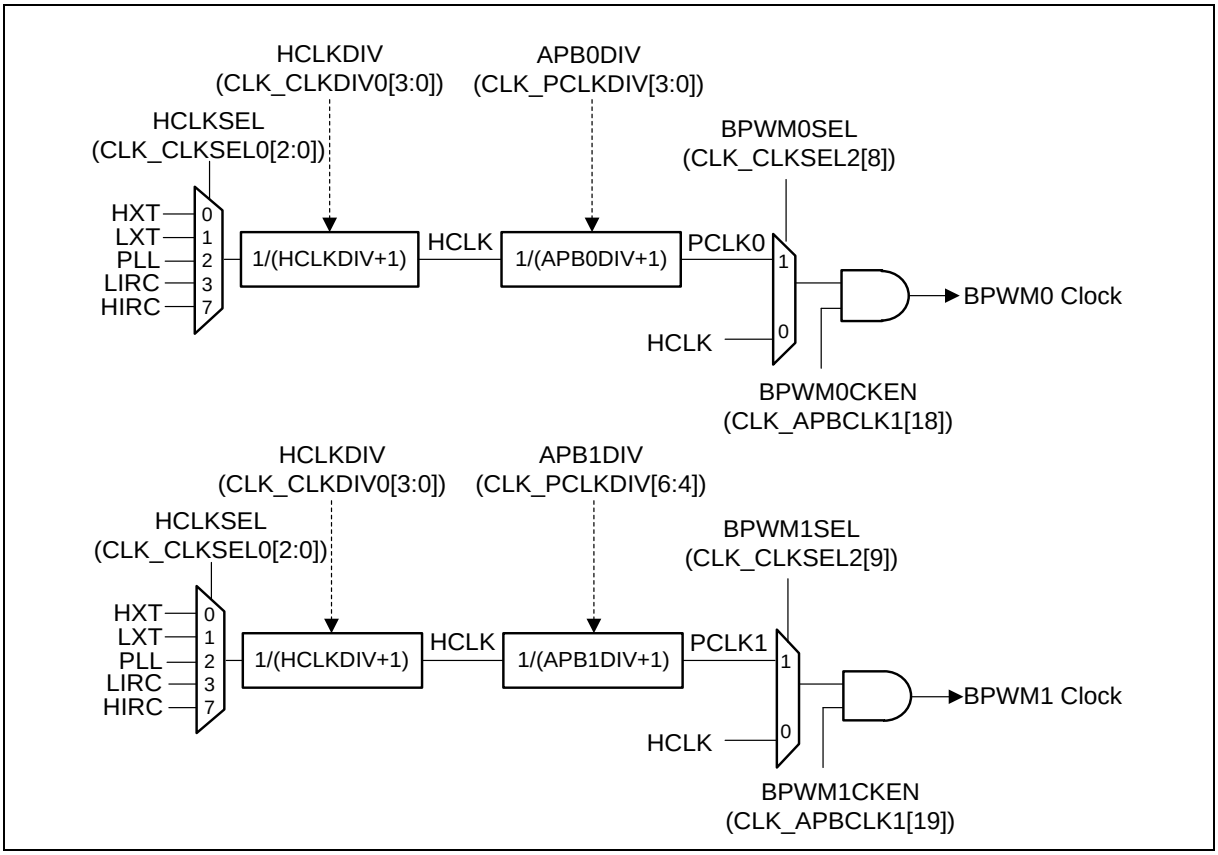


Figure 6.13-3 BPWM Clock Source Control

Frequency Ratio PCLK:BPWM Clock	HCLK	PCLK	BPWM Clock	HCLKSEL CLK_CLKSEL0[2:0]	HCLKDIV CLK_CLKDIV0[3:0]	APBnDIV (CLK_PCLKDIVn [2+4n:4n]), N Denotes 0 Or 1	BPWMnSEL (CLK_CLKSEL2[N+8]), N Denotes 0 Or 1
1:1	HCLK	PCLK	PCLK	Don't care	Don't care	Don't care	1
1:2	HCLK	HCLK/2	HCLK	Don't care	Don't care	1	0

Table 6.13-2 BPWM Clock Source Control Registers Setting Table

Figure 6.13-4 illustrates the architecture of BPWM Independent mode. All six channels share the same counter. When the counter counts to 0, PERIOD (BPWM_PERIOD[15:0]) or equal to the comparator, events will be generated. These events are passed to the corresponding generators to generate BPWM pulse, interrupt signal and trigger signal for EADC0 to start conversion. Output control is used to change the BPWM pulse output state.

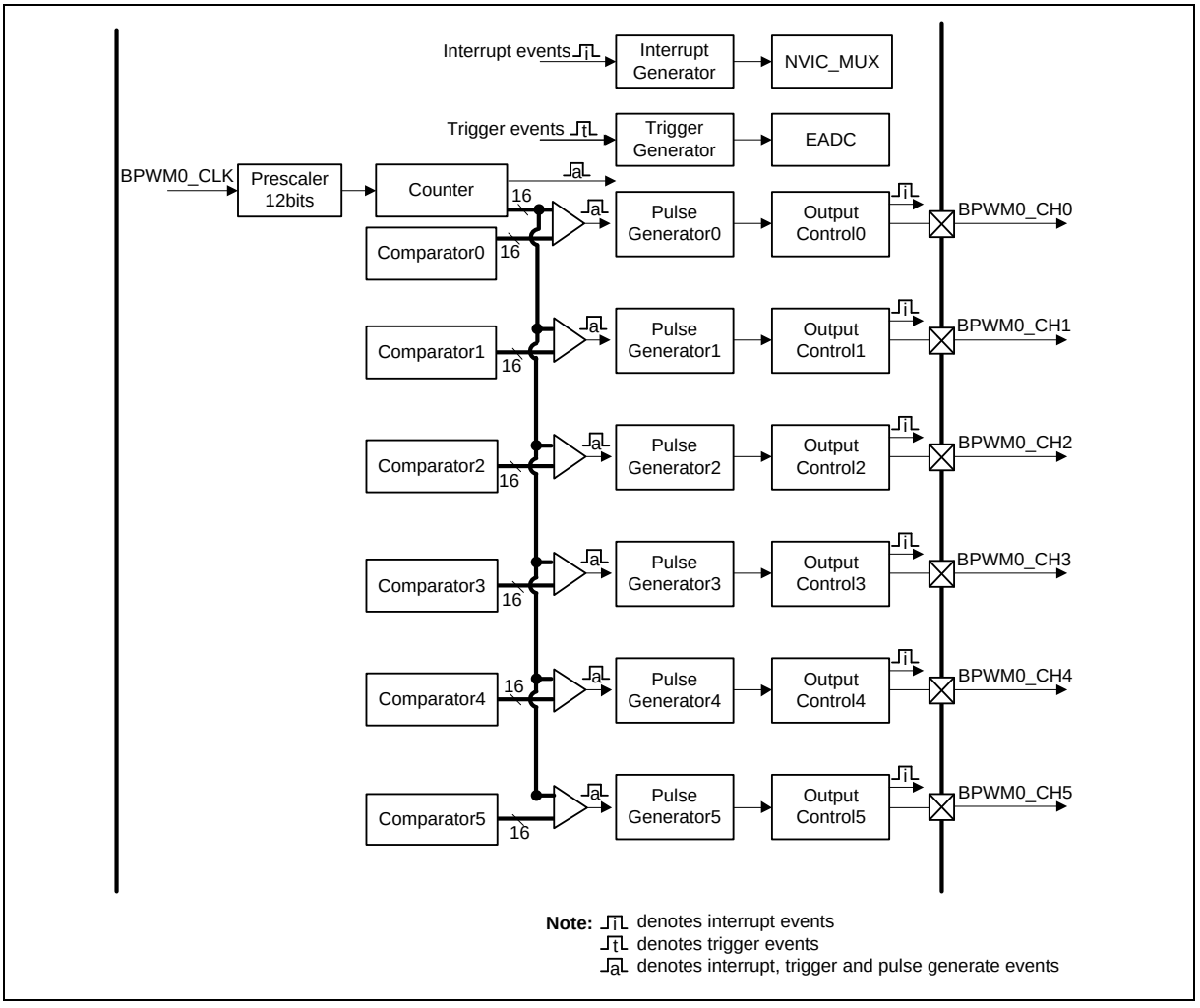


Figure 6.13-4 BPWM Independent Mode Architecture Diagram

6.13.4 Basic Configuration

6.13.4.1 BPWM0 Basic Configuration

- Clock Source Configuration

- Select the source of BPWM0 peripheral clock on BPWM0SEL (CLK_CLKSEL2[8]).
- Enable BPWM0 peripheral clock in BPWM0CKEN (CLK_APBCLK1[18]).

- Reset Configuration
  - Reset BPWM0 controller in BPWM0RST (SYS_IPRST2[18]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
BPWM0	BPWM0_CH0	PA.11	MFP9
		PA.0, PG.14	MFP12
		PE.2	MFP13
	BPWM0_CH1	PA.10	MFP9
		PA.1, PG.13	MFP12
		PE.3	MFP13
	BPWM0_CH2	PA.9	MFP9
		PA.2, PG.12	MFP12
		PE.4	MFP13
	BPWM0_CH3	PA.8	MFP9
		PA.3, PG.11	MFP12
		PE.5	MFP13
	BPWM0_CH4	PF.5	MFP8
		PC.13	MFP9
		PA.4, PG.10	MFP12
		PE.6	MFP13
	BPWM0_CH5	PF.4	MFP8
		PD.12	MFP9
		PA.5, PG.9	MFP12
		PE.7	MFP13

6.13.4.2 BPWM1 Basic Configuration

- Clock Source Configuration
  - Select the source of BPWM1 peripheral clock on BPWM1SEL (CLK_CLKSEL2[9]).
  - Enable BPWM1 peripheral clock in BPWM1CKEN (CLK_APBCLK1[19]).
- Reset Configuration
  - Reset BPWM1 controller in BPWM1RST (SYS_IPRST2[19]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
BPWM1	BPWM1_CH0	PB.11	MFP10
		PF.3	MFP11

		PC.7, PF.0	MFP12
	BPWM1_CH1	PB.10	MFP10
		PF.2	MFP11
		PC.6, PF.1	MFP12
	BPWM1_CH2	PB.9	MFP10
		PA.12	MFP11
		PA.7	MFP12
	BPWM1_CH3	PB.8	MFP10
		PA.13	MFP11
		PA.6	MFP12
	BPWM1_CH4	PB.7	MFP10
		PA.14	MFP11
		PC.8	MFP12
	BPWM1_CH5	PB.6	MFP10
		PA.15	MFP11
		PE.13	MFP12

6.13.5 Functional Description

6.13.5.1 BPWM Prescaler

The BPWM prescaler is used to divide clock source, prescaler counting CLKPSC +1 times, and BPWM counter only count once. The prescale is set by CLKPSC (BPWM_CLKPSC[11:0]). Figure 6.13-5 shows an example of BPWM channel 0 CLKPSC waveform. The prescale counter will reload CLKPSC in the beginning of the next prescale counter down-count.

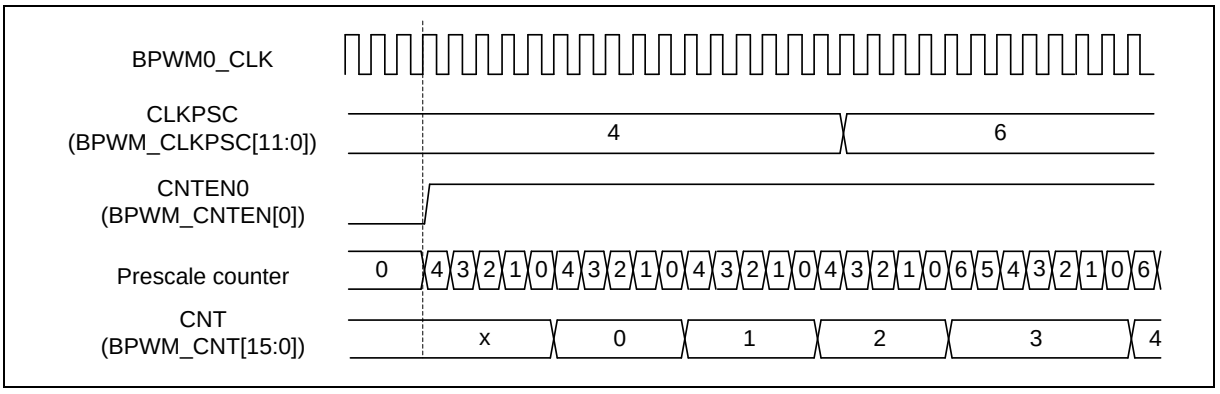


Figure 6.13-5 BPWM_CH0 CLKPSC Waveform

6.13.5.2 BPWM Counter

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

For BPWM channel0, CNT(BPWM_CNT[15:0]) can clear to 0x00 by CNTCLR0 (BPWM_CNTCLR[0]) when prescale counter down count to 0, and CNTCLR0(BPWM_CNTCLR[0]) will be set 0 by hardware automatically.

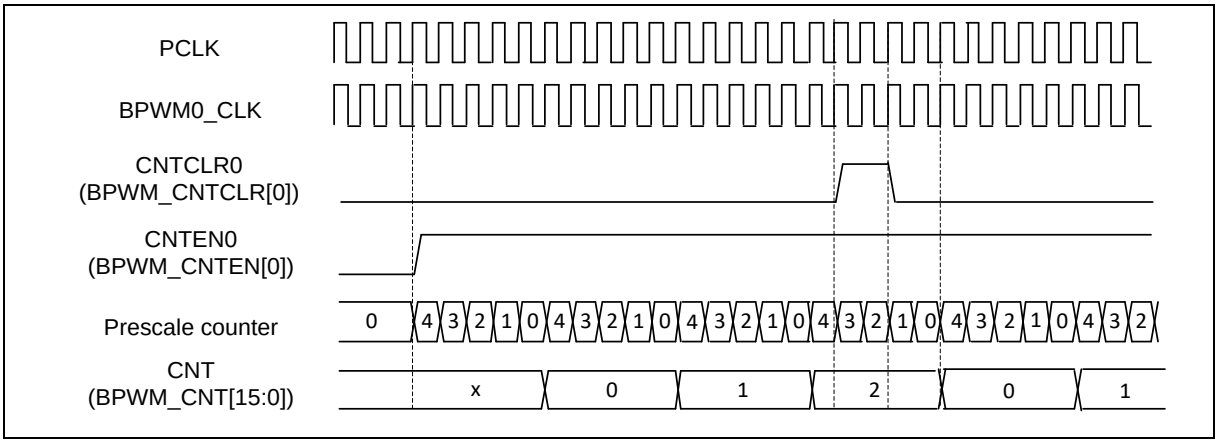


Figure 6.13-6 BPWM Counter Clear Waveform

6.13.5.3 Up Counter Type

In the up counter operation, the 16 bits BPWM counter is an up counter and starts up-counting from 0 to PERIOD (BPWM_PERIOD) to finish a BPWM period. The current counter value can be found by reading the CNT (BPWM_CNT[15:0]). BPWM generates zero point event when counter counts to 0 and generates period point event when counting to PERIOD. An example of the period time in up counter type, the BPWM period time = (PERIOD+1) * (CLKPSC+1) * BPWMx_CLK clock time, as shown in Figure 6.13-7.

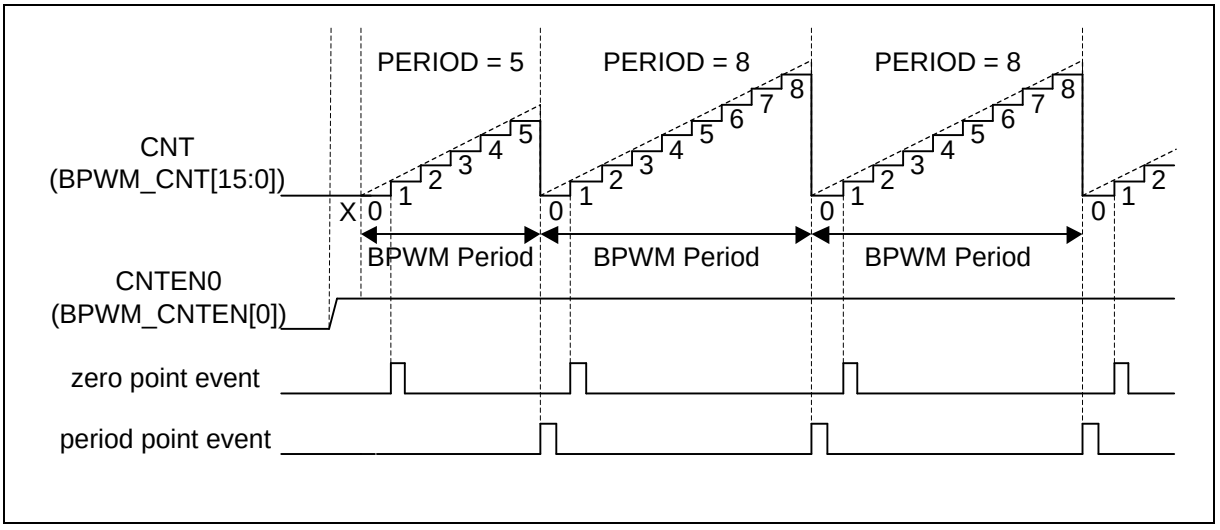


Figure 6.13-7 BPWM Up Counter Type

6.13.5.4 Down Counter Type

In the down counter operation, the 16 bits BPWM counter is a down counter and starts down-counting from PERIOD to 0 to finish a BPWM period. The current counter value can be found by reading the CNT. BPWM generates zero point event when counter counts to 0 and generates period point event when counting to PERIOD. An example of the period time in down counter type, the BPWM period time = (PERIOD+1) * (CLKPSC+1) * BPWMx_CLK clock time, as shown in Figure 6.13-8.

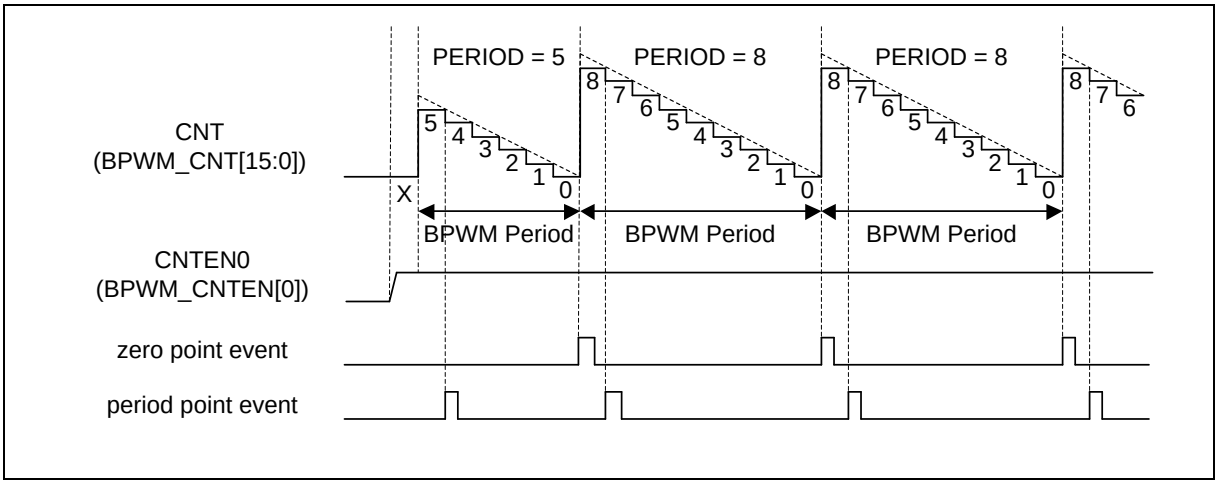
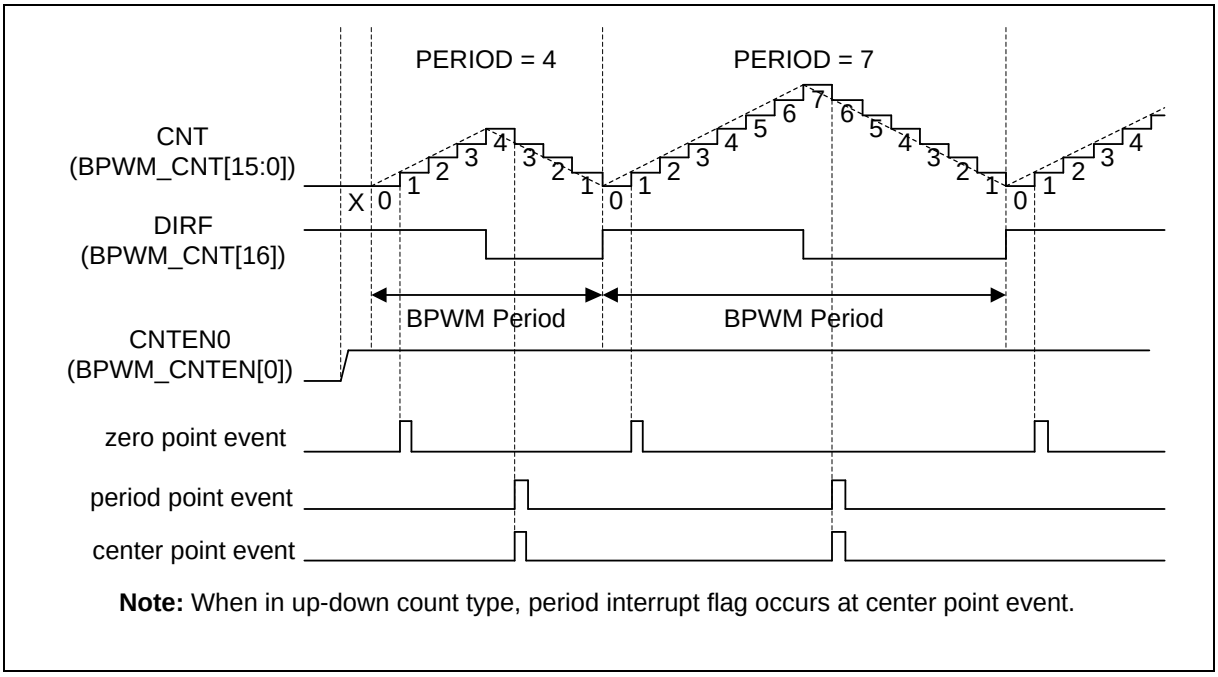


Figure 6.13-8 BPWM Down Counter Type

6.13.5.5 Up-Down Counter Type

In the up-down counter operation, the 16 bits BPWM counter is an up-down counter and starts counting-up from 0 to PERIOD and then starts counting down to 0 to finish a BPWM period. The current counter value can be found by reading the CNT. BPWM generates zero point event when counter counts to 0 and generates center point event when counting to PERIOD. An example of the period time in up-down counter type, the BPWM period time = (2xPERIOD) * (CLKPSC+1) * BPWMx_CLK clock time,as shown in Figure 6.13-9. The DIRF (BPWM_CNT[16]) is counter direction indicator flag, where high is up counting, and low is down counting.



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

Figure 6.13-9 BPWM Up-Down Counter Type

6.13.5.6 BPWM Comparator

The CMPDAT (BPWM_CMPDATn[15:0]) is a basic comparator register of BPWM channel n; each channel only has one CMPDAT. The CMPDAT's value is continuously compared to the counter value. When the counter is equal to compared register, BPWM generates an event and uses the event to generate BPWM pulse, interrupt or use to trigger EADC0. In up-down counter type, two events will be

generated in a BPWM period as shown in Figure 6.13-10.

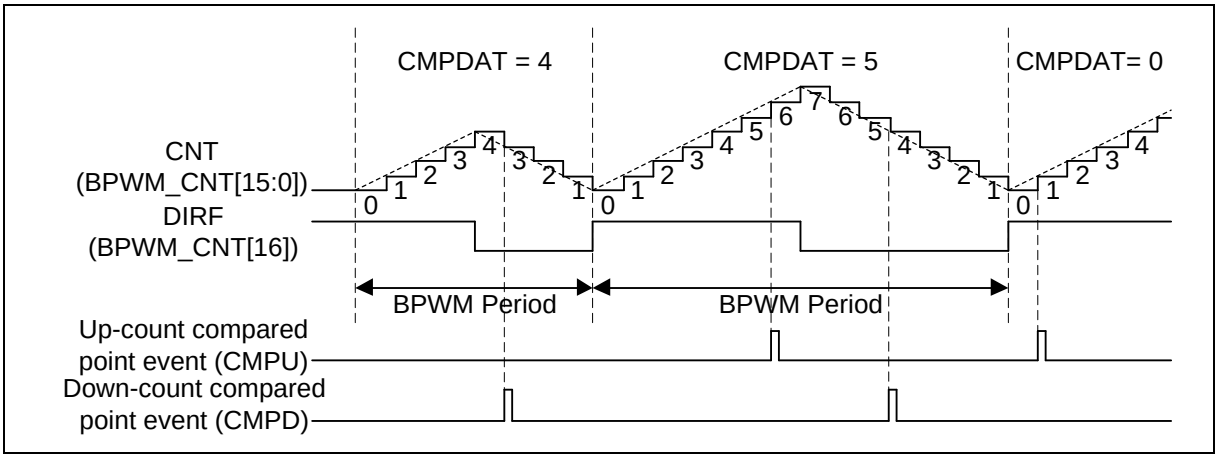


Figure 6.13-10 BPWM CMPDAT Events in Up-Down Counter Type

6.13.5.7 Period Loading Mode

Period Loading mode is the default loading mode. It has lowest priority in loading modes. PERIOD and CMPDAT will both load to their buffer while a period is completed. For example, after BPWM counter up counts from 0 to PERIOD in up-counter operation or down counts from PERIOD to 0 in the down-counter operation or up counts from 0 to PERIOD and then down counts to 0 in up-down counter operation.

Figure 6.13-11 shows period loading timing of up-count operation, where PERIOD DATA0 denotes the initial data of PERIOD, PERIOD DATA1 denotes the first updated PERIOD data by software and so on, CMPDAT also follows this rule. The following describes steps sequence of Figure 6.13-11. User can know the PERIOD and CMPDAT update condition, by watching BPWM 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.

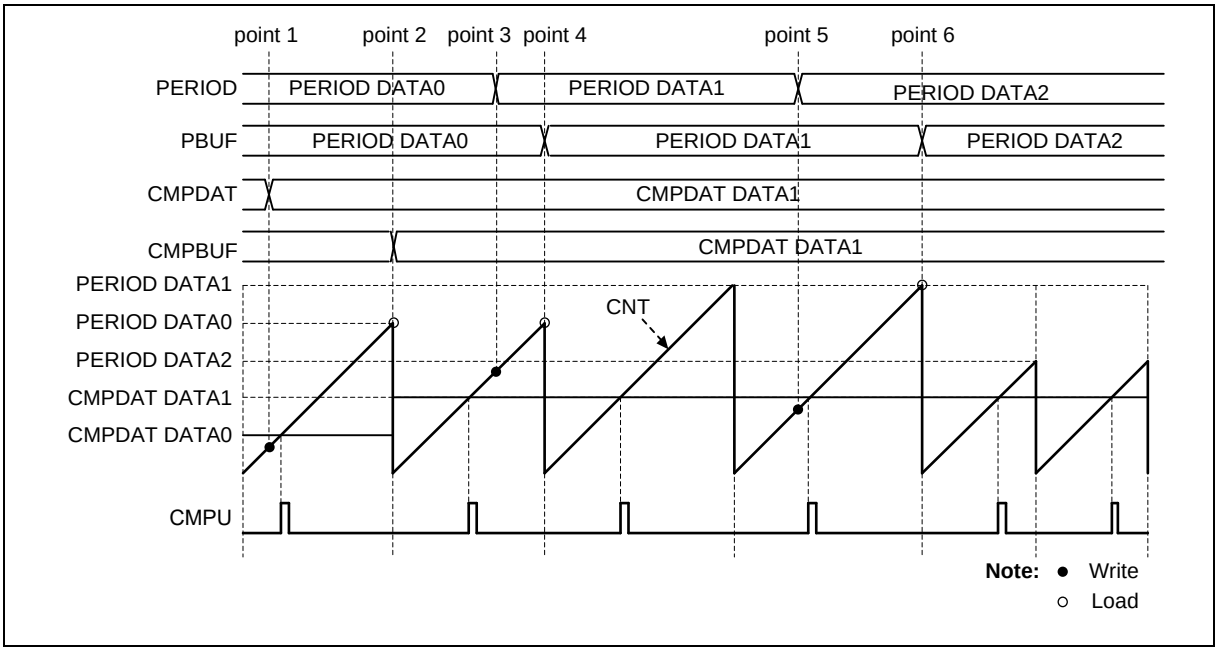


Figure 6.13-11 Period Loading Mode with Up-Counter Type

6.13.5.8 Immediately Loading Mode

If the IMMLDENn (BPWM_CTL0[21:16]) bit which corresponds to BPWM channel n is set to 1, software will load a value to buffer from PERIOD and CMPDAT immediately while software updates PERIOD or CMPDAT. If the update PERIOD value is less than current counter value, counter will count to 0xFFFF, when counter count to 0xFFFF and prescale count to 0, the flag CNTMAXF0(BPWM_STATUS[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 6.13-12 shows an example and its steps sequence is described below.

1. Software writes CMPDAT DATA1 and hardware immediately loading CMPDAT DATA1 to CMPBUF at point 1.
2. Software writes PERIOD DATA1 which is greater than current counter value at point 2; counter will continue counting until equal to PERIOD DATA1 to finish a period loading.
3. Software writes PERIOD DATA2 which is less than the current counter value at point 3; counter will continue counting to its maximum value 0xFFFF and count wraparound from 0 to PERIOD DATA2 to finish this period loading.

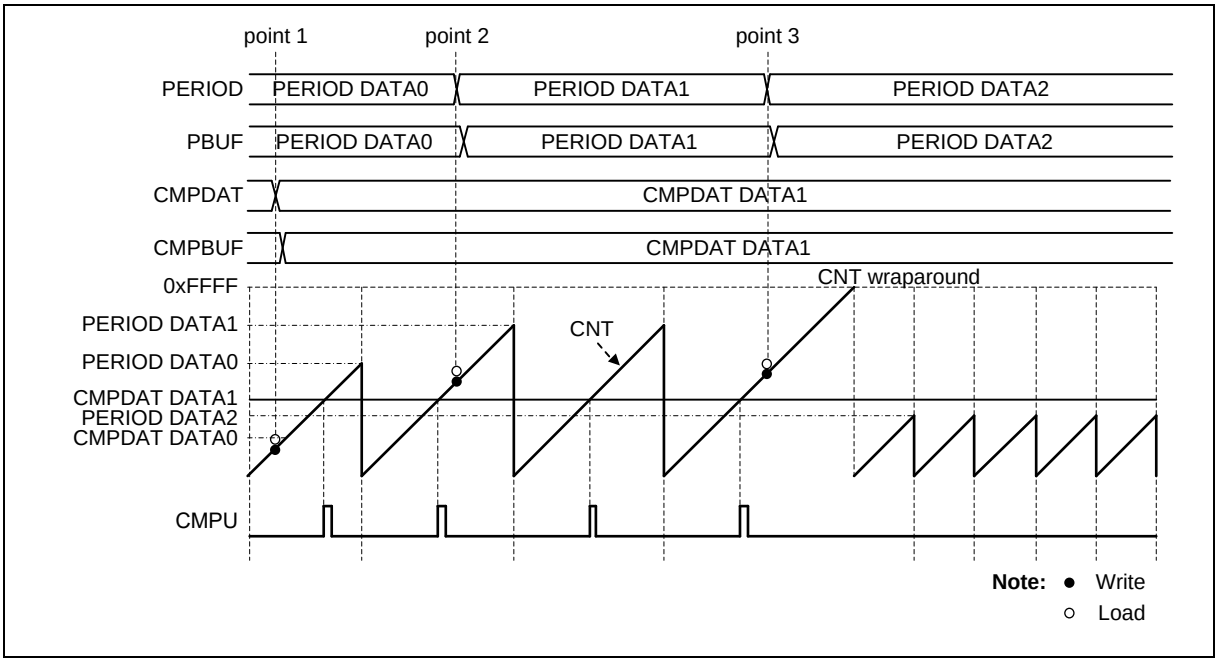


Figure 6.13-12 Immediately Loading Mode with Up-Counter Type

6.13.5.9 Center Loading Mode

If the CTRLDn (BPWM_CTL0[5:0]) bit which corresponds to BPWM channel n is set to 1 and in up-down counter type, CMPDAT will load to CMPBUFn in center of a period, that is, counter counts to PERIOD. PERIOD loading timing is the same as period loading mode. Figure 6.13-13 shows an example and its steps sequence is described below.

1. Software writes CMPDAT DATA1 at point 1.
2. Hardware loads CMPDAT DATA1 to CMPBUF at center of 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.

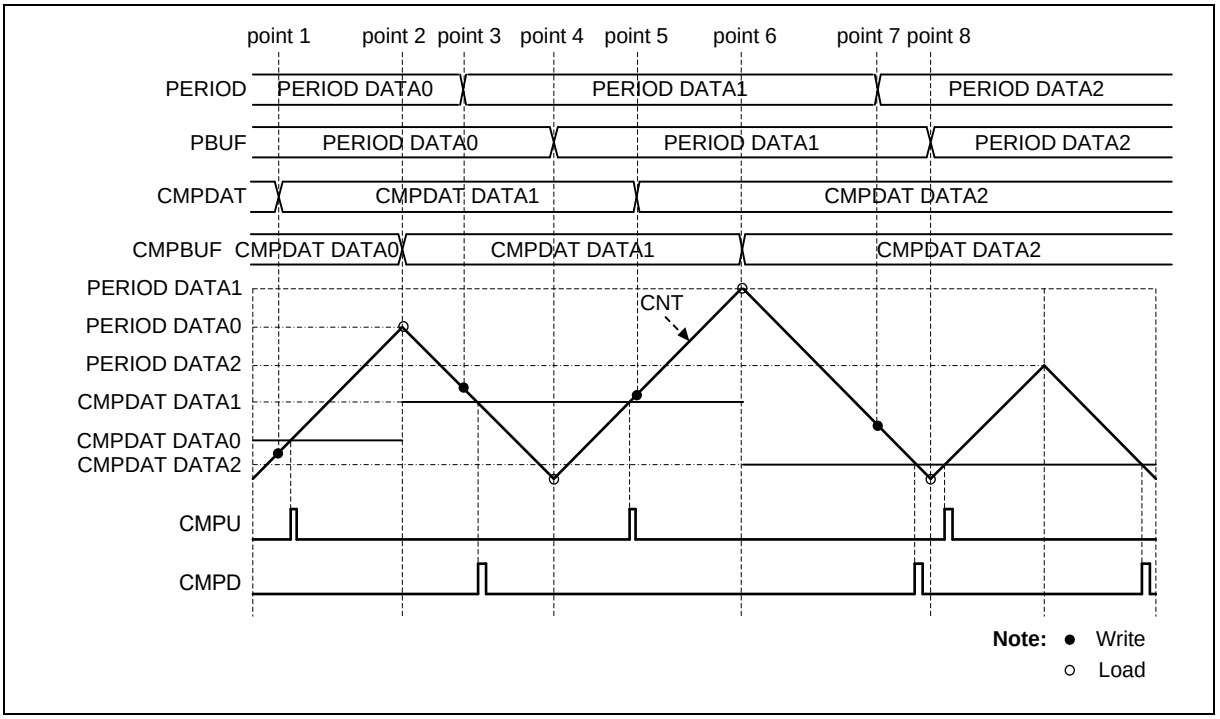


Figure 6.13-13 Center Loading Mode with Up-Down-Counter Type

6.13.5.10 BPWM Pulse Generator

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

Each event point can decide BPWM waveform to do nothing (X), set Low (L), set High (H) or toggle (T) by setting BPWM_WGCTL0 and BPWM_WGCTL1 registers. Using these points can easily generate asymmetric BPWM pulse or variant waveform as shown in Figure 6.13-14. In the figure, there is a comparator n to generate BPWM pulse, where n denotes channel number 0 to 5. CMPU denotes CNT is equal to CMPDAT when counting up, and CMPD denotes CNT is equal to CMPDAT when counting down.

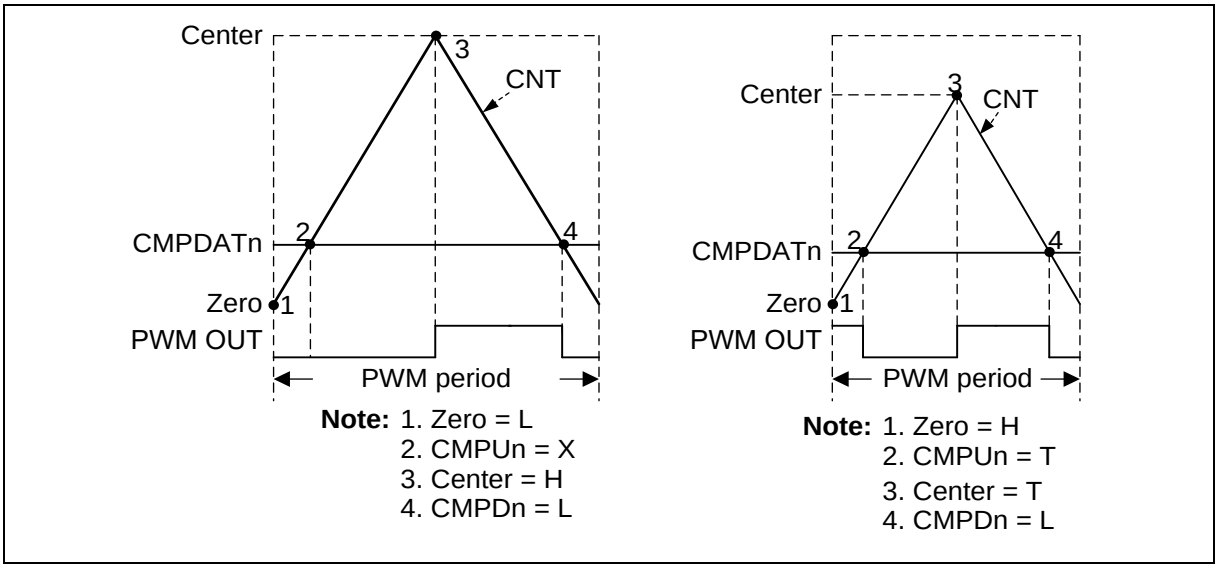


Figure 6.13-14 BPWM Pulse Generation (Left: Asymmetric Pulse, Right: Variety Pulse)

The generation events may be sometimes set to the same value, as the reason, events priority between different counter types are list below, up counter type (Table 6.13-3), down counter type (Table 6.13-4) and up-down counter type (Table 6.13-5). By using event priority, user can easily generate 0% to 100% duty pulse as shown in Figure 6.13-15.

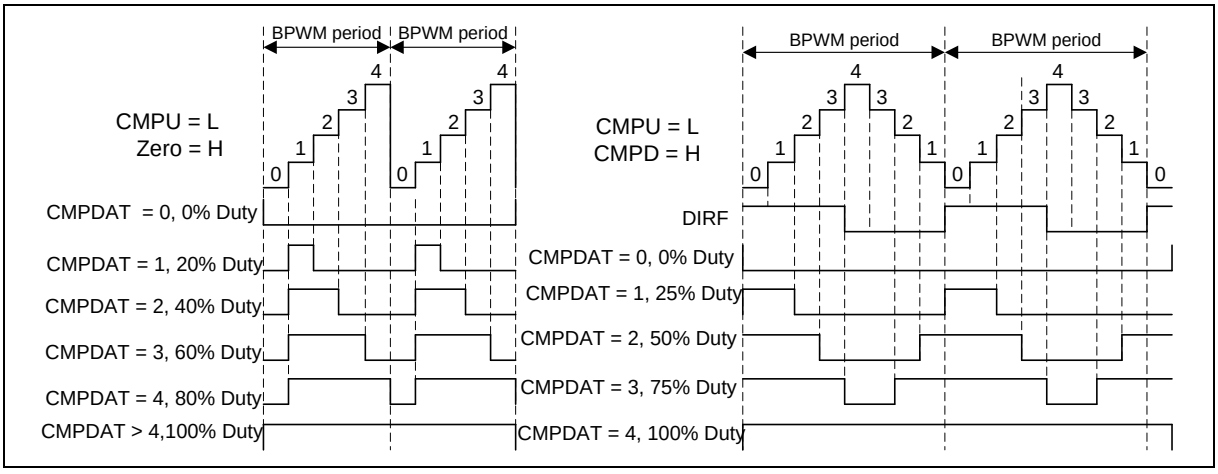


Figure 6.13-15 BPWM 0% to 100% Pulse Generation (Left: Up Counter Type, Right: Up-down Counter Type)

Priority	Up Event
1 (Highest)	Period event (CNT = PERIOD)
2	Compare up event(CNT = CMPUn)
3 (Lowest)	Zero event (CNT = 0)

Table 6.13-3 BPWM Pulse Generation Event Priority for Up-Counter

Priority	Down Event
1 (Highest)	Zero event (CNT = 0)

2	Compare down event (CNT = CMPDn)
3 (Lowest)	Period event (CNT = PERIOD)

Table 6.13-4 BPWM Pulse Generation Event Priority for Down-Counter

Priority	Up Event	Down Event
1 (Highest)	Compare up event (CNT = CMPUn)	Compare down event (CNT = CMPDn)
2 (Lowest)	Zero event (CNT = 0)	Period (center) event (CNT =PERIOD)
3 (Lowest)	Compare down event (CNT = CMPDn)	Compare up event (CNT = CMPUn)

Table 6.13-5 BPWM Pulse Generation Event Priority for Up-Down-Counter

6.13.5.11 Synchronous function

To start BPWM and PWM counters in the same time, user have to set the BPWM Synchronous Start Control Register (BPWM_SSCTL[0]) to enable the channel counters which are planned to start counting together, and select the SSRC(BPWM_SSCTL[9:8]) to choose the Synchronous Start source, followed by setting the BPWM Synchronous Start Trigger Register CNTSEN (BPWM_SSTRG[0]).

6.13.5.12 BPWM Output Control

After BPWM pulse generation, there are three steps to control the output of BPWM channels. There are Mask, Pin Polarity and Output Enable three steps as shown in Figure 6.13-16.

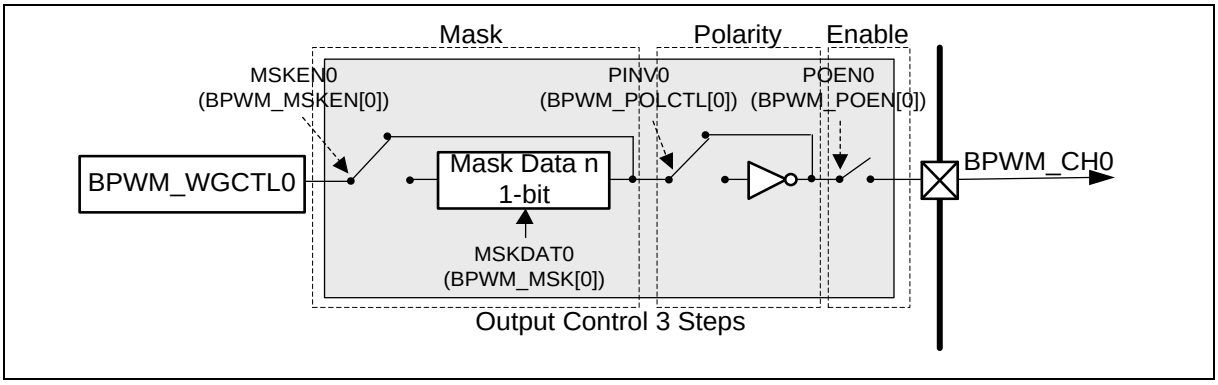


Figure 6.13-16 BPWM_CH0 Output Control 3 Steps

6.13.5.13 BPWM Mask Output Function

Each of the BPWM output channels can be manually overridden by using the appropriate bits in the BPWM Mask Enable Control Register (BPWM_MSKEN) and BPWM Masked Data Register (BPWM_MSK) to drive the BPWM channel outputs to specified logic states independent of the duty cycle comparison units. The BPWM mask bits are useful when controlling various types of Electrically Commutated Motor (ECM) like a BLDC motor. The BPWM_MSKEN register contains six bits, MSKENn(BPWM_MSKEN[5:0]) determine which BPWM channel output will be overridden, MSKENn(BPWM_MSKEN[5:0]) bits are active-high. The BPWM_MSK register contains six bits, MSKDATn(BPWM_MSK[5:0]) determine the state of the BPWM channel output when the channel is masked via the MSKDAT bits. Figure 6.13-17 shows an example of how BPWM mask control can be used for the override feature.

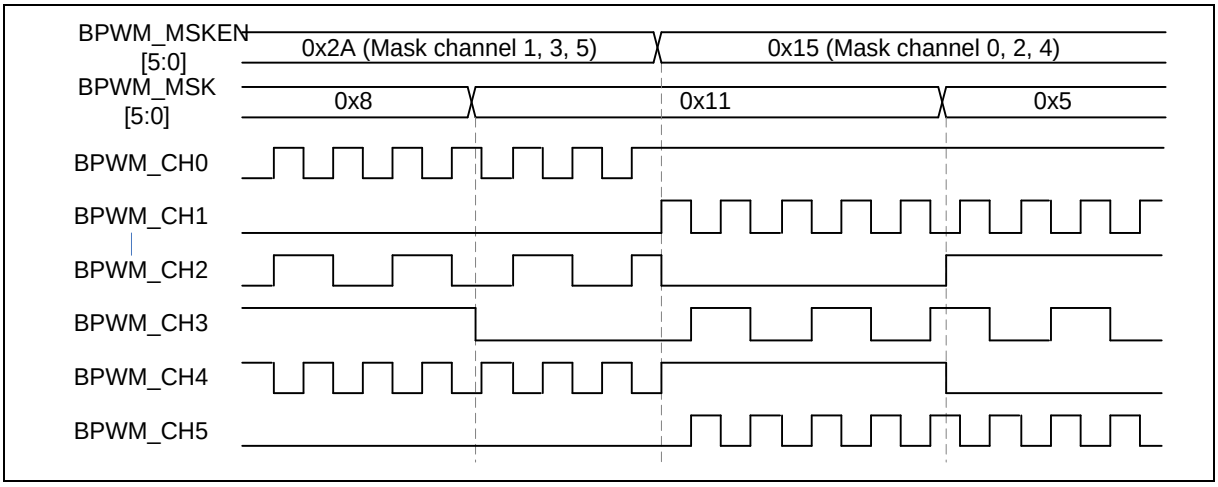
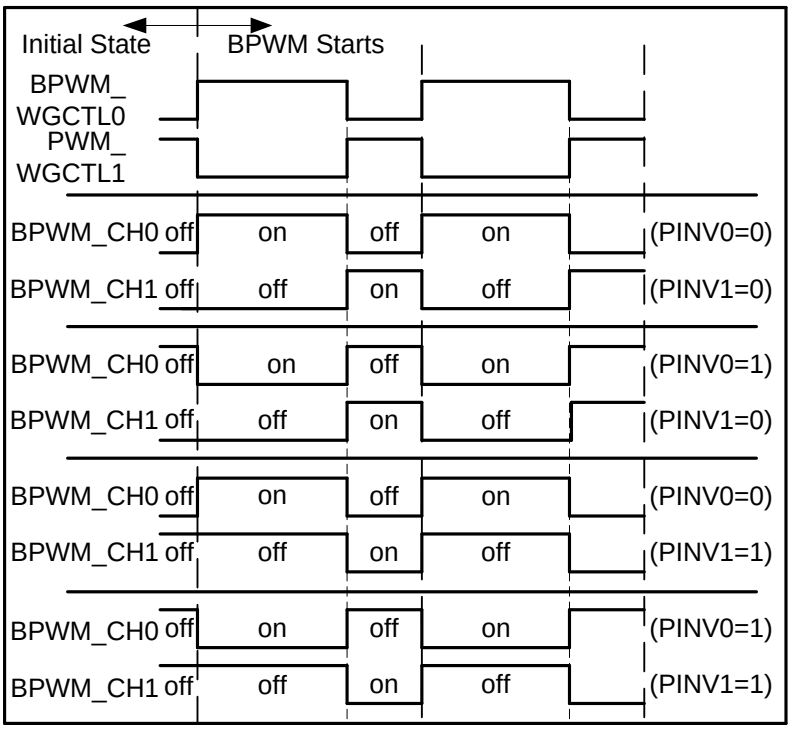


Figure 6.13-17 Mask Control Waveform Illustration

6.13.5.14 Polarity Control

Each BPWM port from BPWM_CH0 to BPWM_CH5 has an independent polarity control module to configure the polarity of the active state of BPWM output. By default, the BPWM output is active high. This implies the BPWM OFF state is low and ON state is high. This definition is variable through setting BPWM Negative Polarity Control Register (BPWM_POLCTL), for each individual BPWM channel. Figure 6.13-18 shows the initial state before BPWM starts with different polarity settings.



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

Figure 6.13-18 Initial State and Polarity Control

6.13.5.15 BPWM Interrupt Generator

There are two independent interrupts for each BPWM as shown in Figure 6.13-19.

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

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

Figure 6.13-19 demonstrates the architecture of the BPWM interrupts.

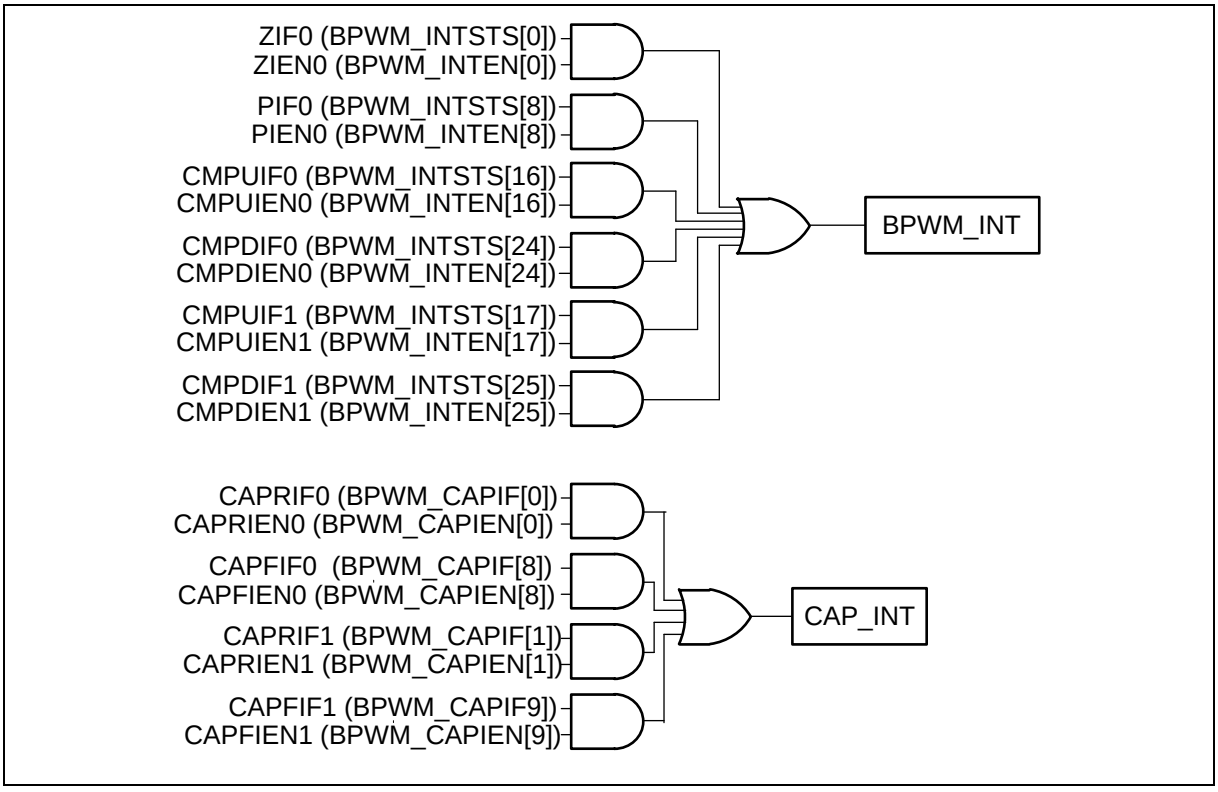


Figure 6.13-19 BPWM_CH0 and BPWM_CH1 Pair Interrupt Architecture Diagram

6.13.5.16 BPWM Trigger EADC Generator

BPWM can be one of the EADC conversion trigger source. Each BPWM pair channels share the same trigger source. Setting TRGSELn will select the trigger sources, where TRGSELn is TRGSEL0, TRGSEL1, …, and TRGSEL5, which are located in BPWM_EADCTS0[3:0], BPWM_EADCTS0[11:8], BPWM_EADCTS0[19:16], BPWM_EADCTS0[27:24], BPWM_EADCTS1[3:0] and BPWM_EADCTS1[11:8], respectively. Setting TRGENn will enable the trigger output to EADC, where TRGENn is TRGEN0, TRGEN1, …, TRGEN5, which are located in BPWM_EADCTS0[7],

BPWM_EADCTS0[15], BPWM_EADCTS0[23], BPWM_EADCTS0[31], BPWM_EADCTS1[7] and BPWM_EADCTS1[15], respectively. The number n (n = 0,1, ..,5) denotes BPWM channel number.

There are 7 BPWM events that an be selected as the trigger source for one pair of channels. Figure 6.13-20 is an example of BPWM_CH0 and BPWM_CH1. BPWM can trigger EADC to start conversion in different timings by setting PERIOD and CMPDAT. Figure 6.13-22 is the trigger EADC timing waveform in the up-down counter type.

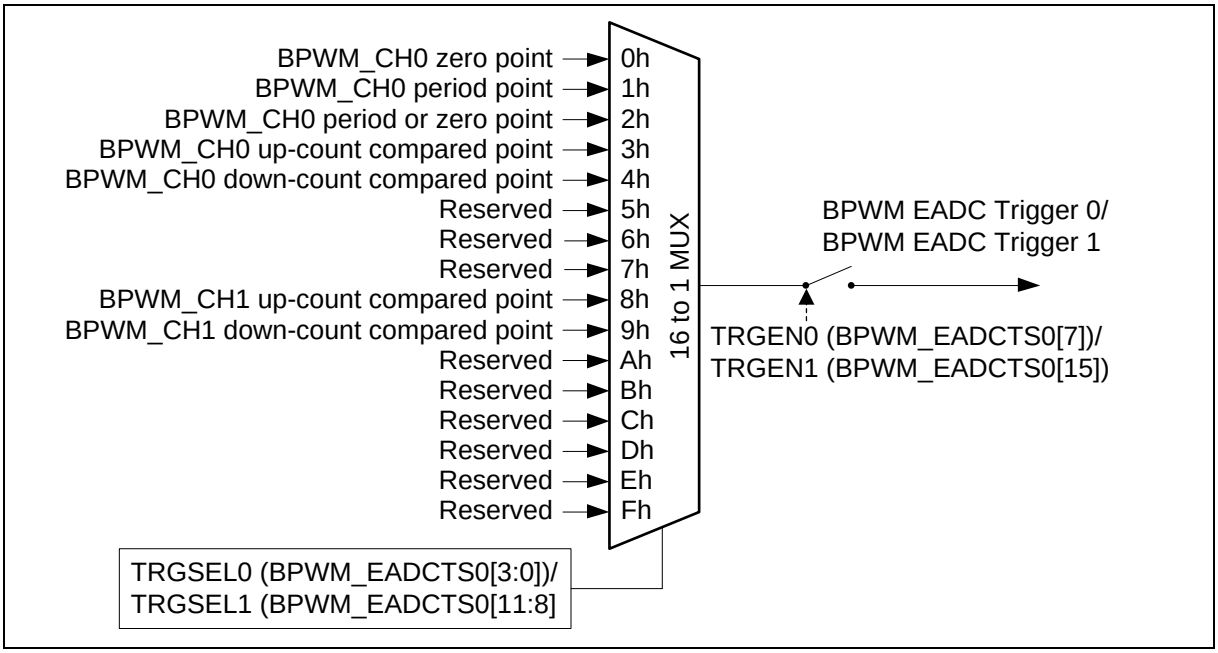


Figure 6.13-20 BPWM_CH0 and BPWM_CH1 Pair Trigger EADC Source Block Diagram

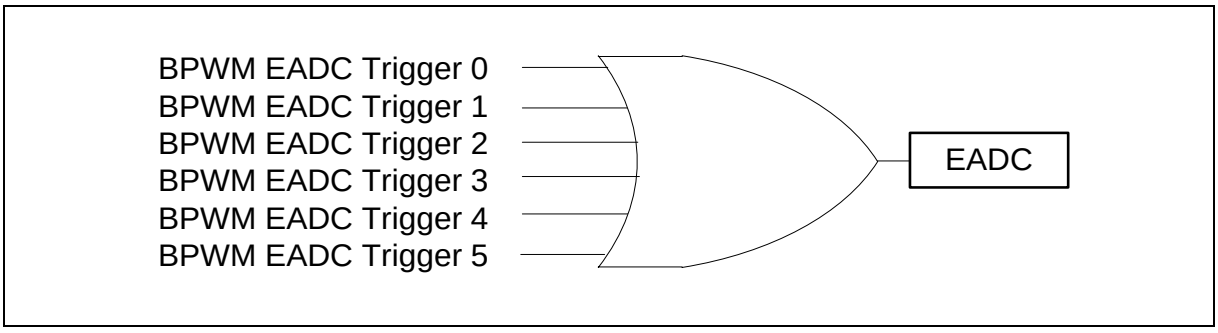


Figure 6.13-21 BPWM CH0~ CH5 Trigger EADC Block Diagram

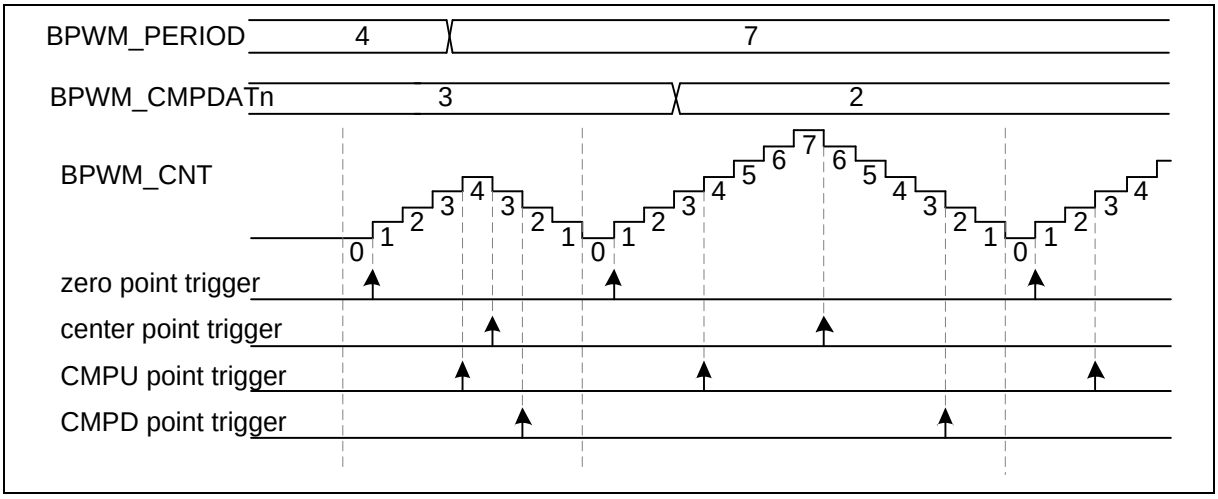


Figure 6.13-22 BPWM Trigger EADC in Up-Down Counter Type Timing Waveform

6.13.5.17 Capture Operation

The channels of the capture input and the BPWM output share the same pin and counter. The counter can operate in up or down counter type. The capture function will always latch the BPWM counter to the register RCAPDATn (BPWM_RCAPDATn[15:0]) or the register FCAPDATn (BPWM_FCAPDATn[15:0]) if the input channel has a rising transition or a falling transition, respectively. The capture function will also generate an interrupt CAP_INT (using BPWM_INT vector) if the rising or falling latch occurs and the corresponding channel n's rising or falling interrupt enable bits are set, where the CAPRIENn (BPWM_CAPIEN[5:0]) is for the rising edge and the CAPFIENn (BPWM_CAPIEN[13:8]) is for the falling edge. When rising or falling latch occurs, the corresponding BPWM counter may be reloaded with the value BPWM_PERIOD, depending on the setting of RCRLDENn or FCRLDENn (where RCRLDENn and FCRLDENn are located at BPWM_CAPCTL[21:16] and BPWM_CAPCTL[29:24], respectively). Note that the corresponding GPIO pins must be configured as the capture function by enable the CAPINENn (BPWM_CAPINEN[5:0]) for the corresponding capture channel n. Figure 6.13-23 is the capture block diagram of channel 0.

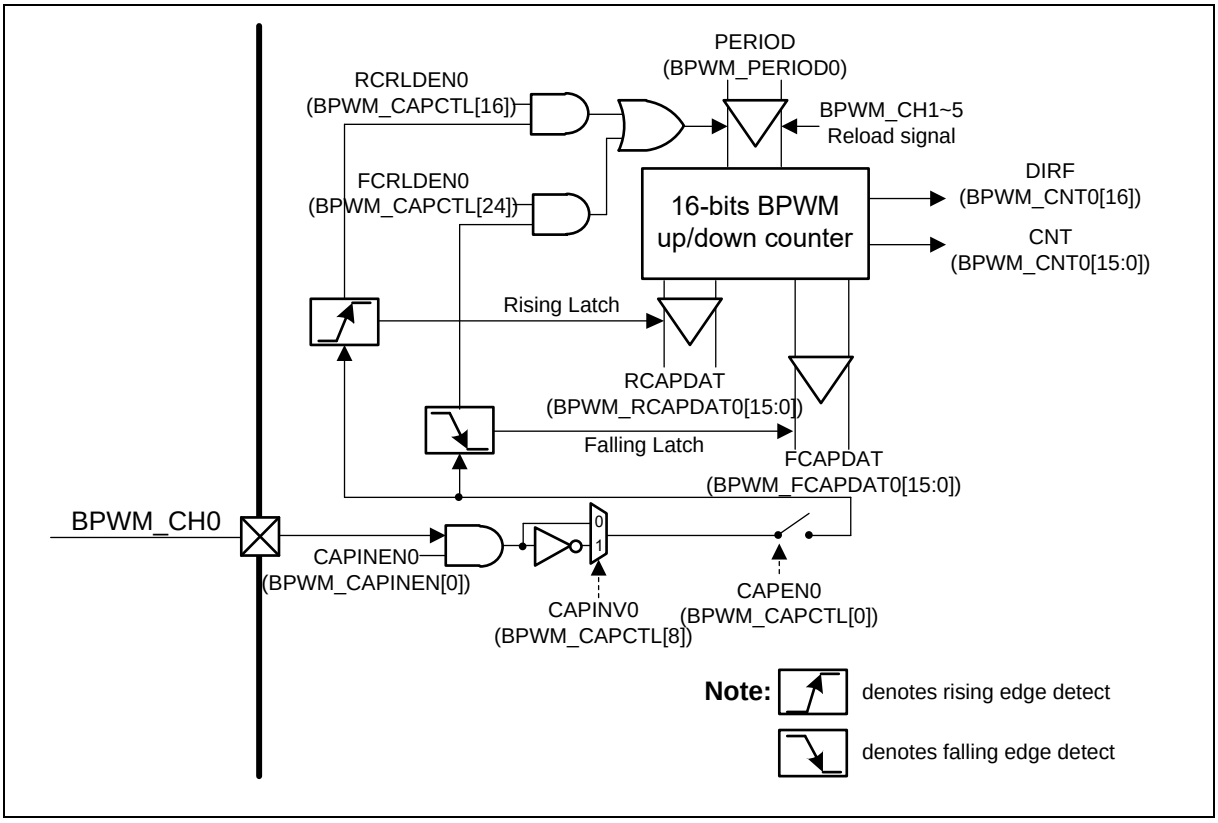


Figure 6.13-23 BPWM_CH0 Capture Block Diagram

Figure 6.13-24 illustrates the capture function timing. In this case, the capture counter is set as BPWM down counter type and the PERIOD is set to 8 so that the counter counts in the down direction, from 8 to 0. When detecting a falling edge at the capture input pin, the capture function latches counter value to the BPWM_FCAPDATn. When detecting the rising edge, it latches the counter value to the BPWM_RCAPDATn. In this timing diagram, when the falling edge is detected at the first time, the capture function will reload the counter value from the PERIOD setting because the FCRLDENn is enabled. But at the second time, the falling edge does not result in a reload because of the disabled FCRLDENn. In this example, the counter also reloads at the rising edge of the capture input because the RCRLDENn is enabled, too.

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

Figure 6.13-24 also illustrates the timing example for the interrupt and interrupt flag generation. When the rising edge at channel n is detected, the corresponding bit CAPRIFn (BPWM_CAPIF[5:0]) is set by hardware. Similarly, a falling edge detection at chnnel n causes the corresponding bit CAPFIFn (BPWM_CAPIF[13:8]) set by hardware. CAPRIFn (BPWM_CAPIF[5:0]) and CAPFIFn (BPWM_CAPIF[13:8]) can be cleared by software by writing '1'. If the CAPRIFn (BPWM_CAPIF[5:0]) is set and the CAPRIENn is enabled, the capture function generates an interrupt. If the CAPFIFn (BPWM_CAPIF[13:8]) is set and the CAPFIENn is enabled, the interrupt also happens.

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

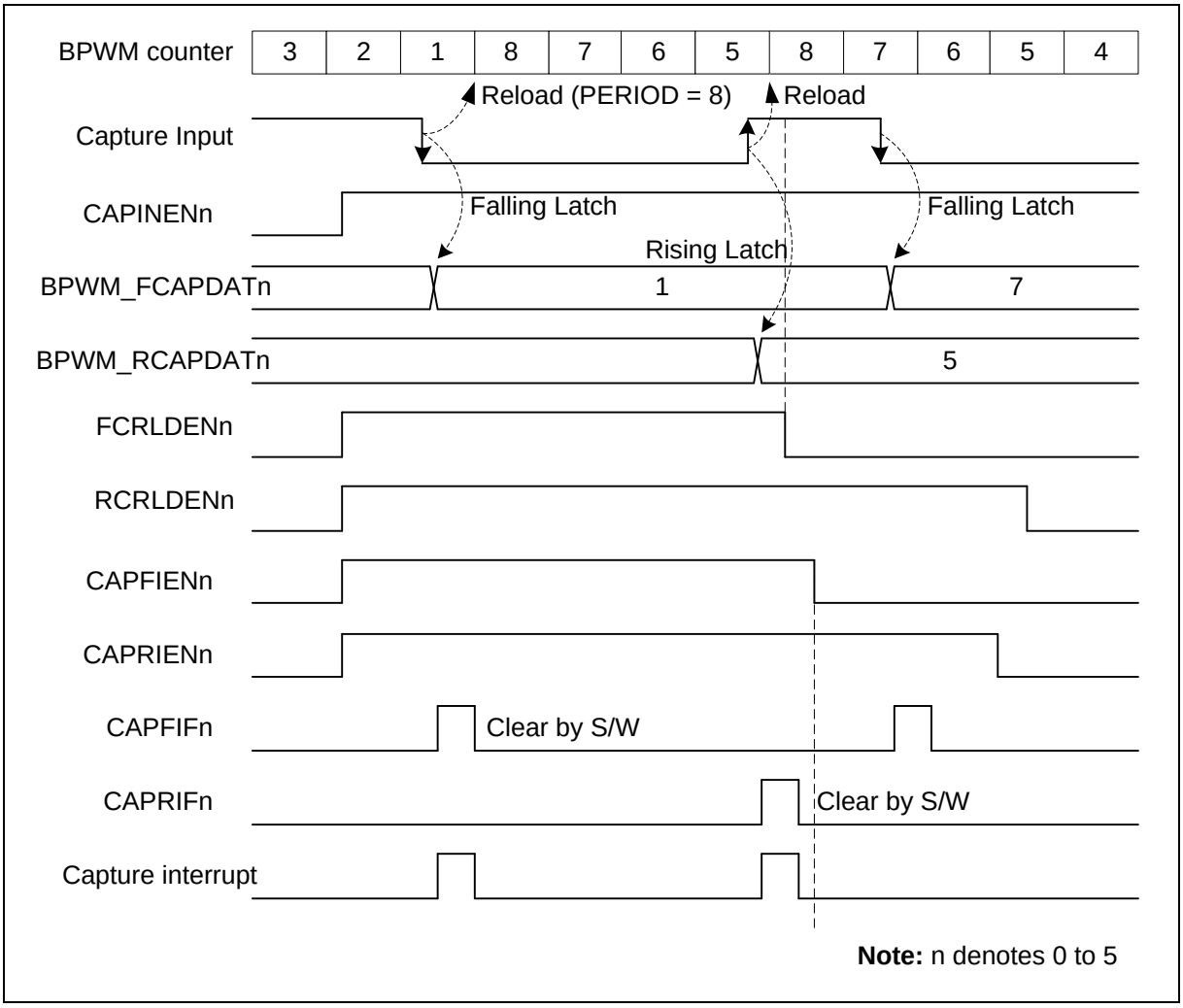


Figure 6.13-24 Capture Operation Waveform

The capture pulse width meeting the following conditions can be calculated according to the formula.

- The capture positive or negative pulse width is shorter than a counter period.
- The counter operates in down counter type.
- The counter can be reloaded by both falling and rising capture events through setting FCRLDENn and RCRLDENn bits of PWM_CAPCTL register to 1.

For the negative pulse case, the channel low pulse width is calculated as (BPWM_PERIOD + 1 - BPWM_RCAPDATn) BPWM counter time, where one BPWM counter time is (CLKPSC+1) * BPWMx_CLK clock time. In the case shown in Figure 6.13-24, low pulse width is 8+1-5 = 4 BPWM counter time.

For the positive pulse case, the channel high pulse width is calculated as (BPWM_PERIOD + 1 - BPWM_FCAPDATn) BPWM counter time, where one BPWM counter time is (CLKPSC+1) * BPWMx_CLK clock time. In the case shown in Figure 6.13-24, high pulse width is 8+1-7 = 2 BPWM counter time.

6.13.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
BPWM Base Address: **BPWM0_BA = 0x4005_A000** **BPWM1_BA = 0x4005_B000**				
BPWM_CTL0 **x=0, 1**	BPWMx_BA+0x00	R/W	BPWM Control Register 0	0x0000_0000
BPWM_CTL1 **x=0, 1**	BPWMx_BA+0x04	R/W	BPWM Control Register 1	0x0000_0000
BPWM_CLKSRC **x=0, 1**	BPWMx_BA+0x10	R/W	BPWM Clock Source Register	0x0000_0000
BPWM_CLKPSC **x=0, 1**	BPWMx_BA+0x14	R/W	BPWM Clock Prescale Register	0x0000_0000
BPWM_CNTEN **x=0, 1**	BPWMx_BA+0x20	R/W	BPWM Counter Enable Register	0x0000_0000
BPWM_CNTCLR **x=0, 1**	BPWMx_BA+0x24	R/W	BPWM Clear Counter Register	0x0000_0000
BPWM_PERIOD **x=0, 1**	BPWMx_BA+0x30	R/W	BPWM Period Register	0x0000_0000
BPWM_CMPDAT0 **x=0, 1**	BPWMx_BA+0x50	R/W	BPWM Comparator Register 0	0x0000_0000
BPWM_CMPDAT1 **x=0, 1**	BPWMx_BA+0x54	R/W	BPWM Comparator Register 1	0x0000_0000
BPWM_CMPDAT2 **x=0, 1**	BPWMx_BA+0x58	R/W	BPWM Comparator Register 2	0x0000_0000
BPWM_CMPDAT3 **x=0, 1**	BPWMx_BA+0x5C	R/W	BPWM Comparator Register 3	0x0000_0000
BPWM_CMPDAT4 **x=0, 1**	BPWMx_BA+0x60	R/W	BPWM Comparator Register 4	0x0000_0000
BPWM_CMPDAT5 **x=0, 1**	BPWMx_BA+0x64	R/W	BPWM Comparator Register 5	0x0000_0000
BPWM_CNT **x=0, 1**	BPWMx_BA+0x90	R	BPWM Counter Register	0x0000_0000
BPWM_WGCTL0 **x=0, 1**	BPWMx_BA+0xB0	R/W	BPWM Generation Register 0	0x0000_0000
BPWM_WGCTL1 **x=0, 1**	BPWMx_BA+0xB4	R/W	BPWM Generation Register 1	0x0000_0000
BPWM_MSKEN **x=0, 1**	BPWMx_BA+0xB8	R/W	BPWM Mask Enable Register	0x0000_0000
BPWM_MSK	BPWMx_BA+0xBC	R/W	BPWM Mask Data Register	0x0000_0000

x=0, 1				
BPWM_POLCTL **x=0, 1**	BPWMx_BA+0xD4	R/W	BPWM Pin Polar Inverse Register	0x0000_0000
BPWM_POEN **x=0, 1**	BPWMx_BA+0xD8	R/W	BPWM Output Enable Register	0x0000_0000
BPWM_INTEN **x=0, 1**	BPWMx_BA+0xE0	R/W	BPWM Interrupt Enable Register	0x0000_0000
BPWM_INTSTS **x=0, 1**	BPWMx_BA+0xE8	R/W	BPWM Interrupt Flag Register	0x0000_0000
BPWM_EADCTS0 **x=0, 1**	BPWMx_BA+0xF8	R/W	BPWM Trigger EADC0 Source Select Register 0	0x0000_0000
BPWM_EADCTS1 **x=0, 1**	BPWMx_BA+0xFC	R/W	BPWM Trigger EADC0 Source Select Register 1	0x0000_0000
BPWM_SSCTL **x=0, 1**	BPWMx_BA+0x110	R/W	BPWM Synchronous Start Control Register	0x0000_0000
BPWM_SSTRG **x=0, 1**	BPWMx_BA+0x114	W	BPWM Synchronous Start Trigger Register	0x0000_0000
BPWM_STATUS **x=0, 1**	BPWMx_BA+0x120	R/W	BPWM Status Register	0x0000_0000
BPWM_CAPINEN **x=0, 1**	BPWMx_BA+0x200	R/W	BPWM Capture Input Enable Register	0x0000_0000
BPWM_CAPCTL **x=0, 1**	BPWMx_BA+0x204	R/W	BPWM Capture Control Register	0x0000_0000
BPWM_CAPSTS **x=0, 1**	BPWMx_BA+0x208	R	BPWM Capture Status Register	0x0000_0000
BPWM_RCAPDAT0 **x=0, 1**	BPWMx_BA+0x20C	R	BPWM Rising Capture Data Register 0	0x0000_0000
BPWM_FCAPDAT0 **x=0, 1**	BPWMx_BA+0x210	R	BPWM Falling Capture Data Register 0	0x0000_0000
BPWM_RCAPDAT1 **x=0, 1**	BPWMx_BA+0x214	R	BPWM Rising Capture Data Register 1	0x0000_0000
BPWM_FCAPDAT1 **x=0, 1**	BPWMx_BA+0x218	R	BPWM Falling Capture Data Register 1	0x0000_0000
BPWM_RCAPDAT2 **x=0, 1**	BPWMx_BA+0x21C	R	BPWM Rising Capture Data Register 2	0x0000_0000
BPWM_FCAPDAT2 **x=0, 1**	BPWMx_BA+0x220	R	BPWM Falling Capture Data Register 2	0x0000_0000
BPWM_RCAPDAT3 **x=0, 1**	BPWMx_BA+0x224	R	BPWM Rising Capture Data Register 3	0x0000_0000
BPWM_FCAPDAT3 **x=0, 1**	BPWMx_BA+0x228	R	BPWM Falling Capture Data Register 3	0x0000_0000

BPWM_RCAPDAT4 **x=0, 1**	BPWMx_BA+0x22C	R	BPWM Rising Capture Data Register 4	0x0000_0000
BPWM_FCAPDAT4 **x=0, 1**	BPWMx_BA+0x230	R	BPWM Falling Capture Data Register 4	0x0000_0000
BPWM_RCAPDAT5 **x=0, 1**	BPWMx_BA+0x234	R	BPWM Rising Capture Data Register 5	0x0000_0000
BPWM_FCAPDAT5 **x=0, 1**	BPWMx_BA+0x238	R	BPWM Falling Capture Data Register 5	0x0000_0000
BPWM_CAPIEN **x=0, 1**	BPWMx_BA+0x250	R/W	BPWM Capture Interrupt Enable Register	0x0000_0000
BPWM_CAPIF **x=0, 1**	BPWMx_BA+0x254	R/W	BPWM Capture Interrupt Flag Register	0x0000_0000
BPWM_PBUF **x=0, 1**	BPWMx_BA+0x304	R	BPWM PERIOD Buffer	0x0000_0000
BPWM_CMPBUF0 **x=0, 1**	BPWMx_BA+0x31C	R	BPWM CMPDAT 0 Buffer	0x0000_0000
BPWM_CMPBUF1 **x=0, 1**	BPWMx_BA+0x320	R	BPWM CMPDAT 1 Buffer	0x0000_0000
BPWM_CMPBUF2 **x=0, 1**	BPWMx_BA+0x324	R	BPWM CMPDAT 2 Buffer	0x0000_0000
BPWM_CMPBUF3 **x=0, 1**	BPWMx_BA+0x328	R	BPWM CMPDAT 3 Buffer	0x0000_0000
BPWM_CMPBUF4 **x=0, 1**	BPWMx_BA+0x32C	R	BPWM CMPDAT 4 Buffer	0x0000_0000
BPWM_CMPBUF5 **x=0, 1**	BPWMx_BA+0x330	R	BPWM CMPDAT 5 Buffer	0x0000_0000

6.13.7 Register Description

BPWM Control Register 0 (BPWM_CTL0)

Register	Offset	R/W	Description	Reset Value
BPWM_CTL0	BPWMx_BA+0x00	R/W	BPWM Control Register 0	0x0000_0000

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

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

BPWM Control Register 1 (BPWM_CTL1)

Register	Offset	R/W	Description	Reset Value
BPWM_CTL1	BPWMx_BA+0x04	R/W	BPWM Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**CNTTYPE0**	

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

BPWM Clock Source Register (BPWM_CLKSRC)

Register	Offset	R/W	Description	Reset Value
BPWM_CLKSRC	BPWMx_BA+0x10	R/W	BPWM Clock Source Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**ECLKSRC0**		

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**ECLKSRC0**	**BPWM_CH01 External Clock Source Select** 000 = BPWMx_CLK, x denotes 0 or 1. 001 = TIMER0 overflow. 010 = TIMER1 overflow. 011 = TIMER2 overflow. 100 = TIMER3 overflow. Others = Reserved.

BPWM Clock Prescale Register (BPWM_CLKPSC)

Register	Offset	R/W	Description	Reset Value
BPWM_CLKPSC	BPWMx_BA+0x14	R/W	BPWM Clock Prescale Register	0x0000_0000

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

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

BPWM Counter Enable Register (BPWM_CNTEN)

Register	Offset	R/W	Description	Reset Value
BPWM_CNTEN	BPWMx_BA+0x20	R/W	BPWM Counter Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**CNTEN0**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**CNTEN0**	**BPWM Counter 0 Enable Bit** 0 = BPWM Counter and clock prescaler stop running. 1 = BPWM Counter and clock prescaler start running.

BPWM Clear Counter Register (BPWM_CNTCLR)

Register	Offset	R/W	Description	Reset Value
BPWM_CNTCLR	BPWMx_BA+0x24	R/W	BPWM 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							**CNTCLR0**

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

BPWM Period Register (BPWM_PERIOD)

Register	Offset	R/W	Description	Reset Value
BPWM_PERIOD	BPWMx_BA+0x30	R/W	BPWM Period Register	0x0000_0000

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

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

BPWM Comparator Register 0~5 (BPWM_CMPDAT0~5)

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

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CMPDAT**	**BPWM Comparator Register** CMPDAT use to compare with CNT to generate BPWM waveform, interrupt and trigger EADC0. In independent mode, CMPDAT0~5 denote as 6 independent BPWM_CH0~5 compared point.

BPWM Counter Register (BPWM_CNT)

Register	Offset	R/W	Description	Reset Value
BPWM_CNT	BPWMx_BA+0x90	R	BPWM 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.
[16]	**DIRF**	**BPWM Direction Indicator Flag (Read Only)** 0 = Counter is Down count. 1 = Counter is UP count.
[15:0]	**CNT**	**BPWM Data Register (Read Only)** User can monitor CNT to know the current value in 16-bit period counter.

BPWM Generation Register 0 (BPWM_WGCTL0)

Register	Offset	R/W	Description	Reset Value
BPWM_WGCTL0	BPWMx_BA+0xB0	R/W	BPWM Generation Register 0	0x0000_0000

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

Bits	Description	
[31:28]	**Reserved**	Reserved.
[16+2n+1:16+2n] n=0,1..5	**PRDPCTLn**	**BPWM Period (Center) Point Control** Each bit n controls the corresponding BPWM channel n. 00 = Do nothing. 01 = BPWM period (center) point output Low. 10 = BPWM period (center) point output High. 11 = BPWM period (center) point output Toggle. BPWM can control output level when BPWM counter count to (PERIOD+1). **Note:** This bit is center point control when BPWM counter operating in up-down counter type.
[15:12]	**Reserved**	Reserved.
[2n+1:2n] n=0,1..5	**ZPCTLn**	**BPWM Zero Point Control** Each bit n controls the corresponding BPWM channel n. 00 = Do nothing. 01 = BPWM zero point output Low. 10 = BPWM zero point output High. 11 = BPWM zero point output Toggle. **Note:** BPWM can control output level when BPWM counter counts to 0.

BPWM Generation Register 1 (BPWM_WGCTL1)

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

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

Bits	Description	
[31:28]	**Reserved**	Reserved.
[16+2n+1:16+2n] n=0,1..5	**CMPDCTLn**	**BPWM Compare Down Point Control** Each bit n controls the corresponding BPWM channel n. 00 = Do nothing. 01 = BPWM compare down point output Low. 10 = BPWM compare down point output High. 11 = BPWM compare down point output Toggle. **Note:** BPWM can control output level when BPWM counter down counts to CMPDAT.
[15:12]	**Reserved**	Reserved.
[2n+1:2n] n=0,1..5	**CMPUCTLn**	**BPWM Compare Up Point Control** Each bit n controls the corresponding BPWM channel n. 00 = Do nothing. 01 = BPWM compare up point output Low. 10 = BPWM compare up point output High. 11 = BPWM compare up point output Toggle. **Note:** BPWM can control output level when BPWM counter up counts to CMPDAT.

BPWM Mask Enable Register (BPWM_MSKEN)

Register	Offset	R/W	Description	Reset Value
BPWM_MSKEN	BPWMx_BA+0xB8	R/W	BPWM Mask Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		MSKEN5	MSKEN4	MSKEN3	MSKEN2	MSKEN1	MSKEN0

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

BPWM Mask DATA Register (BPWM_MSK)

Register	Offset	R/W	Description	Reset Value
BPWM_MSK	BPWMx_BA+0xBC	R/W	BPWM Mask Data Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**MSKDAT5**	**MSKDAT4**	**MSKDAT3**	**MSKDAT2**	**MSKDAT1**	**MSKDAT0**

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

BPWM Pin Polar Inverse Control (BPWM_POLCTL)

Register	Offset	R/W	Description	Reset Value
BPWM_POLCTL	BPWMx_BA+0xD4	R/W	BPWM Pin Polar Inverse Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**PINV5**	**PINV4**	**PINV3**	**PINV2**	**PINV1**	**PINV0**

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

BPWM Output Enable Register (BPWM_POEN)

Register	Offset	R/W	Description	Reset Value
BPWM_POEN	BPWMx_BA+0xD8	R/W	BPWM Output Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**POEN5**	**POEN4**	**POEN3**	**POEN2**	**POEN1**	**POEN0**

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

BPWM Interrupt Enable Register (BPWM_INTEN)

Register	Offset	R/W	Description	Reset Value
BPWM_INTEN	BPWMx_BA+0xE0	R/W	BPWM Interrupt Enable Register	0x0000_0000

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

Bits	Description	
[31:30]	**Reserved**	Reserved.
[24+n] n=0,1..5	**CMPDIENn**	**BPWM Compare Down Count Interrupt Enable Bits** Each bit n controls the corresponding BPWM channel n. 0 = Compare down count interrupt Disabled. 1 = Compare down count interrupt Enabled.
[23:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**CMPUIENn**	**BPWM Compare Up Count Interrupt Enable Bits** Each bit n controls the corresponding BPWM channel n. 0 = Compare up count interrupt Disabled. 1 = Compare up count interrupt Enabled.
[15:9]	**Reserved**	Reserved.
[8]	**PIEN0**	**BPWM Period Point Interrupt 0 Enable Bit** 0 = Period point interrupt Disabled. 1 = Period point interrupt Enabled. **Note:** When up-down counter type period point means center point.
[7:1]	**Reserved**	Reserved.
[0]	**ZIEN0**	**BPWM Zero Point Interrupt 0 Enable Bit** 0 = Zero point interrupt Disabled. 1 = Zero point interrupt Enabled.

BPWM Interrupt Flag Register (BPWM_INTSTS)

Register	Offset	R/W	Description	Reset Value
BPWM_INTSTS	BPWMx_BA+0xE8	R/W	BPWM Interrupt Flag Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		CMPDIF5	CMPDIF4	CMPDIF3	CMPDIF2	CMPDIF1	CMPDIF0
23	22	21	20	19	18	17	16
Reserved		CMPUIF5	CMPUIF4	CMPUIF3	CMPUIF2	CMPUIF1	CMPUIF0
15	14	13	12	11	10	9	8
Reserved							PIF0
7	6	5	4	3	2	1	0
Reserved							ZIF0

Bits	Description	
[31:30]	**Reserved**	Reserved.
[24+n] n=0,1..5	**CMPDIFn**	**BPWM Compare Down Count Interrupt Flag** Each bit n controls the corresponding BPWM channel n. Flag is set by hardware when BPWM counter down count and reaches BPWM_CMPDATn, software can clear this bit by writing 1 to it. **Note:** If CMPDAT equal to PERIOD, this flag is not working in down counter type selection.
[23:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**CMPUIFn**	**BPWM Compare Up Count Interrupt Flag** Flag is set by hardware when BPWM counter up count and reaches BPWM_CMPDATn, software can clear this bit by writing 1 to it. Each bit n controls the corresponding BPWM channel n. **Note:** If CMPDAT equal to PERIOD, this flag is not working in up counter type selection.
[15:9]	**Reserved**	Reserved.
[8]	**PIF0**	**BPWM Period Point Interrupt Flag 0** This bit is set by hardware when BPWM_CH0 counter reaches BPWM_PERIOD0, software can write 1 to clear this bit to 0.
[7:1]	**Reserved**	Reserved.
[0]	**ZIF0**	**BPWM Zero Point Interrupt Flag 0** This bit is set by hardware when BPWM_CH0 counter reaches 0, software can write 1 to clear this bit to 0.

BPWM Trigger EADC0 Source Select Register 0 (BPWM_EADCTS0)

Register	Offset	R/W	Description	Reset Value
BPWM_EADCTS0	BPWMx_BA+0xF8	R/W	BPWM Trigger EADC0 Source Select Register 0	0x0000_0000

31	30	29	28	27	26	25	24
TRGEN3	Reserved			TRGSEL3			
23	22	21	20	19	18	17	16
TRGEN2	Reserved			TRGSEL2			
15	14	13	12	11	10	9	8
TRGEN1	Reserved			TRGSEL1			
7	6	5	4	3	2	1	0
TRGEN0	Reserved			TRGSEL0			

Bits	Description	
[31]	**TRGEN3**	BPWM_CH3 Trigger EADC0 Enable Bit
[30:28]	**Reserved**	Reserved.
[27:24]	**TRGSEL3**	**BPWM_CH3 Trigger EADC0 Source Select** 0000 = BPWM_CH2 zero point. 0001 = BPWM_CH2 period point. 0010 = BPWM_CH2 zero or period point. 0011 = BPWM_CH2 up-count CMPDAT point. 0100 = BPWM_CH2 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = BPWM_CH3 up-count CMPDAT point. 1001 = BPWM_CH3 down-count CMPDAT point. Others reserved.
[23]	**TRGEN2**	BPWM_CH2 Trigger EADC0 Enable Bit
[22:20]	**Reserved**	Reserved.
[19:16]	**TRGSEL2**	**BPWM_CH2 Trigger EADC0 Source Select** 0000 = BPWM_CH2 zero point. 0001 = BPWM_CH2 period point. 0010 = BPWM_CH2 zero or period point. 0011 = BPWM_CH2 up-count CMPDAT point. 0100 = BPWM_CH2 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = BPWM_CH3 up-count CMPDAT point.

		1001 = BPWM_CH3 down-count CMPDAT point. Others reserved
[15]	**TRGEN1**	BPWM_CH1 Trigger EADC0 Enable Bit
[14:12]	**Reserved**	Reserved.
[11:8]	**TRGSEL1**	**BPWM_CH1 Trigger EADC0 Source Select** 0000 = BPWM_CH0 zero point. 0001 = BPWM_CH0 period point. 0010 = BPWM_CH0 zero or period point. 0011 = BPWM_CH0 up-count CMPDAT point. 0100 = BPWM_CH0 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = BPWM_CH1 up-count CMPDAT point. 1001 = BPWM_CH1 down-count CMPDAT point. Others reserved
[7]	**TRGEN0**	BPWM_CH0 Trigger EADC0 Enable Bit
[6:4]	**Reserved**	Reserved.
[3:0]	**TRGSEL0**	**BPWM_CH0 Trigger EADC0 Source Select** 0000 = BPWM_CH0 zero point. 0001 = BPWM_CH0 period point. 0010 = BPWM_CH0 zero or period point. 0011 = BPWM_CH0 up-count CMPDAT point. 0100 = BPWM_CH0 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = BPWM_CH1 up-count CMPDAT point. 1001 = BPWM_CH1 down-count CMPDAT point. Others reserved

BPWM Trigger EADC0 Source Select Register 1 (BPWM_EADCTS1)

Register	Offset	R/W	Description	Reset Value
BPWM_EADCTS1	BPWMx_BA+0xFC	R/W	BPWM Trigger EADC0 Source Select Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TRGEN5	**Reserved**			**TRGSEL5**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TRGEN4	**Reserved**			**TRGSEL4**			

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**TRGEN5**	BPWM_CH5 Trigger EADC0 Enable Bit
[14:12]	**Reserved**	Reserved.
[11:8]	**TRGSEL5**	**BPWM_CH5 Trigger EADC0 Source Select** 0000 = BPWM_CH4 zero point. 0001 = BPWM_CH4 period point. 0010 = BPWM_CH4 zero or period point. 0011 = BPWM_CH4 up-count CMPDAT point. 0100 = BPWM_CH4 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = BPWM_CH5 up-count CMPDAT point. 1001 = BPWM_CH5 down-count CMPDAT point. Others reserved
[7]	**TRGEN4**	BPWM_CH4 Trigger EADC0 Enable Bit
[6:4]	**Reserved**	Reserved.
[3:0]	**TRGSEL4**	**BPWM_CH4 Trigger EADC0 Source Select** 0000 = BPWM_CH4 zero point. 0001 = BPWM_CH4 period point. 0010 = BPWM_CH4 zero or period point. 0011 = BPWM_CH4 up-count CMPDAT point. 0100 = BPWM_CH4 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved.

		1000 = BPWM_CH5 up-count CMPDAT point. 1001 = BPWM_CH5 down-count CMPDAT point. Others reserved

BPWM Synchronous Start Control Register (BPWM_SSCTL)

Register	Offset	R/W	Description	Reset Value
BPWM_SSCTL	BPWMx_BA+0x110	R/W	BPWM Synchronous Start Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						SSRC	
7	6	5	4	3	2	1	0
Reserved							SSEN0

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:8]	**SSRC**	**BPWM Synchronous Start Source Select** 00 = Synchronous start source come from PWM0. 01 = Synchronous start source come from PWM1. 10 = Synchronous start source come from BPWM0. 11 = Synchronous start source come from BPWM1.
[7:1]	**Reserved**	Reserved.
[0]	**SSEN0**	**BPWM Synchronous Start Function 0 Enable Bit** When synchronous start function is enabled, the BPWM_CH0 counter enable bit (CNTEN0) can be enabled by writing BPWM synchronous start trigger bit (CNTSEN). 0 = BPWM synchronous start function Disabled. 1 = BPWM synchronous start function Enabled.

BPWM Synchronous Start Trigger Register (BPWM_SSTRG)

Register	Offset	R/W	Description	Reset Value
BPWM_SSTRG	BPWMx_BA+0x114	W	BPWM Synchronous Start Trigger Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**CNTSEN**	**BPWM Counter Synchronous Start Enable Bit(Write Only)** BPMW counter synchronous enable function is used to make PWM or BPWM channels start counting at the same time. Writing this bit to 1 will also set the counter enable bit if correlated BPWM channel counter synchronous start function is enabled.

BPWM Status Register (BPWM_STATUS)

Register	Offset	R/W	Description	Reset Value
BPWM_STATUS	BPWMx_BA+0x120	R/W	BPWM Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved		EADCTRG5	EADCTRG4	EADCTRG3	EADCTRG2	EADCTRG1	EADCTRG0
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							CNTMAXF0

Bits	Description	
[31:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**EADCTRGn**	**EADC0 Start of Conversion Status** Each bit n controls the corresponding BPWM channel n. 0 = No EADC0 start of conversion trigger event has occurred. 1 = An EADC0 start of conversion trigger event has occurred. Software can write 1 to clear this bit.
[15:1]	**Reserved**	Reserved.
[0]	**CNTMAXF0**	**Time-base Counter 0 Equal to 0xFFFF Latched Flag** 0 = The time-base counter never reached its maximum value 0xFFFF. 1 = The time-base counter reached its maximum value. Software can write 1 to clear this bit.

BPWM Capture Input Enable Register (BPWM_CAPINEN)

Register	Offset	R/W	Description	Reset Value
BPWM_CAPINEN	BPWMx_BA+0x200	R/W	BPWM Capture Input Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved	Reserved	CAPINEN5	CAPINEN4	CAPINEN3	CAPINEN2	CAPINEN1	CAPINEN0

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**CAPINENn**	**Capture Input Enable Bits** Each bit n controls the corresponding BPWM channel n. 0 = BPWM Channel capture input path Disabled. The input of BPWM channel capture function is always regarded as 0. 1 = BPWM Channel capture input path Enabled. The input of BPWM channel capture function comes from correlative multifunction pin.

BPWM Capture Control Register (BPWM_CAPCTL)

Register	Offset	R/W	Description	Reset Value
BPWM_CAPCTL	BPWMx_BA+0x204	R/W	BPWM Capture Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		FCRLDEN5	FCRLDEN4	FCRLDEN3	FCRLDEN2	FCRLDEN1	FCRLDEN0
23	22	21	20	19	18	17	16
Reserved		RCRLDEN5	RCRLDEN4	RCRLDEN3	RCRLDEN2	RCRLDEN1	RCRLDEN0
15	14	13	12	11	10	9	8
Reserved		CAPINV5	CAPINV4	CAPINV3	CAPINV2	CAPINV1	CAPINV0
7	6	5	4	3	2	1	0
Reserved		CAPEN5	CAPEN4	CAPEN3	CAPEN2	CAPEN1	CAPEN0

Bits	Description	
[31:30]	**Reserved**	Reserved.
[24+n] n=0,1..5	**FCRLDENn**	**Falling Capture Reload Enable Bits** Each bit n controls the corresponding BPWM channel n. 0 = Falling capture reload counter Disabled. 1 = Falling capture reload counter Enabled.
[23:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**RCRLDENn**	**Rising Capture Reload Enable Bits** Each bit n controls the corresponding BPWM channel n. 0 = Rising capture reload counter Disabled. 1 = Rising capture reload counter Enabled.
[15:14]	**Reserved**	Reserved.
[8+n] n=0,1..5	**CAPINVn**	**Capture Inverter Enable Bits** Each bit n controls the corresponding BPWM channel n. 0 = Capture source inverter Disabled. 1 = Capture source inverter Enabled. Reverse the input signal from GPIO.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**CAPENn**	**Capture Function Enable Bits** Each bit n controls the corresponding BPWM channel n. 0 = Capture function Disabled. RCAPDAT/FCAPDAT register will not be updated. 1 = Capture function Enabled. Capture latched the BPWM counter value when detected rising or falling edge of input signal and saved to RCAPDAT (Rising latch) and FCAPDAT (Falling latch).

BPWM Capture Status Register (BPWM_CAPSTS)

Register	Offset	R/W	Description	Reset Value
BPWM_CAPSTS	BPWMx_BA+0x208	R	BPWM Capture Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		CFIFOV5	CFIFOV4	CFIFOV3	CFIFOV2	CFIFOV1	CFIFOV0
7	6	5	4	3	2	1	0
Reserved		CRIFOV5	CRIFOV4	CRIFOV3	CRIFOV2	CRIFOV1	CRIFOV0

Bits	Description	
[31:14]	**Reserved**	Reserved.
[8+n] n=0,1..5	**CFIFOVn**	**Capture Falling Interrupt Flag Overrun Status (Read Only)** This flag indicates if falling latch happened when the corresponding CAPFIF is 1. Each bit n controls the corresponding BPWM channel n. **Note:** This bit will be cleared automatically when user clears corresponding CAPFIF.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**CRIFOVn**	**Capture Rising Interrupt Flag Overrun Status (Read Only)** This flag indicates if rising latch happened when the corresponding CAPRIF is 1. Each bit n controls the corresponding BPWM channel n. **Note:** This bit will be cleared automatically when user clears corresponding CAPRIF.

BPWM Rising Capture Data Register 0~5 (BPWM_RCAPDAT 0~5)

Register	Offset	R/W	Description	Reset Value
BPWM_RCAPDAT0	BPWMx_BA+0x20C	R	BPWM Rising Capture Data Register 0	0x0000_0000
BPWM_RCAPDAT1	BPWMx_BA+0x214	R	BPWM Rising Capture Data Register 1	0x0000_0000
BPWM_RCAPDAT2	BPWMx_BA+0x21C	R	BPWM Rising Capture Data Register 2	0x0000_0000
BPWM_RCAPDAT3	BPWMx_BA+0x224	R	BPWM Rising Capture Data Register 3	0x0000_0000
BPWM_RCAPDAT4	BPWMx_BA+0x22C	R	BPWM Rising Capture Data Register 4	0x0000_0000
BPWM_RCAPDAT5	BPWMx_BA+0x234	R	BPWM Rising Capture Data Register 5	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**RCAPDAT**	**BPWM Rising Capture Data (Read Only)** When rising capture condition happened, the BPWM counter value will be saved in this register.

BPWM Falling Capture Data Register 0~5 (BPWM_FCAPDAT 0~5)

Register	Offset	R/W	Description	Reset Value
BPWM_FCAPDAT0	BPWMx_BA+0x210	R	BPWM Falling Capture Data Register 0	0x0000_0000
BPWM_FCAPDAT1	BPWMx_BA+0x218	R	BPWM Falling Capture Data Register 1	0x0000_0000
BPWM_FCAPDAT2	BPWMx_BA+0x220	R	BPWM Falling Capture Data Register 2	0x0000_0000
BPWM_FCAPDAT3	BPWMx_BA+0x228	R	BPWM Falling Capture Data Register 3	0x0000_0000
BPWM_FCAPDAT4	BPWMx_BA+0x230	R	BPWM Falling Capture Data Register 4	0x0000_0000
BPWM_FCAPDAT5	BPWMx_BA+0x238	R	BPWM Falling Capture Data Register 5	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**FCAPDAT**	**BPWM Falling Capture Data (Read Only)** When falling capture condition happened, the BPWM counter value will be saved in this register.

BPWM Capture Interrupt Enable Register (BPWM_CAPIEN)

Register	Offset	R/W	Description	Reset Value
BPWM_CAPIEN	BPWMx_BA+0x250	R/W	BPWM Capture Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**CAPFIEN5**	**CAPFIEN4**	**CAPFIEN3**	**CAPFIEN2**	**CAPFIEN1**	**CAPFIEN0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**CAPRIEN5**	**CAPRIEN4**	**CAPRIEN3**	**CAPRIEN2**	**CAPRIEN1**	**CAPRIEN0**

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13:8]	**CAPFIENn**	**BPWM Capture Falling Latch Interrupt Enable Bits** Each bit n controls the corresponding BPWM channel n. 0 = Capture falling edge latch interrupt Disabled. 1 = Capture falling edge latch interrupt Enabled.
[7:6]	**Reserved**	Reserved.
[5:0]	**CAPRIENn**	**BPWM Capture Rising Latch Interrupt Enable Bits** Each bit n controls the corresponding BPWM channel n. 0 = Capture rising edge latch interrupt Disabled. 1 = Capture rising edge latch interrupt Enabled.

BPWM Capture Interrupt Flag Register (BPWM_CAPIF)

Register	Offset	R/W	Description	Reset Value
BPWM_CAPIF	BPWMx_BA+0x254	R/W	BPWM Capture Interrupt Flag Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**CAPFIF5**	**CAPFIF4**	**CAPFIF3**	**CAPFIF2**	**CAPFIF1**	**CAPFIF0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**CAPRIF5**	**CAPRIF4**	**CAPRIF3**	**CAPRIF2**	**CAPRIF1**	**CAPRIF0**

Bits	Description	
[31:14]	**Reserved**	Reserved.
[8+n] n=0,1..5	**CAPFIFn**	**BPWM Capture Falling Latch Interrupt Flag** Each bit n controls the corresponding BPWM channel n. 0 = No capture falling latch condition happened. 1 = Capture falling latch condition happened, this flag will be set to high. **Note:** This bit is cleared by writing 1 to it.
[7:6]	**Reserved**	Reserved.
[n] n=0,1..5	**CAPRIFn**	**BPWM Capture Rising Latch Interrupt Flag** Each bit n controls the corresponding BPWM channel n. 0 = No capture rising latch condition happened. 1 = Capture rising latch condition happened, this flag will be set to high. **Note:** This bit is cleared by writing 1 to it.

BPWM Period Register Buffer (BPWM_PBUF)

Register	Offset	R/W	Description	Reset Value
BPWM_PBUF	BPWMx_BA+0x304	R	BPWM PERIOD Buffer	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PBUF**	**BPWM Period Buffer (Read Only)** Used as PERIOD active register.

BPWM Comparator Register Buffer 0~5 (BPWM_CMPBUF0~5)

Register	Offset	R/W	Description	Reset Value
BPWM_CMPBUF0	BPWMx_BA+0x31C	R	BPWM CMPDAT 0 Buffer	0x0000_0000
BPWM_CMPBUF1	BPWMx_BA+0x320	R	BPWM CMPDAT 1 Buffer	0x0000_0000
BPWM_CMPBUF2	BPWMx_BA+0x324	R	BPWM CMPDAT 2 Buffer	0x0000_0000
BPWM_CMPBUF3	BPWMx_BA+0x328	R	BPWM CMPDAT 3 Buffer	0x0000_0000
BPWM_CMPBUF4	BPWMx_BA+0x32C	R	BPWM CMPDAT 4 Buffer	0x0000_0000
BPWM_CMPBUF5	BPWMx_BA+0x330	R	BPWM CMPDAT 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**
CMPBUF							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMPBUF							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CMPBUF**	**BPWM Comparator Buffer (Read Only)** Used as CMP active register.

6.14 UART Interface Controller (UART)

6.14.1 Overview

The chip provides six channels of Universal Asynchronous Receiver/Transmitters (UART). The UART controller performs serial-to-parallel conversion on data received from the peripheral and parallel-to-serial conversion on data transmitted from the CPU. Each UART controller channel supports eleven types of interrupts. The UART controller supports flow control function. The UART controller also supports IrDA SIR, LIN, RS-485 and Single-wire function modes and auto-baud rate measuring function.

6.14.2 Features

- Full-duplex asynchronous communications
- Separates receive and transmit 16/16 bytes or 1/1 byte entry FIFO for data payloads
- Supports hardware auto-flow control
- Programmable receiver buffer trigger level
- Supports programmable baud rate generator for each channel individually
- Supports nCTS, incoming data, Received Data FIFO reached threshold and RS-485 Address Match (AAD mode) wake-up function
- Supports 8-bit receiver buffer time-out detection function
- Programmable transmitting data delay time between the last stop and the next START bit by setting DLY (UART_TOUT[15:8])
- Supports Auto-Baud Rate measurement and baud rate compensation function
  - 9600 bps for UART_CLK is selected LXT.
- Supports break error, frame error, parity error and receive/transmit buffer overflow detection function
- Fully programmable serial-interface characteristics
  - Programmable number of data bit, 5-, 6-, 7-, 8- bit character
  - Programmable PARITY bit, even, odd, no parity or stick PARITY bit generation and detection
  - Programmable STOP bit, 1, 1.5, or 2 STOP bit generation
- Supports IrDA SIR function mode
  - 3/16 bit duration for normal mode
- Supports LIN function mode (Only UART0/UART1 with LIN function)
  - LIN master/slave mode
  - Programmable break generation function for transmitter
  - Break detection function for receiver
- Supports RS-485 function mode
  - RS-485 9-bit mode
  - Hardware or software enables to program nRTS pin to control RS-485 transmission direction
- Supports PDMA transfer function
- Supports Single-wire function mode

- Supports baud rate compensation function

UART Feature	UART0/ UART1	UART2 ~ UART5
FIFO	16 Bytes	1 Bytes
Auto Flow Control (CTS/RTS)	√	√
IrDA	√	√
LIN	√	-
RS-485 Function Mode	√	√
nCTS Wake-up	√	√
Incoming Data Wake-up	√	√
Received Data 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	1, 1.5, 2 bit
Word Length	5, 6, 7, 8 bits	5, 6, 7, 8 bits
Even / Odd Parity	√	√
Stick Bit	√	√
Baud Rate Compensation	√	-

Table 6.14-1 M471V/M471K/M471C series UART Features

6.14.3 Block Diagram

The UART clock control and block diagram are shown in Figure 6.14-1 and Figure 6.14-2 respectively.

Note: The frequency of UARTx_CLK should not be greater than 30 times HCLK.

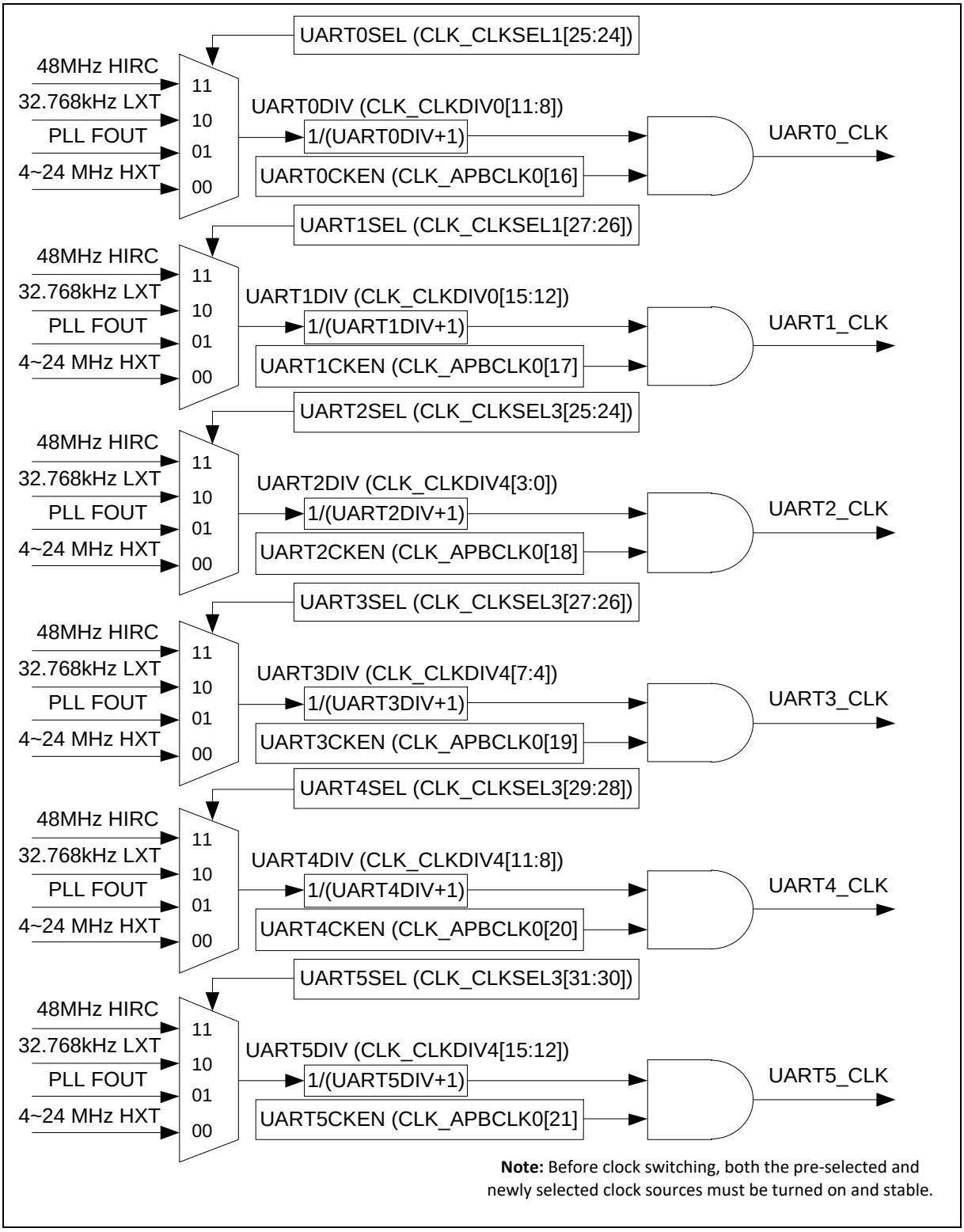


Figure 6.14-1 UART Clock Control Diagram

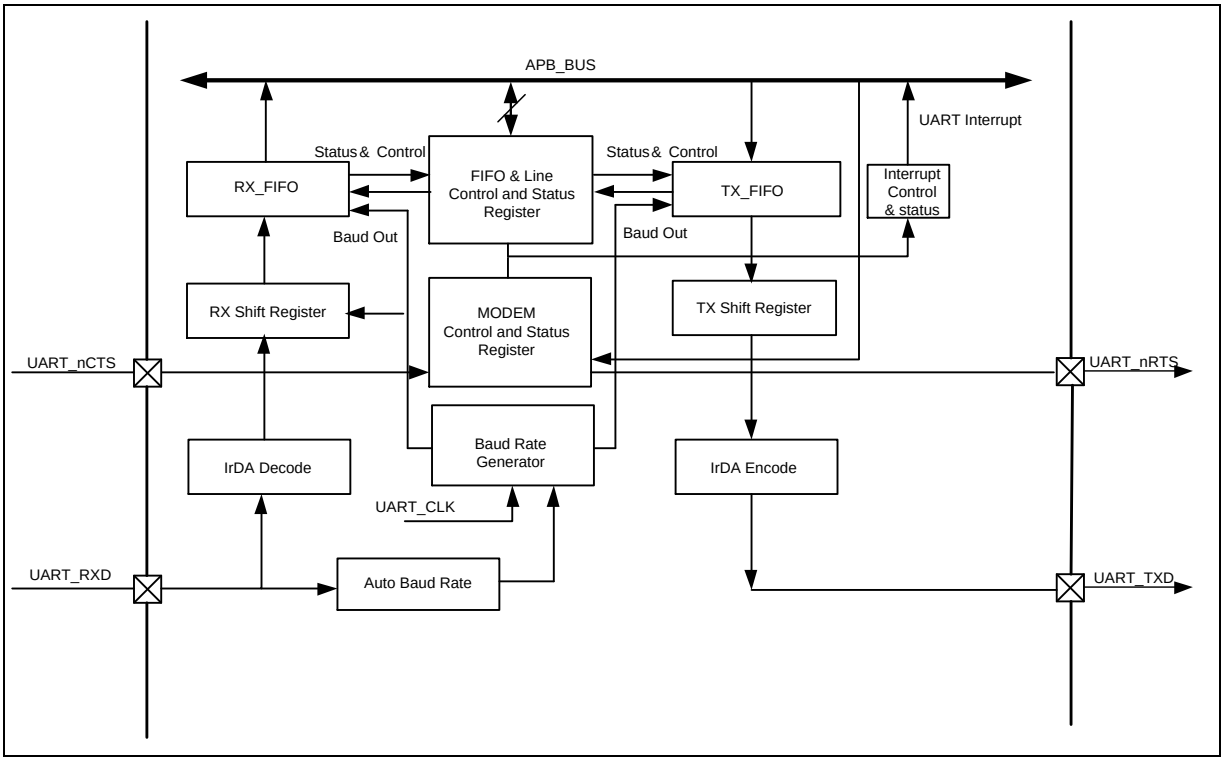


Figure 6.14-2 UART Block Diagram

Each block is described in detail as follows:

TX_FIFO

The transmitter is buffered with a 16 bytes or 1 byte FIFO to reduce the number of interrupts presented to the CPU.

RX_FIFO

The receiver is buffered with a 16 bytes or 1 byte FIFO (plus three error bits, BIF (UART_FIFOSTS[6]), FEF (UART_FIFOSTS[5]), PEF (UART_FIFOSTS[4])) to reduce the number of interrupts presented to the CPU.

TX Shift Register

This block is responsible for shifting out the transmitting data serially.

RX Shift Register

This block is responsible for shifting in the receiving data serially.

Modem Control and Status Register

This register controls the interface to the MODEM or data set (or a peripheral device emulating a MODEM).

Baud Rate Generator

Divide the external clock by the divisor to get the desired baud rate clock. Refer to baud rate equation.

IrDA Encode

This block is IrDA encoding control block.

IrDA Decode

This block is IrDA decoding control block.

FIFO & Line Control and Status Register

This field is register set that including the FIFO control register (UART_FIFO), FIFO status register (UART_FIFOSTS), and line control register (UART_LINE) for transmitter and receiver. The time-out register (UART_TOUT) identifies the condition of time-out interrupt.

Auto-Baud Rate Measurement

This block is responsible for auto-baud rate measurement.

Interrupt Control and Status Register

There are eleven types of interrupts. Interrupt enable register (UART_INTEN) enable or disable the responding interrupt and interrupt status register (UART_INTSTS) identifying the occurrence of the responding interrupt.

Interrupt	Description
RDAINT	Receive Data Available Interrupt.
THERINT	Transmit Holding Register Empty Interrupt.
TXENDINT	Transmitter Empty Interrupt.
RLSINT	Receive Line Status Interrupt (parity error or frame error or break error).
MODEMINT	MODEM Status Interrupt.
RXTOINT	Receiver Buffer Time-out Interrupt.
BUFERRINT	Buffer Error Interrupt.
LININT	LIN Bus Interrupt.
WKINT	Wake-up Interrupt.
ABRINT	Auto-Baud Rate Interrupt.
SWBEINT	Single-wire Bit Error Detect Interrupt.

Table 6.14-2 UART Interrupt

6.14.4 Basic Configuration

6.14.4.1 UART0 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_APBCLK0[16]).
- Reset UART0 controller in UART0RST (SYS_IPRST1[16]).
- UART1 Basic Configuration
- Clock Source Configuration
  - Select the source of UART1 peripheral clock on UART1SEL (CLK_CLKSEL1[27:26]).
  - Select the clock divider number of UART1 peripheral clock on UART1DIV (CLK_CLKDIV0[15:12]).
  - Enable UART1 peripheral clock in UART1CKEN (CLK_APBCLK0[17]).

- Reset UART1 controller in UART1RST (SYS_IPRST1[17]).
- UART2 Basic Configuration
- Clock Source Configuration
  - Select the source of UART2 peripheral clock on UART2SEL (CLK_CLKSEL3[25:24]).
  - Select clock divider number of UART2 peripheral clock on UART2DIV (CLK_CLKDIV4[3:0]).
  - Enable UART2 peripheral clock in UART2CKEN (CLK_APBCLK0[18]).
- Reset UART2 controller in UART2RST (SYS_IPRST1[18]).
- UART3 Basic Configuration
- Clock Source Configuration
  - Select the source of UART3 peripheral clock on UART3SEL (CLK_CLKSEL3[27:26]).
  - Select the clock divider number of UART3 peripheral clock on UART3DIV (CLK_CLKDIV4[7:4]).
  - Enable UART3 peripheral clock in UART3CKEN (CLK_APBCLK0[19]).
- Reset UART3 controller in UART3RST (SYS_IPRST1[19]).
- UART4 Basic Configuration
- Clock Source Configuration
  - Select the source of UART4 peripheral clock on UART4SEL (CLK_CLKSEL3[29:28]).
  - Select the clock divider number of UART4 peripheral clock on UART4DIV (CLK_CLKDIV4[11:8]).
  - Enable UART4 peripheral clock in UART4CKEN (CLK_APBCLK0[20]).
- Reset UART4 controller in UART4RST (SYS_IPRST1[20]).
- UART5 Basic Configuration
- Clock Source Configuration
  - Select the source of UART5 peripheral clock on UART5SEL (CLK_CLKSEL3[31:30]).
  - Select the clock divider number of UART5 peripheral clock on UART5DIV (CLK_CLKDIV4[15:12]).
  - Enable UART5 peripheral clock in UART5CKEN (CLK_APBCLK0[21]).
- Reset UART5 controller in UART5RST (SYS_IPRST1[21]).

UART Interface Controller Pin description is shown in Table 6.14-3:

Pin	Type	Description
UARTx_TXD	Output	UARTx transmit
UARTx_RXD	Input	UARTx receive
UARTx_nCTS	Input	UARTx modem clear to send
UARTx_nRTS	Output	UARTx modem request to send

Table 6.14-3 UART Interface Controller Pin

6.14.5 Functional Description

The UART controller supports five function modes including UART, IrDA, LIN, RS-485, and Single-

wire mode. User can select a function by setting the UART_FUNCSEL register. The five function modes will be described in following section.

6.14.5.1 UART Controller Baud Rate Generator

The UART controller includes a programmable baud rate generator capable of dividing clock input by divisors to produce the serial clock that transmitter and receiver need. Table 6.14-4 list the UART baud rate equations in the various conditions. Table 6.14-5 and Table 6.14-6 list the UART baud rate parameter and register setting example. In IrDA function mode, the baud rate generator must be set in mode 0. More detail register description is shown in UART_BAUD register. There are three setting mode. Mode 0 is set by UART_BAUD[29:28] with 00. Mode 1 is set by UART_BAUD[29:28] with 10. Mode 2 is set by UART_BAUD[29:28] with 11.

Mode	BAUDM1	BAUDM0	Baud Rate Equation
Mode 0	0	0	UART_CLK / [16 * (BRD+2)].
Mode 1	1	0	UART_CLK / [(EDIVM1+1) * (BRD+2)], EDIVM1 must >= 8.
Mode 2	1	1	UART_CLK / (BRD+2) If UART_CLK <= 3*HCLK, BRD must >= 8. If UART_CLK > 3*HCLK, BRD must >= 3*N – 1. N is the smallest integer larger than or equal to the ratio of UART_CLK /HCLK. For example, if 3*HCLK < UART_CLK =< 4*HCLK, BRD must >=11. if 4*HCLK < UART_CLK =< 5*HCLK, BRD must >=14. (If the UART_CLK is selected from LXT, BRD can be greater than or equal to 1)

Table 6.14-4 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 =12	BRD=24
230400	Not recommended	BRD =2, EDIVM1 =12	BRD =50
115200	Not recommended	BRD =6, EDIVM1 =12	BRD =102
57600	BRD =11	BRD =14, EDIVM1 =12	BRD =206
38400	BRD =18	BRD =22, EDIVM1 =12	BRD =311
19200	BRD =37	BRD =123, EDIVM1 =4	BRD =623
9600	BRD =76	BRD =123, EDIVM1 =9	BRD =1248
4800	BRD =154	BRD =248, EDIVM1 =9	BRD =2498

Table 6.14-5 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	0x2C00_0000	0x3000_0018
230400	Not recommended	0x2C00_0002	0x3000_0032
115200	Not recommended	0x2C00_0006	0x3000_0066
57600	0x0000_000B	0x2C00_000E	0x3000_00CE
38400	0x0000_0012	0x2C00_0016	0x3000_0137
19200	0x0000_0025	0x2400_007B	0x3000_026F
9600	0x0000_004C	0x2900_007B	0x3000_04E0
4800	0x0000_009A	0x2900_00F8	0x3000_09C2

Table 6.14-6 UART controller Baud Rate Register Setting Example Table

6.14.5.2 UART Controller Baud Rate Compensation

The UART controller supports baud rate compensation function. It is used to optimize the precision in each bit. The precision of the compensation is half of UART module clock because there is BRCOMPDEC (UART_BRCOMP[31]) to define the positive or negative compensation in each bit. If the BRCOMPDEC (UART_BRCOMP[31]) = 0, it is positive compensation for each bit, one more module clock will be append in the compensated bit. If the BRCOMPDEC (UART_BRCOMP[31]) = 1, it is negative compensation for each bit, decrease one module clock in the compensated bit.

There is 9-bits location, BRCOMP[8:0] (UART_BRCOMP[8:0]), can be configured by user to define the relative bit is compensated or not. BRCOMP[7:0] is used to define the compensation of DAT (UART_DAT[7:0]) and BRCOMP[8] is used to define PARITY (UART_DAT[8]).

Example:

1. UART's peripheral clock = 32.768 kHz and baud rate is 9600 bps

Baud rate is 9600 bps, UART peripheral clock is 32.768 kHz → 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 (UART_BRCOMP[31]) =0, so that the BRCOMP (UART_BRCOMP[8:0]) can be set as 9'b010100101 = 0xa5.

Bit	Name	Total INACCURACY	BRCOMP Compensated	Final Inaccuracy
0	START	-0.413	x	-0.413
1	UART_DAT[0]	-0.826(-0.413-0.413)	1	0.174
2	UART_DAT[1]	-0.239(0.174-0.413)	0	-0.239
3	UART_DAT[2]	-0.652(-0.239-0.413)	1	0.348
4	UART_DAT[3]	-0.065(0.348-0.413)	0	-0.065
5	UART_DAT[4]	-0.478(-0.065-0.413)	0	-0.478
6	UART_DAT[5]	-0.891(-0.478-0.413)	1	0.109
7	UART_DAT[6]	-0.304(0.109-0.413)	0	-0.304
8	UART_DAT[7]	-0.717(-0.304-0.413)	1	0.283
9	PARITY	-0.130(0.283-0.413)	0	-0.13

Table 6.14-7 Baud Rate Compensation Example Table 1

2. UART's peripheral clock = 32.768 kHz and baud rate is 4800

Baud rate is 4800, UART peripheral clock is 32.768 kHz → 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 (UART_BRCOMP[31]) =1, so that the BRCOMP (UART_BRCOMP[8:0]) can be set as 9'b010000010 = 0x82.

Bit	Name	Total INACCURACY	BRCOMP Compensated	Final Inaccuracy
0	START	0.173	x	0.173
1	UART_DAT[0]	0.346(0.173+0.173)	0	0.346
2	UART_DAT[1]	0.519(0.346+0.173)	1	-0.481
3	UART_DAT[2]	-0.308(-0.481+0.173)	0	-0.308
4	UART_DAT[3]	-0.135(-0.308+0.173)	0	-0.135
5	UART_DAT[4]	-0.038(-0.135+0.173)	0	0.038
6	UART_DAT[5]	0.211(0.038+0.173)	0	0.211
7	UART_DAT[6]	0.384(0.211+0.173)	0	0.384
8	UART_DAT[7]	0.557(0.384+0.173)	1	-0.443
9	PARITY	-0.270(-0.443+0.173)	0	-0.270

Table 6.14-8 Baud Rate Compensation Example Table 2

UART Controller Auto-Baud Rate Function Mode:

Auto-Baud Rate function can measure baud rate of receiving data from UART RX pin automatically. When the Auto-Baud Rate measurement is finished, the measuring baud rate is loaded to BRD (UART_BAUD[15:0]). Both of the BAUDM1 (UART_BAUD[29]) and BAUDM0 (UART_BAUD[28]) are set to 1 automatically. UART RX data from START bit to 1st rising edge time is set by $2^{ABRDBITS}$ bit time in Auto-Baud Rate function detection frame.

$2^{ABRDBITS}$ bit time from START bit to the 1st rising edge is calculated by setting ABRDBITS (UART_ALTCTL[20:19]). Set ABRDEN (UART_ALTCTL[18]) to enable auto-baud rate function. In the beginning when no data transfer, the UART RX is held high. Once the falling edge is detected, START bit is received. The auto-baud rate counter will be reset and then start counting. The auto-baud rate counter will be stop when the 1st rising edge is detected. Then, the 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. The Auto-Baud measurement is shown in Figure 6.14-3. 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. If ABRDIF (UART_FIFOSTS[1]) or ABRDTOIF (UART_FIFOSTS[2]) is set, the auto-baud rate flag ABRIF (UART_ALTCTL[17]) is generated. If ABRIEN (UART_INTEN[18]) is enabled, the auto-baud rate interrupt ABRINT (UART_INTSTS[31]) is generated when ABRIF (UART_ALTCTL[17]) is set.

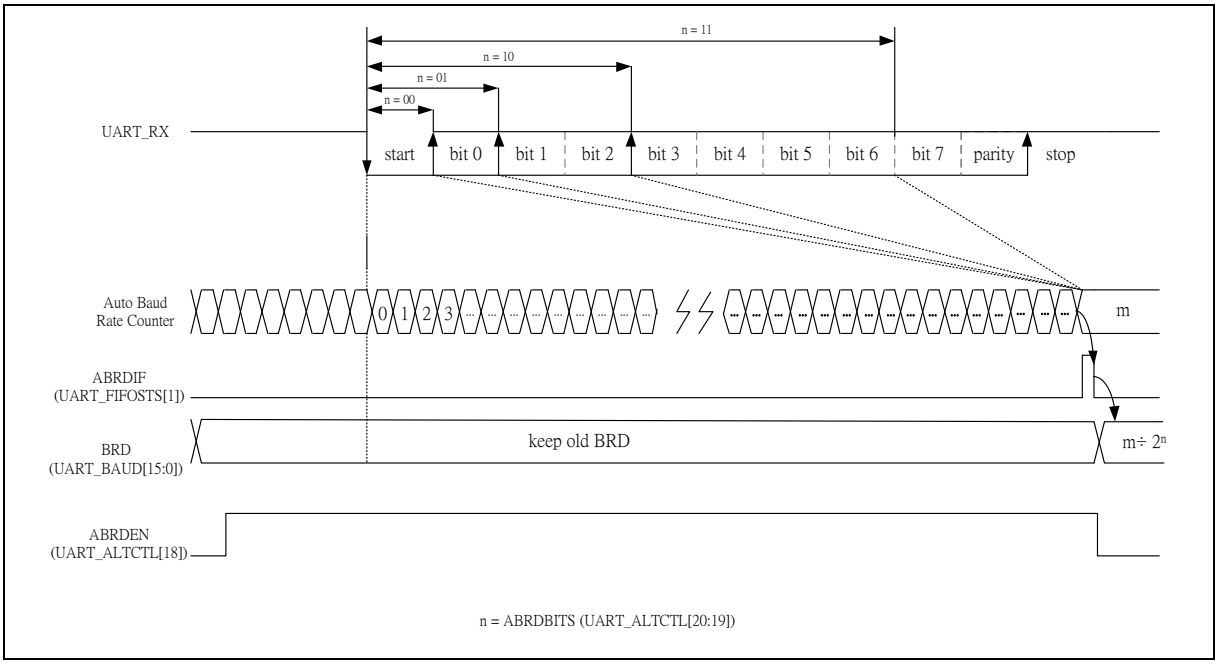


Figure 6.14-3 Auto-Baud Rate Measurement

6.14.5.3 Programming Sequence Example

1. Program ABRDBITS (UART_ALTCTL[20:19]) to determine 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.

6.14.5.4 UART Controller Transmit Delay Time Value

The UART controller programs DLY (UART_TOUT[15:8]) to control the transfer delay time between the last STOP bit and next START bit in transmission. The unit is baud. The operation is shown in Figure 6.14-4.

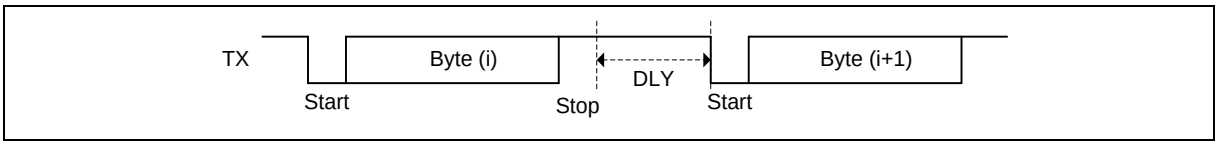
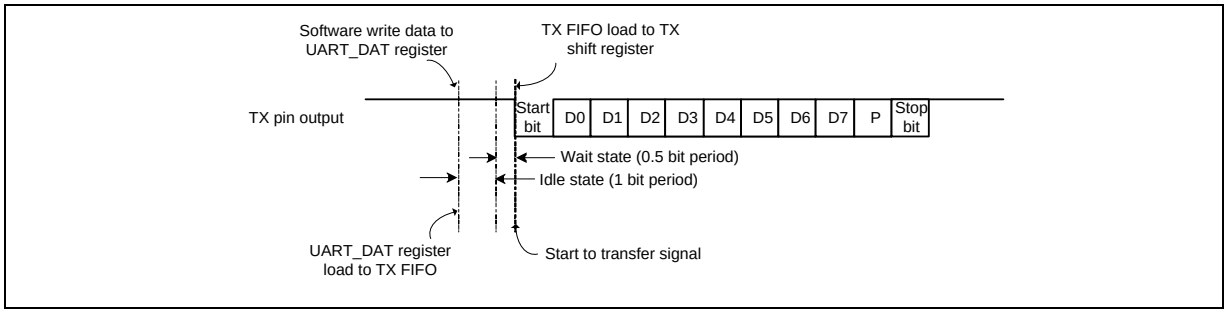


Figure 6.14-4 Transmit Delay Time Operation

In addition, if M471 write data byte to DAT (UART_DAT[7:0]) when TX FIFO is empty and the STOP bit of the last byte has been transmitted, the UART controller will have a transfer delay time 1.5 baud period to start UART_TXD signal.

Example:

1. UART's baud rate is 38400 bps. The UART will start to send out data after 1.5 * 1/38400 ≈ 40 us.



6.14.5.5 UART Controller FIFO Control and Status

The UART controller is built-in with a 16 bytes or 1 byte transmitter FIFO (TX_FIFO) and a 16 bytes or 1 byte receiver FIFO (RX_FIFO) that reduces the number of interrupts presented to the CPU. The CPU can read the status of the UART at any time during operation. The reported status information includes condition of the transfer operations being performed by the UART, as well as 3 error conditions (parity error, framing error, break interrupt) occur if receiving data has parity, frame or break error. UART, IrDA, LIN and RS-485 mode support FIFO control and status function.

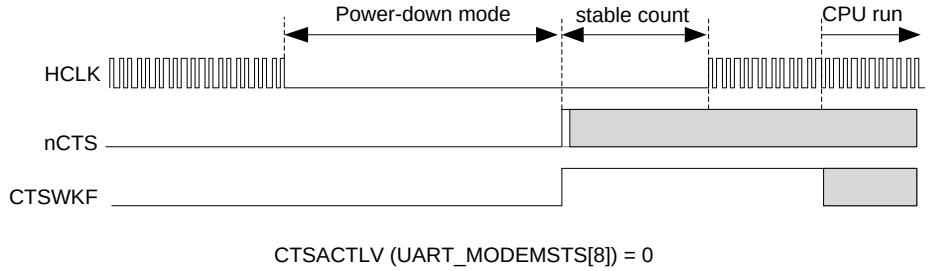
6.14.5.6 UART Controller Wake-up Function

The UART controller supports wake-up system function. The wake-up function includes nCTS pin, incoming data wake-up, Received Data FIFO reached threshold wake-up, RS-485 Address Match (AAD mode) wake-up and Received Data FIFO threshold time-out wake-up function. CTSWKF (UART_WKSTS[0]), DATWKF (UART_WKSTS[1]), RFRTWKF (UART_WKSTS[2]), RS485WKF (UART_WKSTS[3]) or TOUTWKF (UART_WKSTS[4]) cause the wake-up interrupt flag WKIF (UART_INTSTS[6]) generated. If the WKIEN (UART_INTEN[6]) is enabled, the wake-up interrupt flag WKIF (UART_INTSTS[6]) cause the wake-up interrupt WKINT (UART_INTSTS[14]) generated.

nCTS pin wake-up:

When the system is in Power-down mode and WKCTSEN (UART_WKCTL[0]) is set, the toggle of nCTS pin can wake up system. If the WKCTSEN (UART_WKCTL[0]) is enabled, the toggle of nCTS pin cause the nCTS wake-up flag CTSWKF (UART_WKSTS[0]) generated. The nCTS wake-up is shown in Figure 6.14-5 and Figure 6.14-6.

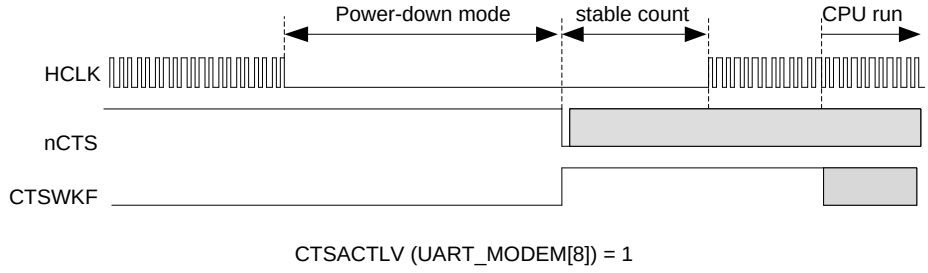
nCTS Wake-up Case 1 (nCTS transition from low to high)



Note: Stable count means HCLK source recovery stable count.

Figure 6.14-5 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 6.14-6 UART nCTS Wake-up Case2

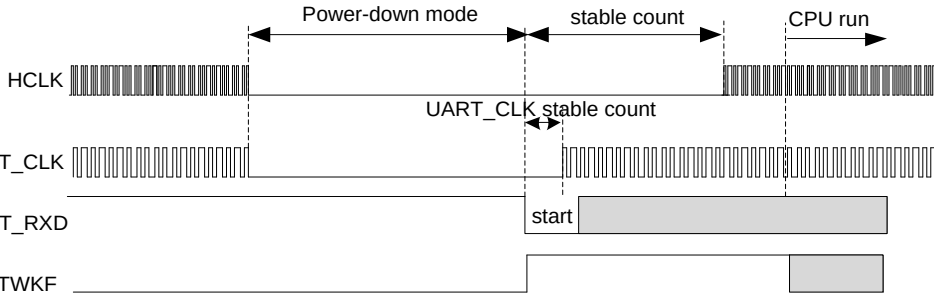
Incoming Data Wake-up

When system is in Power-down mode and the WKDATEN (UART_WKCTL[1]) is set, the toggle of incoming data (UART_RXD) pin can wake up the system. In order to receive the incoming data after the system wake-up, the STCOMP (UART_DWKCOMP[15:0]) shall be set. These bits field of STCOMP indicate how many clock cycle selected by UART_CLK do the UART controller can get the 1st bit (START bit) when the system is woken up from Power-down mode.

When incoming data wakes system up, the incoming data will be received and stored in FIFO. If the WKDATEN (UART_WKCTL[1]) is enabled, the toggle of incoming data (UART_RXD) pin cause the incoming data wake-up flag DATWKF (UART_WKSTS[1]) is generated. The incoming data wake-up is shown in Figure 6.14-7.

Note 1: The UART controller clock source should be selected as HIRC, and the compensation time for START bit that refer to the Datasheet for detailed information about wakeup time electrical characteristics. The STCOMP (UART_DWKCOMP[15:0]) = (wake-up stable time) * (HIRC frequency).

Note 2: The value of BRD (UART_BAUD[15:0]) should be greater than STCOMP (UART_DWKCOMP[15:0]).



Note 1: Stable count means HCLK source recovery stable count.

Note 2: UART_CLK stable count means UART clock source recovery stable count.

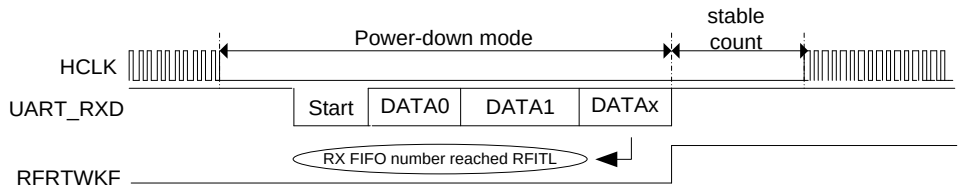
Figure 6.14-7 UART Data Wake-up

Received Data FIFO Reached Threshold Wake-up

The received data FIFO threshold reached wake-up function is enabled by setting WKRFRTEN (UART_WKCTL[2]). In Power-down mode, when the number of received data in RX FIFO reaches the threshold value RFITL (UART_FIFO[7:4]), it can wake up the system. If the WKRFRTEN (UART_WKCTL[2]) is enabled, the number of received data in RX FIFO reaches the threshold value RFITL (UART_FIFO[7:4]) cause the received data FIFO reached threshold wake-up flag RFRTWKF (UART_WKSTS[2]) is generated. The Received Data FIFO reached threshold wake-up is shown in Figure 6.14-8.

Note: The UART controller clock source should be selected as LXT in Power-down mode to receive

data.



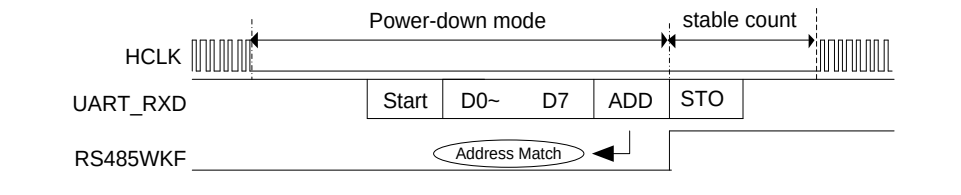
Note: Stable count means HCLK source recovery stable count.

Figure 6.14-8 UART Received Data FIFO reached threshold wake-up

RS-485 Address Match (AAD Mode) Wake-up

The RS-485 address match wake-up function is enabled by setting WKRFRTEN (UART_WKCTL[2]) and WKRS485EN (UART_WKCTL[3]). This function is used for RS-485 Auto Address Detection (AAD) mode in RS-485 function mode and ADDRDEN (UART_ALTCTL[15]) is set to 1. In Power-down mode, when an address byte is detected and matches the ADDRMV (UART_ALTCTL[31:24]) or the number of received data in RX FIFO reaches the threshold value RFITL (UART_FIFO[7:4]), it can wake up the system. If the WKRS485EN (UART_WKCTL[3]) is enabled, when an address byte is detected and matches the ADDRMV (UART_ALTCTL[31:24]), the RS485 address match (AAD mode) wake-up flag RS485WKF (UART_WKSTS[3]) is generated. The RS-485 Address Match (AAD mode) wake-up is shown in Figure 6.14-9.

Note: The UART controller clock source should be selected as LXT in Power-down mode to receive data.



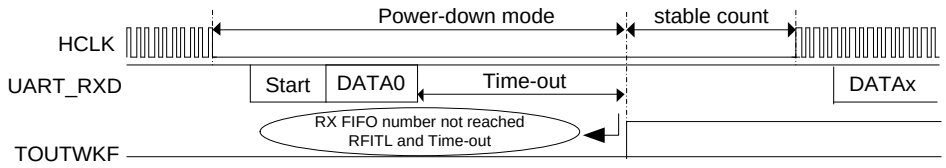
Note: Stable count means HCLK source recovery stable count.

Figure 6.14-9 UART RS-485 AAD Mode Address Match Wake-up

Received Data FIFO Threshold Time-out Wake-up

The received data FIFO threshold time-out wake-up function is enabled by setting WKRFRTEN (UART_WKCTL[2]) and WKTOUTEN (UART_WKCTL[4]). Setting TOCNTEN (UART_INTEN[11]) to enable receiver buffer time-out counter. In Power-down mode, when the number of received data in RX FIFO does not reach the threshold value RFITL (UART_FIFO[7:4]) and the time-out counter equals to the time-out value TOIC (UART_TOUT[7:0]), it can wake up the system. If the WKTOUTEN (UART_WKCTL[4]) is enabled, when the time-out counter equals to the time-out value TOIC (UART_TOUT[7:0]), the Received Data FIFO threshold time-out wake-up flag TOUTWKF (UART_WKSTS[4]) is generated. The Received Data FIFO threshold time-out wake-up is shown in Figure 6.14-10.

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 6.14-10 UART Received Data FIFO threshold time-out wake-up

6.14.5.7 UART Controller Interrupt and Status

Each UART controller supports eleven types of interrupts including:

- Receive Data Available Interrupt (RDAINT)
- Transmit Holding Register Empty Interrupt (THERINT)
- Transmitter Empty Interrupt (TXENDIF)
- Receive Line Status Interrupt (RLSINT)
  - Break Interrupt Flag (BIF)
  - Framing Error Flag (FEF)
  - Parity Error Flag (PEF)
  - RS-485 Address Byte Detect Flag (ADDRDETF)
- MODEM Status Interrupt (MODEMINT)
  - Detect nCTS State Change Flag (CTSDETF)
- Receiver Buffer Time-out Interrupt (RXTOINT)
- Buffer Error Interrupt (BUFERRINT)
  - TX Overflow Error Interrupt Flag (TXOVIF)
  - RX Overflow Error Interrupt Flag (RXOVIF)
- LIN Bus Interrupt (LININT)
  - LIN Break Detection Flag (BRKDETF)
  - Bit Error Detect Status Flag (BITEF)
  - LIN Slave ID Parity Error Flag (SLVIDPEF)
  - LIN Slave Header Error Flag (SLVHEF)
  - LIN Slave Header Detection Flag (SLVHDETF)
- Wake-up Interrupt (WKINT)
  - nCTS Wake-up Flag (CTSWKF)
  - Incoming Data Wake-up Flag (DATWKF)
  - Received Data FIFO Reached Threshold Wake-up Flag (RFRTWKF)
  - RS-485 Address Match (AAD mode) Wake-up Flag (RS485WKF)
  - Received Data FIFO Threshold Time-out Wake-up Flag (TOUTWKF)
- Auto-Baud Rate Interrupt (ABRINT)
  - Auto-baud Rate Detect Interrupt Flag (ABRDIF)

- Auto-baud Rate Detect Time-out Interrupt Flag (ABRDTOIF)

- Single-wire Bit Error Detect Interrupt (SWBEINT)

Table 6.14-9 describes the interrupt sources and flags. The interrupt is generated when the interrupt flag is generated and the interrupt enable bit is set. User must clear the interrupt flag after the interrupt is generated.

Interrupt Source	Interrupt Indicator	Interrupt Enable Bit	Interrupt Flag	Flag Caused By	Flag Cleared By
Receive Data Available Interrupt	RDAINT	RDAIEN	RDAIF	N/A	Read UART_DAT
Transmit Holding Register Empty Interrupt	THERINT	THREIEN	THREIF	N/A	Write UART_DAT
Transmitter Empty Interrupt	TXENDINT	TXENDIEN	TXENDIF	N/A	Write UART_DAT
Receive Line Status Interrupt	RLSINT	RLSIEN	RLSIF	RLSIF = BIF	Write '1' to BIF
				RLSIF = FEF	Write '1' to FEF
				RLSIF = PEF	Write '1' to PEF
				RLSIF = ADDRDETF	Write '1' to ADDRDETF
Modem Status Interrupt	MODEMINT	MODEMIEN	MODEMIF	MODEMIF = CTSDETF	Write '1' to CTSDETF
Receiver Buffer Time-out Interrupt	RXTOINT	RXTOIEN	RXTOIF	N/A	Read UART_DAT
Buffer Error Interrupt	BUFERRINT	BUFERRIEN	BUFERRIF	BUFERRIF = TXOVIF	Write '1' to TXOVIF
				BUFERRIF = RXOVIF	Write '1' to RXOVIF
LIN Bus Interrupt	LININT	LINIEN	LINIF	LINIF = BRKDETF	Write '1' to LINIF and Write '1' to BRKDETF
				LINIF = BITEF	Write '1' to LINIF and Write '1' to BITEF
				LINIF = SLVIDPEF	Write '1' to LINIF and Write '1' to SLVIDPEF
				LINIF = SLVHEF	Write '1' to LINIF and Write '1' to SLVHEF
				LINIF = SLVHDETF	Write '1' to LINIF and Write '1' to SLVHDETF
Wake-up Interrupt	WKINT	WKIEN	WKIF	WKIF = CTSWKF	Write '1' to CTSWKF

				WKIF = DATWKF	Write ‘1’ to DATWKF
				WKIF = RFRTWKF	Write ‘1’ to RFRTWKF
				WKIF = RS485WKF	Write ‘1’ to RS485WKF
				WKIF = TOUTWKF	Write ‘1’ to TOUTWKF
Auto-Baud Rate Interrupt	ABRINT	ABRIEN	ABRIF	ABRIF = ABRDIF	Write ‘1’ to ABRDIF
				ABRIF = ABRDTOIF	Write ‘1’ to ABRDTOIF
Single-wire Bit Error Detect Interrupt	SWBEINT	SWBEIEN	SWBEIF	N/A	Writing ‘1’ to SWBEIF

Table 6.14-9 UART Controller Interrupt Source and Flag List

6.14.5.8 UART Function Mode

The UART controller provides UART function (setting FUNCSEL (UART_FUNCSEL[2:0]) to ‘000’ to enable UART function mode). The UART baud rate is up to 10 Mbps.

The UART provides full-duplex and asynchronous communications. The transmitter and receiver contain 16 bytes FIFO for payloads. User can program receiver buffer trigger level and receiver buffer time-out detection for receiver. The transmitting data delay time between the last stop and the next START bit can be programed by setting DLY (UART_TOUT[15:8]) register. The UART supports hardware auto-flow control that provides programmable nRTS flow control trigger level. The number of data bytes in RX FIFO is equal to or greater than RTSTRGLV (UART_FIFO[19:16]), the nRTS is de-asserted.

UART Line Control Function

The UART controller supports fully programmable serial-interface characteristics by setting the UART_LINE register. User can program UART_LINE register for the word length, STOP bit and PARITY bit setting. Table 6.14-10 and Table 6.14-11 list the UART word, STOP bit length and the PARITY bit settings.

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

Table 6.14-10 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 6.14-11 UART Line Control of PARITY Bit Setting

UART Auto-Flow Control Function

The UART supports auto-flow control function that uses two signals, nCTS (clear-to-send) and nRTS (request-to-send), to control the flow of data transfer between the UART and external devices (e.g. Modem). When auto-flow is enabled, the UART is not allowed to receive data until the UART asserts nRTS to external device. When the number of bytes stored in the RX FIFO equals the value of RTSTRGLV (UART_FIFO[19:16]), the nRTS is de-asserted. The UART sends data out when UART detects nCTS is asserted from external device. If the valid asserted nCTS is not detected, the UART will not send data out. The auto flow control block diagram is shown in Figure 6.14-11.

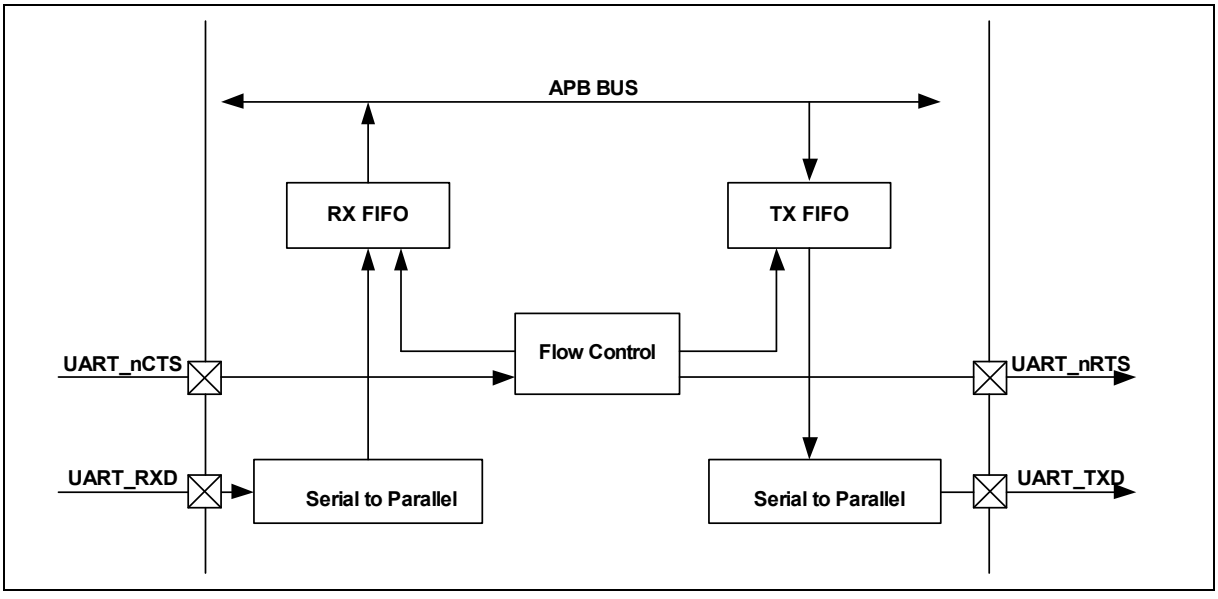


Figure 6.14-11 Auto-Flow Control Block Diagram

Figure 6.14-12 demonstrates the nCTS auto-flow control of UART function mode. User must set ATOCTSEN (UART_INTEN[13]) to enable nCTS auto-flow control function. The CTSACTLV

(UART_MODEMSTS[8]) can set nCTS pin input active state. The CTSDETF (UART_MODEMSTS[0]) is set when any state change of nCTS pin input has occurred, and then TX data will be automatically transmitted from TX FIFO.

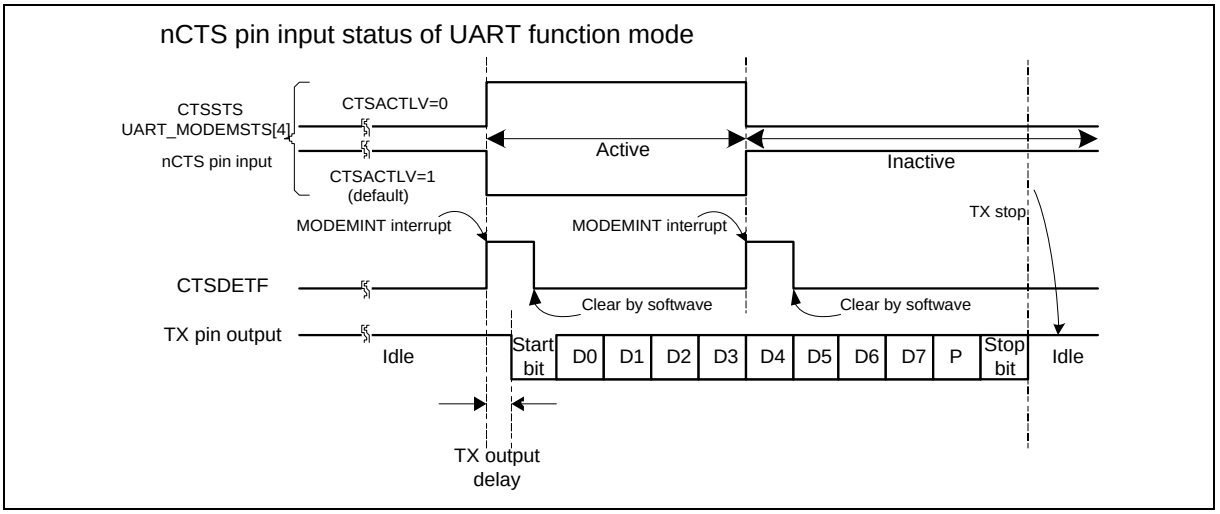


Figure 6.14-12 UART nCTS Auto-Flow Control Enabled

As shown in Figure 6.14-13, 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.

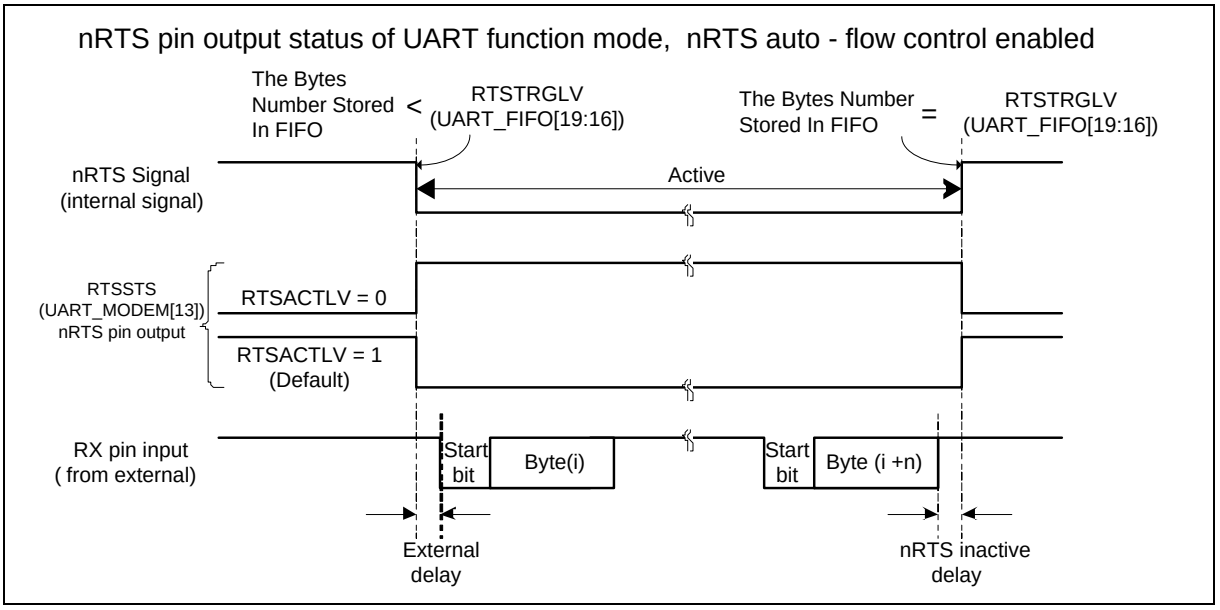


Figure 6.14-13 UART nRTS Auto-Flow Control Enabled

As shown in Figure 6.14-14, in software mode (ATORTSEN (UART_INTEN[12]) =0), the nRTS flow is directly controlled by software programming of RTS (UART_MODEM[1]) control bit.

Setting RTSACTLV (UART_MODEM[9]) can control the nRTS pin output is inverse or non-inverse from RTS (UART_MODEM[1]) control bit. User can read the RTSSTS (UART_MODEM[13]) bit to get

real nRTS pin output voltage logic status.

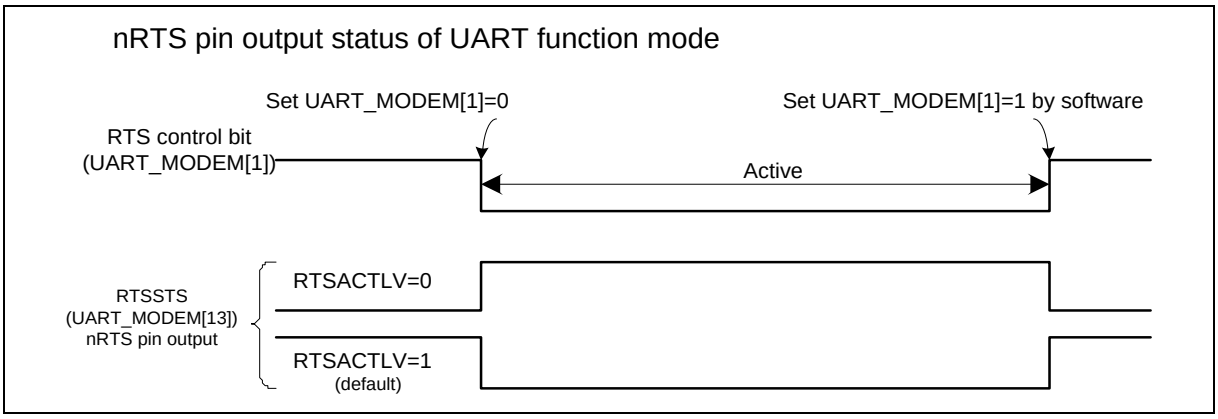


Figure 6.14-14 UART nRTS Auto-Flow with Software Control

6.14.5.9 IrDA Function Mode

The UART controller also provides Serial IrDA (SIR, Serial Infrared) function (Setting UART_FUNCSEL[2:0] to '010' to enable the IrDA function). The SIR specification defines a short-range infrared asynchronous serial transmission mode with one START bit, 8 data bits, and 1 STOP bit. The maximum data rate is 115.2 kbps. The IrDA SIR block contains an IrDA SIR protocol encoder/decoder. The IrDA SIR protocol is half-duplex only. So, it cannot transmit and receive data at the same time. The IrDA SIR physical layer specifies a minimum 10 ms transfer delay between transmission and reception, and this delay feature must be implemented by software.

In IrDA mode, the BAUDM1 (UART_BAUD[29]) must be cleared.

Baud Rate = Clock / (16 * (BRD +2)), where BRD (UART_BAUD[15:0]) is Baud Rate Divider in UART_BAUD register.

Note: The tolerance of baud-rate is ±5% between IrDA master and IrDA slave.

The IrDA control block diagram is shown in Figure 6.14-15.

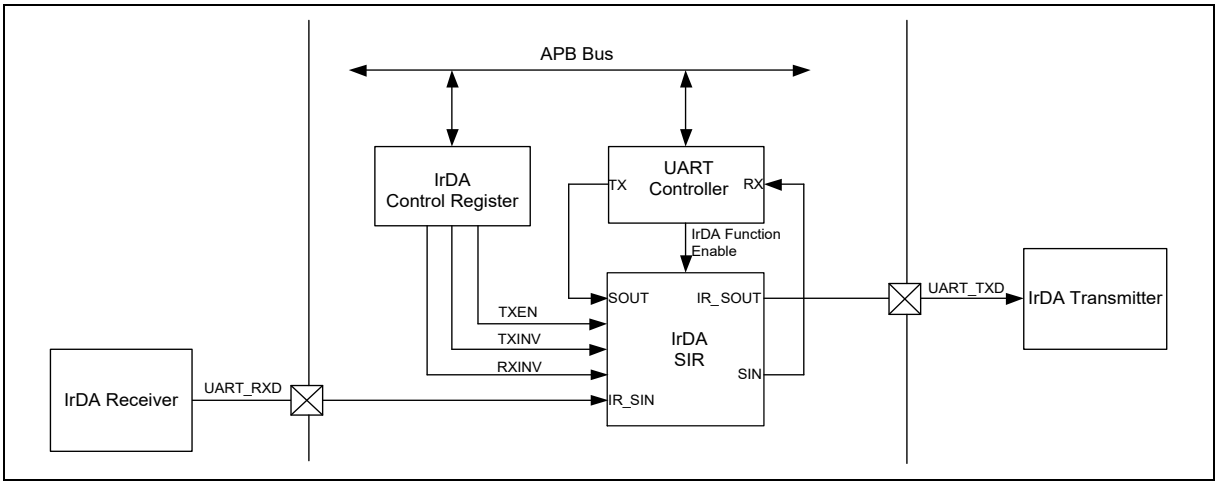


Figure 6.14-15 IrDA Control Block Diagram

IrDA SIR Transmit Encoder

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

The transmitted pulse width is specified as 3/16 period of baud rate.

IrDA SIR Receive Decoder

The IrDA SIR Receive Decoder demodulates the Return-to-Zero bit stream from the input detector and outputs the NRZ serial bits stream to the UART received data input.

In idle state, the decoder input is high. A START bit is detected when the decoder input is LOW. In normal operation, the RXINV (UART_IRDA[6]) is set to '1' and TXINV (UART_IRDA[5]) is set to '0'.

IrDA SIR Operation

The IrDA SIR encoder/decoder provides functionality which converts between UART data stream and half-duplex serial SIR interface. Figure 6.14-16 is IrDA encoder/decoder waveform.

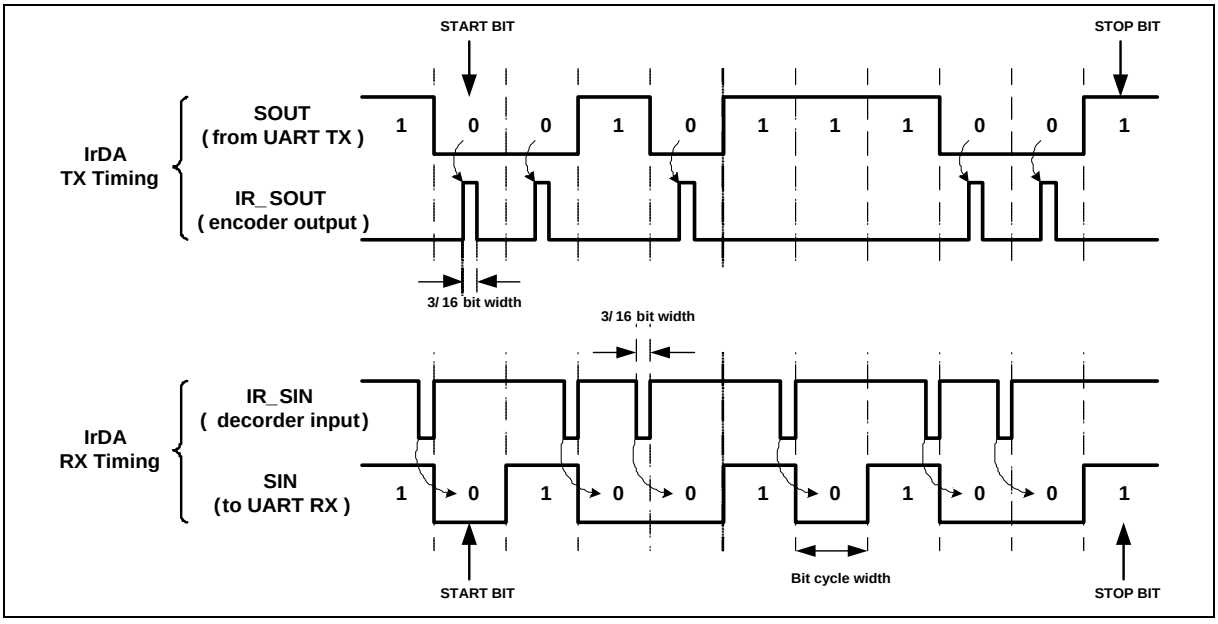


Figure 6.14-16 IrDA TX/RX Timing Diagram

6.14.5.10 LIN Function Mode (Local Interconnection Network)

The UART Controller supports LIN function. Setting FUNCSEL (UART_FUNCSEL[2:0]) to '001' to select LIN mode operation. The UART Controller supports LIN break/delimiter generation and break/delimiter detection in LIN master mode, and supports header detection and automatic resynchronization in LIN Slave mode.

Structure of LIN Frame

According to the LIN protocol, all information transmitted is packed as frames; a frame consists of a header (provided by the master task) and a response (provided by a slave task). The header (provided by the master task) consists of a break field and a sync field followed by a frame identifier (frame ID). The frame identifier uniquely defines the purpose of the frame. The slave task is appointed for providing the response associated with the frame ID. The response consists of a data field and a checksum field. Figure 6.14-17 is the structure of LIN Frame.

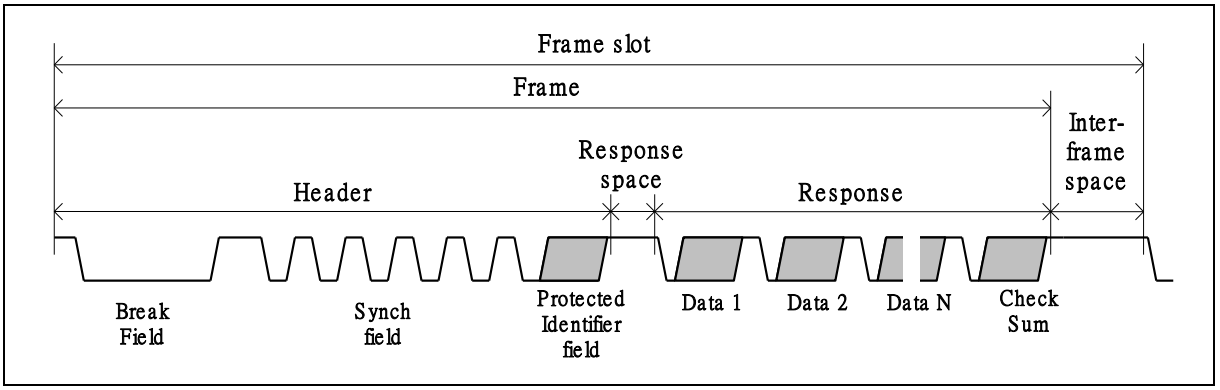


Figure 6.14-17 Structure of LIN Frame

Structure of LIN Byte

In LIN mode, each byte field is initiated by a START bit with value 0 (dominant), followed by 8 data bits and no PARITY bit, LSB is first and ended by 1 STOP bit with value 1 (recessive) in accordance with the LIN standard. The structure of Byte is shown in Figure 6.14-18.

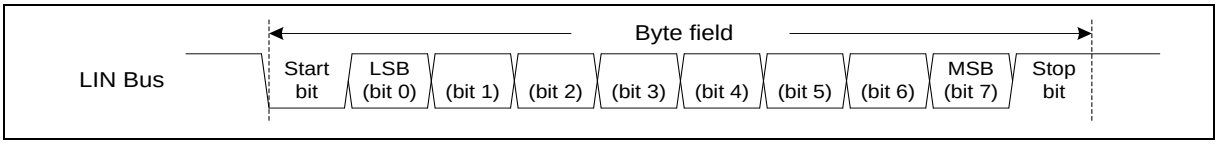


Figure 6.14-18 Structure of LIN Byte

LIN Master Mode

The UART Controller supports LIN Master mode. To enable and initialize the LIN Master mode, the following steps are necessary:

1. Set the UART_BAUD register to select the desired baud rate.
2. Set WLS (UART_LINE[1:0]) to '11' to configure the word length with 8 bits, clearing PBE (UART_LINE[3]) bit to disable parity check and clearing NSB (UART_LINE[2]) bit to configure with one STOP bit.
3. Set FUNCSEL (UART_FUNCSEL[2:0]) to '001' to select LIN function mode operation.

A complete header consists of a break field and sync field followed by a frame identifier (frame ID). The UART controller can be selected header sending by three header selected modes. The header selected mode can be "break field" or "break field and sync field" or "break field, sync field and frame ID field" by setting HSEL (UART_LINCTL[23:22]). If the selected header is "break field", software must handle the following sequence to send a complete header to bus by filling sync data (0x55) and frame ID data to the UART_DAT register. If the selected header is "break field and sync field", software must handle the sequence to send a complete header to bus by filling the frame ID data to UART_DAT register, and if the selected header is "break field, sync field and frame ID field", hardware will control the header sending sequence automatically but software must filled frame ID data to PID (UART_LINCTL[31:24]). When operating in header selected mode in which the selected header is "break field, sync field and frame ID field", the frame ID PARITY bit can be calculated by software or hardware depending whether the IDPEN (UART_LINCTL[9]) bit is set or not.

HSEL	Break Field	Sync Field	ID Field
0	Generated by Hardware	Handled by Software	Handled by Software
1	Generated by Hardware	Generated by Hardware	Handled by Software

2	Generated by Hardware	Generated by Hardware	Generated by Hardware (But Software needs to fill ID to PID (UART_LINCTL[31:24]) first)

Table 6.14-12 LIN Header Selection in Master Mode

When UART is operated in LIN data transmission, LIN bus transfer state can be monitored by hardware or software. User can enable hardware monitoring by setting BITERREN (UART_LINCTL[12]) to “1”, if the input pin (UART_RX) state is not equal to the output pin (UART_TX) state in LIN transmitter state that hardware will generate an interrupt to CPU. Software can also monitor the LIN bus transfer state by checking the read back data in UART_DAT register. The following sequence is a program sequence example.

The procedure without software error monitoring in Master mode:

1. Fill Protected Identifier to PID (UART_LINCTL[31:24]).
2. Select the hardware transmission header field including “break field + sync field + protected identifier field” by setting HSEL (UART_LINCTL[23:22]) to “10”.
3. Set SENDH (UART_LINCTL[8]) bit to 1 for requesting header transmission.
4. Wait until SENDH (UART_LINCTL[8]) bit cleared by hardware.
5. Wait until TXEMPTYF (UART_FIFOSTS[28]) set to 1 by hardware.

Note 1: The default setting of break field is 12 dominant bits (break field) and 1 recessive bit break/sync delimiter. Setting BRKFL (UART_LINCTL[19:16]) and BSL (UART_LINCTL[21:20]) to change the LIN break field length and break/sync delimiter length.

Note 2: The default setting of break/sync delimiter length is 1-bit time and the inter-byte spaces default setting is also 1-bit time. Setting BSL (UART_LINCTL[21:20]) and DLY (UART_TOUT[15:8]) can change break/sync delimiter length and inter-byte spaces.

Note 3: If the header includes the “break field, sync field and frame ID field”, software must fill frame ID to PID (UART_LINCTL[31:24]) before trigger header transmission (setting the SENDH (UART_LINCTL[8])). The frame ID parity can be generated by software or hardware depending on IDPEN (UART_LINCTL[9]) setting. If the parity generated by software with IDPEN (UART_LINCTL[9]) is set to ‘0’, software must fill 8 bit data (include 2 bit parity) in this field. If the parity generated by hardware with IDPEN (UART_LINCTL[9]) is set to ‘1’, software fills ID0~ID5 and hardware calculates P0 and P1.

Procedure with software error monitoring in Master mode:

1. Choose the hardware transmission header field to only include “break field” by setting HSEL (UART_LINCTL[23:22])] to ‘00’.
2. Enable break detection function by setting BRKDETEN (UART_LINCTL[10]).
3. Request break + break/sync delimiter transmission by setting the SENDH (UART_LINCTL[8]).
4. Wait until the BRKDETF (UART_LINSTS[8]) flag is set to “1” by hardware.
5. Request sync field transmission by writing 0x55 into UART_DAT register.
6. Wait until the RDAIF (UART_INTSTS[0]) is set to “1” by hardware and then read back the UART_DAT register.
7. Request header frame ID transmission by writing the protected identifier value to UART_DAT register.
8. Wait until the RDAIF (UART_INTSTS[0]) is set to “1” by hardware and then read back the UART_DAT register.

LIN Break and Delimiter Detection

When software enables the break detection function by setting BRKDETEN (UART_LINCTL[10]), the break detection circuit is activated. The break detection circuit is totally independent from the UART receiver.

When the break detection function is enabled, the circuit looks at the input UART_RX pin for a start signal. If UART LIN controller detects consecutive dominant is greater than 11 bits dominant followed by a recessive bit (delimiter), the BRKDETF (UART_LINSTS[8]) flag is set at the end of break field. If the LINIEN (UART_INTEN[8]) bit is set to 1, an interrupt LININT (UART_INTSTS[15]) will be generated. The behavior of the break detection and break flag are shown in Figure 6.14-19.

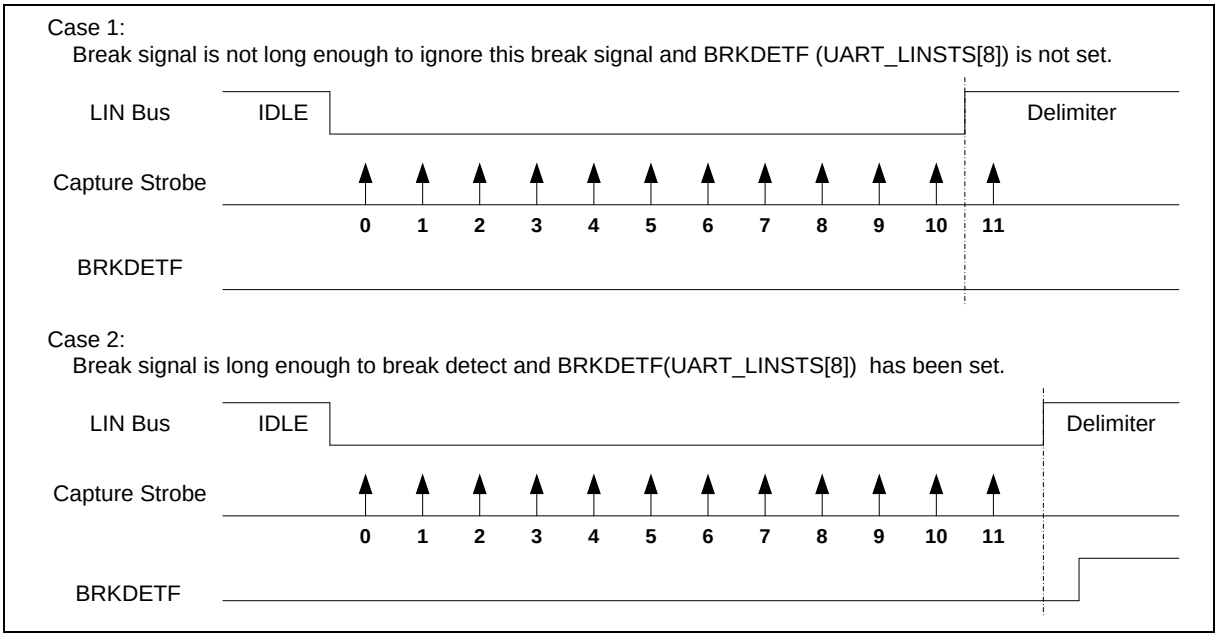


Figure 6.14-19 Break Detection in LIN Mode

LIN Frame ID and Parity Format

The LIN frame ID value in LIN function mode is shown, the frame ID parity can be generated by software or hardware depends on IDPEN (UART_LINCTL[9]).

If the parity generated by hardware (IDPEN (UART_LINCTL[9]) = 1), user fill ID0~ID5 (UART_LINCTL[29:24]) hardware will calculate P0 (UART_LINCTL[30]) and P1 (UART_LINCTL[31]) otherwise user must filled frame ID and parity in this field.

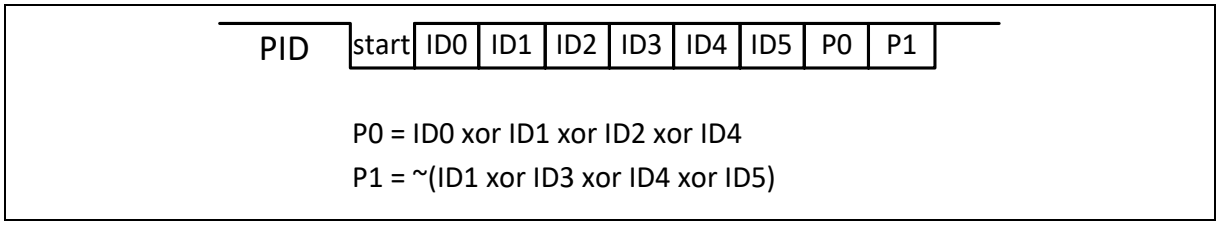


Figure 6.14-20 LIN Frame ID and Parity Format

LIN Slave Mode

The UART Controller supports LIN Slave mode. To enable and initialize the LIN Slave mode, the following steps are necessary:

1. Set the UART_BAUD register to select the desired baud rate.
2. Configure the data length to 8 bits by setting WLS (UART_LINE[1:0]) to '11' and disable parity check by clearing PBE (UART_LINE[3]) bit and configure with one STOP bit by clearing NSB

(UART_LINE[2]) bit.

3. Select LIN function mode by setting FUNCSEL (UART_FUNCSEL[2:0]) to '001'.
4. Enable LIN slave mode by setting the SLVEN (UART_LINCTL[0]) to 1.

LIN Header Reception

According to the LIN protocol, a slave node must wait for a valid header which comes from the master node. Next the slave task will take one of following actions (depend on the master header frame ID value).

- Receive the response.
- Transmit the response.
- Ignore the response and wait for next header.

In LIN Slave mode, user can enable the slave header detection function by setting the SLVHDEN (UART_LINCTL[1]) to detect complete frame header (receive "break field", "sync field" and "frame ID field"). When a LIN header is received, the SLVHDETF (UART_LINSTS[0]) flag will be set. If the LINIEN (UART_INTEN[8]) bit is set to 1, an interrupt will be generated. User can enable the frame ID parity check function by setting IDPEN (UART_LINCTL[9]). If only received frame ID parity is not correct (break and sync filed are correct), the SLVIDPEF (UART_LINSTS[2]) flag is set to '1'. If the LINIEN (UART_INTEN[8]) is set to 1, an interrupt will be generated and SLVHDETF (UART_LINSTS[0]) is set to '1'. User can also put LIN in mute mode by setting MUTE (UART_LINCTL[4]) to '1'. This mode allows detection of headers only (break + sync + frame ID) and prevents the reception of any other characters. In order to avoid bit rate tolerance, the controller supports automatic resynchronization function to avoid clock deviation error, user can enable this feature by setting SLVAREN (UART_LINCTL[2]).

LIN Response Transmission

The LIN slave node can transmit response and receive response. When slave node is the publisher of the response, the slave node sends response by filling data to the UART_DAT register. If the slave node is the subscriber of the response, the slave node receives data from LIN bus.

LIN Header Time-out Error

The LIN slave controller contains a header time-out counter. If the entire header is not received within the maximum time limit of 57 bit times, the header error flag SLVHEF (UART_LINSTS[1]) will be set. The time-out counter is enabled at each break detect edge and stopped in the following conditions.

- A LIN frame ID field has been received.
- The header error flag asserts.
- Writing 1 to the SLVSYNCF (UART_LINSTS[3]) to re-search a new frame header.

Mute Mode and LIN Exit from Mute Mode Condition

In Mute mode, a LIN slave node will not receive any data until specified condition occurred. It allows header detection only and prevents the reception of any other characters. User can enable Mute mode by setting the MUTE (UART_LINCTL[4]) and exiting from Mute mode condition can be selected by HSEL (UART_LINCTL[23:22]).

Note: It is recommended to set LIN slave node to Mute mode after checksum transmission.

The LIN slave controller exiting from Mute mode is described as follows: If HSEL (UART_LINCTL[23:22]) is set to "break field", when LIN slave controller detects a valid LIN break and delimiter, the controller will enable the receiver (exit from Mute mode) and subsequent data (sync data, frame ID data, response data) are received in RX FIFO.

If HSEL (UART_LINCTL[23:22]) is set to "break field and sync field", when the LIN slave controller detects a valid LIN break and delimiter followed by a valid sync field without frame error, the controller

will enable the receiver (exit from mute mode) and subsequent data (ID data, response data) are received in RX FIFO. If HSEL (UART_LINCTL[23:22]) is set to “break field, sync field and ID field”, when the LIN slave controller detects a valid LIN break and delimiter and valid sync field without frame error followed by ID data without frame error and received ID data matched PID (UART_LINCTL[31:24]) value. The controller will enable the receiver (exit from mute mode) and subsequent data (response data) are received in RX FIFO.

Slave Mode Non-automatic Resynchronization (NAR)

User can disable the automatic resynchronization function to fix the communication baud rate. When operating in Non-Automatic Resynchronization mode, software needs some initial process, and the initialization process flow of Non-Automatic Resynchronization mode is shown as follows:

1. Select the desired baud rate by setting the UART_BAUD register.
2. Select LIN function mode by setting FUNCSEL (UART_FUNCSEL[2:0]) to ‘001’.
3. Disable automatic resynchronization function by setting SLVAREN (UART_LINCTL[2]) is set to 0.
4. Enable LIN slave mode by setting the SLVEN (UART_LINCTL[0]) is set to 1.

Slave Mode with Automatic Resynchronization (AR)

In Automatic Resynchronization (AR) mode, the controller will adjust the baud rate generator after each sync field reception. The initialization process flow of Automatic Resynchronization mode is shown as follows:

1. Select the desired baud rate by setting the UART_BAUD register.
2. Select LIN function mode by setting UART_FUNCSEL (UART_FUNCSEL[2:0]) to ‘001’.
3. Enable automatic resynchronization function by setting SLVAREN (UART_LINCTL[2]) to ‘1’.
4. Enable LIN slave mode by setting the SLVEN (UART_LINCTL[0]) is set to ‘1’.

When the automatic resynchronization function is enabled, after each LIN break field, the time duration between five falling edges is sampled on peripheral clock and the result of this measurement is stored in an internal 13-bit register and the UART_BAUD register value will be automatically updated at the end of the fifth falling edge. If the measure timer (13-bit) overflows before five falling edges, then the header error flag SLVHEF (UART_LINSTS[1]) will be set.

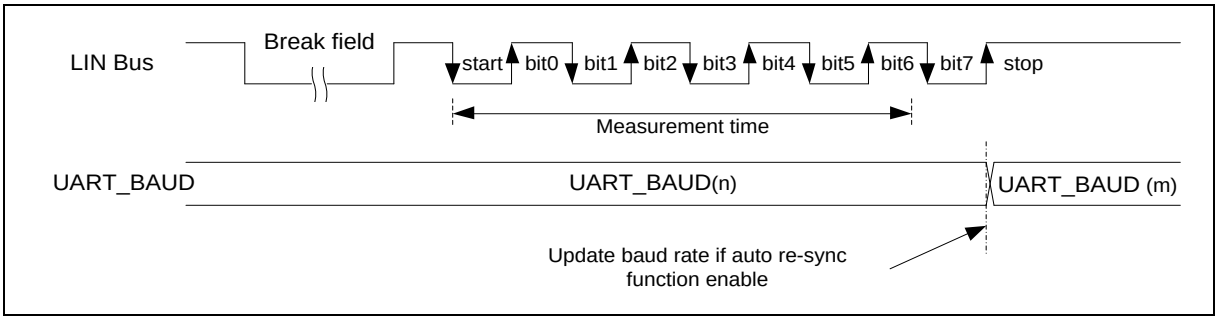


Figure 6.14-21 LIN Sync Field Measurement

When operating in Automatic Resynchronization (AR) mode, software must select the desired baud rate by setting the UART_BAUD register and hardware will store it at internal TEMP_REG register, after each LIN break field, the time duration between five falling edges is sampled on peripheral clock and the result of this measurement is stored in an internal 13-bit register BAUD_LIN and the result will be updated to UART_BAUD register automatically.

To guarantee the transmission baud rate, the baud rate generator must reload the initial value before each new break reception. The initial value is programmed by the application during initialization (TEMP_REG). User can set SLVDUEN (UART_LINCTL[3]) to enable auto reload initial baud rate

value function. If the SLVDUEN (UART_LINCTL[3]) is set, when received the next character, hardware will auto reload the initial value to UART_BAUD, and when the UART_BAUD be updated, the SLVDUEN (UART_LINCTL[3]) will be cleared automatically. The behavior of LIN updated method as shown in Figure 6.14-22.

Note 1: It is recommended to set the SLVDUEN bit before every checksum reception.

Note 2: When a header error is detected, user must write 1 to SLVSYNCF (UART_LINSTS[3]) to re-search new frame header. When writing 1 to it, hardware will reload the initial baud rate TEMP_REG and re-search new frame header.

Note 3: When operating in Automatic Resynchronization mode, the baud rate setting must be operated at mode2 (BAUDM1 (UART_BAUD[29]) and BAUDM0 (UART_BAUD[28]) must be 1).

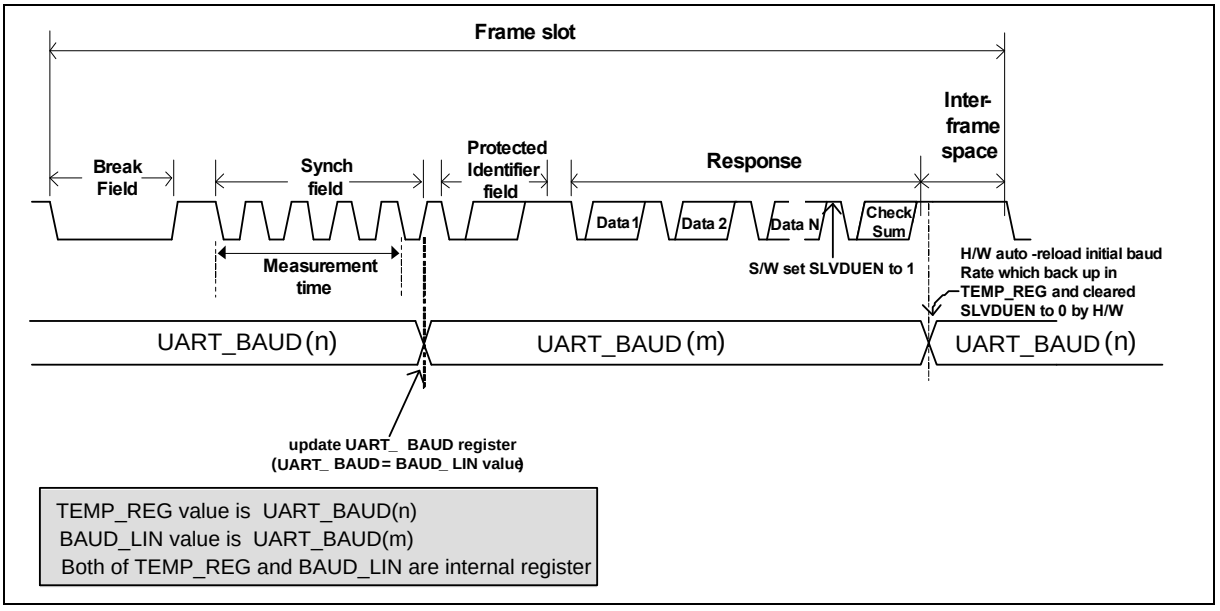


Figure 6.14-22 UART_BAUD Update Sequence in AR mode if SLVDUEN is 1

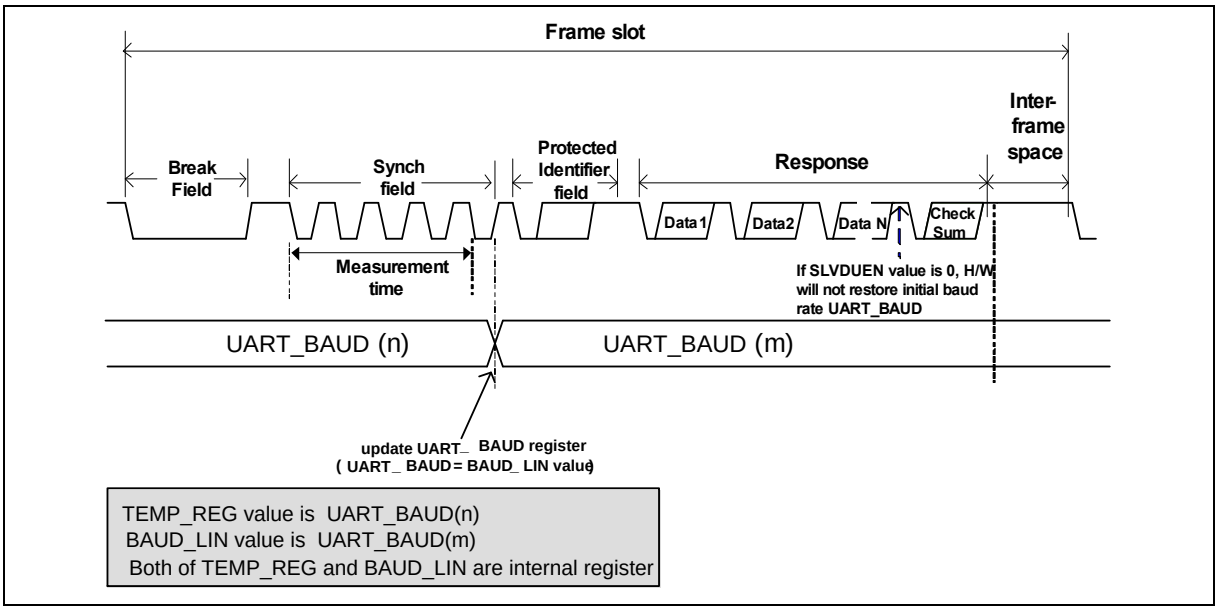


Figure 6.14-23 UART_BAUD Update Sequence in AR mode if SLVDUEN is 0

Deviation Error on the Sync Field

When operating in Automatic Resynchronization mode, the controller will check the deviation error on the sync field. The deviation error is checked by comparing the current baud rate with the received sync field. Two checks are performed in parallel.

Check1: Based on measurement between the first falling edge and the last falling edge of the sync field.

- If the difference is more than 14.84%, the header error flag SLVHEF (UART_LINSTS[1]) will be set.
- If the difference is less than 14.06%, the header error flag SLVHEF (UART_LINSTS[1]) will not be set.
- If the difference is between 14.84% and 14.06%, the header error flag SLVHEF (UART_LINSTS[1]) may either set or not.

Check2: Based on measurement of time between each falling edge of the sync field.

- If the difference is more than 18.75%, the header error flag SLVHEF (UART_LINSTS[1]) will be set.
- If the difference is less than 15.62%, the header error flag SLVHEF (UART_LINSTS[1]) will not be set.
- If the difference is between 18.75% and 15.62%, the header error flag SLVHEF (UART_LINSTS[1]) may either set or not.

Note: The deviation check is based on the current baud rate clock. Therefore, in order to guarantee correct deviation checking, the baud rate must reload the nominal value before each new break reception by setting SLVDUEN (UART_LINCTL[3]) register (It is recommend setting the SLVDUEN (UART_LINCTL[3]) bit before every checksum reception).

LIN Header Error Detection

In LIN Slave function mode, when user enables the header detection function by setting the SLVHDEN (UART_LINCTL[1]), hardware will handle the header detect flow. If the header has an error, the LIN header error flag SLVHEF (UART_LINSTS[1]) will be set and an interrupt is generated if the LINIEN (UART_INTEN[8]) bit is set. When header error is detected, user must reset the detect circuit to re-search a new frame header by writing 1 to SLVSYNCF (UART_LINSTS[3]) to re-search a new frame header.

The LIN header error flag SLVHEF (UART_LINSTS[1]) is set if one of the following conditions occurs:

- Break Delimiter is too short (less than 0.5-bit time).
- Frame error in sync field or Identifier field.
- The sync field data is not 0x55 (Non-Automatic Resynchronization mode).
- The sync field deviation error (With Automatic Resynchronization mode).
- The sync field measure time-out (With Automatic Resynchronization mode).
- LIN header reception time-out.

6.14.5.11 RS-485 Function Mode

Another alternate function of UART controller is RS-485 function (user must set UART_FUNCSEL[2:0] to '011' 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 the same as UART in RS-485 mode.

The UART controller can be configured as a RS-485 addressable slave or a RS-485 master. 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 9th bit is transmitted 0 and when PBE and SPE are set and EPE is cleared, the 9th 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), software must first decide the data which before the address byte be detected will be stored in RX FIFO or not. To ignore any data before address byte is detected, set RXOFF (UART_FIFO[8]) to 1 and enable RS485NMM (UART_ALTCTL[8]). The receiver will ignore any data until an address byte is detected (bit 9 = 1) and store the detected address byte data in the RX FIFO. To receive any data before address byte is detected, set RXOFF (UART_FIFO[8])to 0 and enable RS485NMM (UART_ALTCTL[8]).

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.

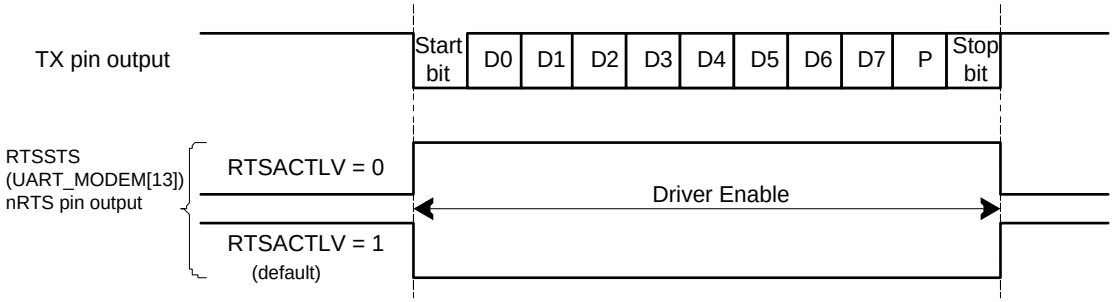
RS-485 Auto Direction Function (AUD)

RS-485 controllers supports 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 6.14-24 demonstrates the RS-485 nRTS driving level in AUD mode. The nRTS pin will be automatically driven during TX data transmission.

Setting RTSACTLV (UART_MODEM[9]) can control nRTS pin output driving level. User can read the RTSSTS (UART_MODEM[13]) bit to get real nRTS pin output voltage logic status.

nRTS pin output status of RS-485 function mode (RS-485 AUD mode enabled)

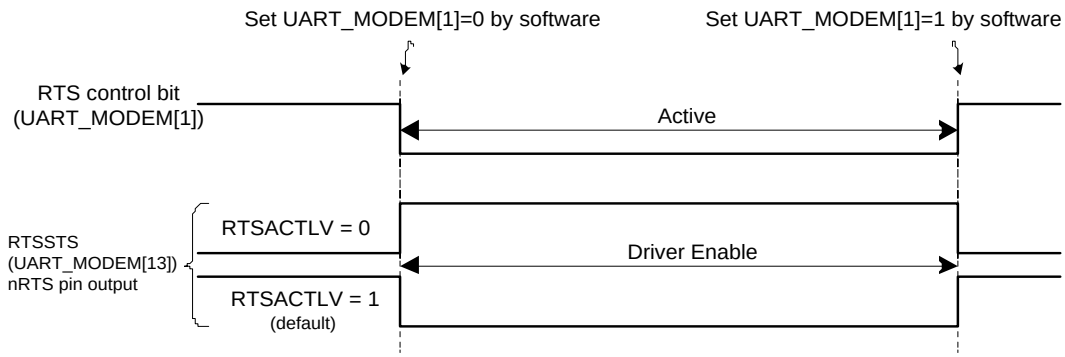


Note: RS485AUD (UART_ALTCTL[10]) = 1, the nRTS pin output by hardware control only.

Figure 6.14-24 RS-485 nRTS Driving Level in Auto Direction Mode

Figure 6.14-25 demonstrates the RS-485 nRTS driving level in software control (RS485AUD (UART_ALTCTL[10])=0). The nRTS driving level is controlled by programing the RTS (UART_MODEM[1]) control bit.

Setting RTSACTLV (UART_MODEM[9]) can control the nRTS pin output is inverse or non-inverse from RTS (UART_MODEM[1]) control bit. User can read the RTSSTS (UART_MODEM[13]) bit to get real nRTS pin output voltage logic status. The structure of RS-485 frame is shown in Figure 6.14-26.

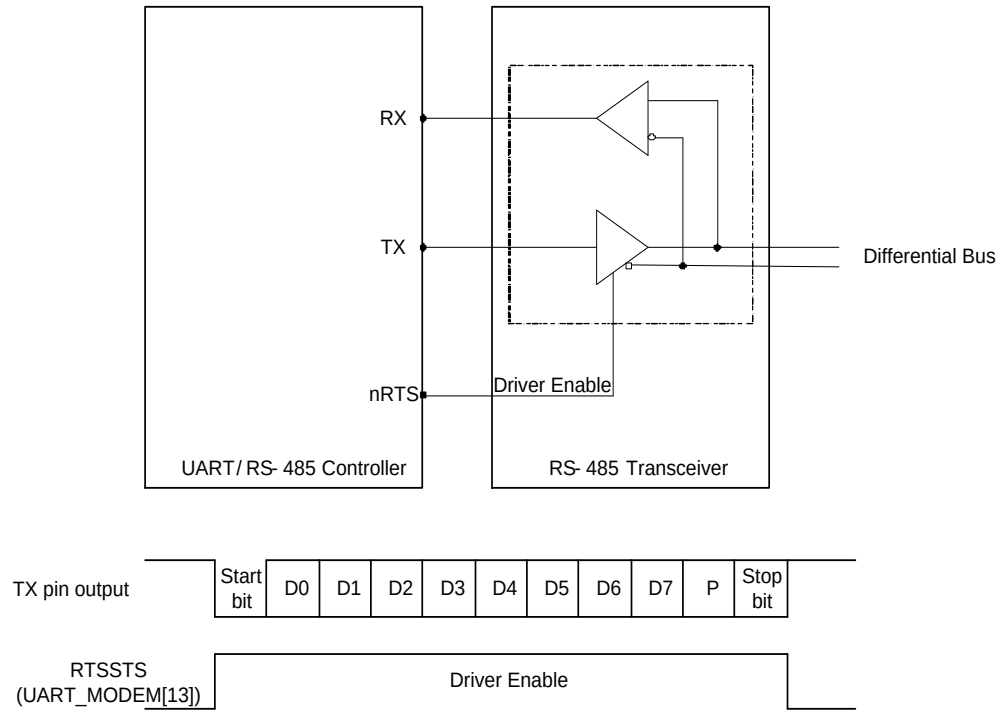


Note: RS485AUD (UART_ALTCTL[10]) = 0, support nRTS control bit by software control only.

Figure 6.14-25 RS-485 nRTS Driving Level with Software Control

Programming Sequence Example:

1. Program FUNCSEL in UART_FUNCSEL to select RS-485 function.
2. Program the RXOFF (UART_FIFO[8]) to determine enable or disable the receiver RS-485 receiver.
3. Program the RS485NMM (UART_ALTCTL[8]) or RS485AAD (UART_ALTCTL[9]) mode.
4. If the RS485AAD (UART_ALTCTL[9]) mode is selected, the ADDRMV (UART_ALTCTL[31:24]) is programmed for auto address match value.
5. Determine auto direction control by programming RS485AUD (UART_ALTCTL[10]).



Note: RS485AUD (UART_ALTCTL[10]) must be set to 1, and RTSACTLV (UART_MODEM[9]) must be set to 0.

Figure 6.14-26 Structure of RS-485 Frame

6.14.5.12 UART Single-wire Half Duplex

The UART controller provides single-wire half duplex function in UART function mode. Setting UART_FUNCSEL[2:0] to '100' to enable the UART Single-wire function. The single-wire bus is idle (RXIDLE (UART_FIFOSTS[29]) = 1) when in RX state. Before writing data to transmit buffer (UART_DAT[7:0]), the bus state should be checked in idle (RXIDLE (UART_FIFOSTS[29])). By writing data to transmit buffer, the bus state transfers to TX state immediately. After the transmittion, the bus state transfers from TX state to RX state.

The UART will not allowed to receive data when in single-wire half duplex function TX mode. If nRTS is asserted in TX mode, nRTS will make the docking UART device send out data and cause bus confliction. To reduce the bus confliction, the UART controller supports flow control function and bit error detection but do not support auto-flow control function. The nRTS is automatically inactivated in single-wire TX state. When in TX state, the UART controller will monitor bus state. If the bus state is not equal to TX state, the SWBEIF (UART_INTSTS[16]) is set.

6.14.5.13 PDMA Transfer Function

The UART controller supports PDMA transfer function.

By configuring PDMA parameter and setting UART_DAT as the PDMA destination address, when TXPDMAEN (UART_INTEN[14]) is set to 1, the UART controller will issue a request to the PDMA controller to start the PDMA transmission process automatically.

By configuring PDMA parameter and setting UART_DAT as the PDMA source address, when RXPDMAEN (UART_INTEN[15]) is set to 1, the controller will start the PDMA reception process. The UART controller will issue a request to the 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.

6.14.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
UART Base Address: **UARTx_BA = 0x4007_0000 + (0x1000 * x)** **x=0,1,2,3,4,5**				
UART_DAT **x=0,1,2,3,4,5**	UARTx_BA+0x00	R/W	UART Receive/Transmit Buffer Register	Undefined
UART_INTEN **x=0,1,2,3,4,5**	UARTx_BA+0x04	R/W	UART Interrupt Enable Register	0x0000_0000
UART_FIFO **x=0,1,2,3,4,5**	UARTx_BA+0x08	R/W	UART FIFO Control Register	0x0000_0101
UART_LINE **x=0,1,2,3,4,5**	UARTx_BA+0x0C	R/W	UART Line Control Register	0x0000_0000
UART_MODEM **x=0,1,2,3,4,5**	UARTx_BA+0x10	R/W	UART Modem Control Register	0x0000_0200
UART_MODEMSTS **x=0,1,2,3,4,5**	UARTx_BA+0x14	R/W	UART Modem Status Register	0x0000_0110
UART_FIFOSTS **x=0,1,2,3,4,5**	UARTx_BA+0x18	R/W	UART FIFO Status Register	0xB040_4000
UART_INTSTS **x=0,1,2,3,4,5**	UARTx_BA+0x1C	R/W	UART Interrupt Status Register	0x0040_0002
UART_TOUT **x=0,1,2,3,4,5**	UARTx_BA+0x20	R/W	UART Time-out Register	0x0000_0000
UART_BAUD **x=0,1,2,3,4,5**	UARTx_BA+0x24	R/W	UART Baud Rate Divider Register	0x0F00_0000
UART_IRDA **x=0,1,2,3,4,5**	UARTx_BA+0x28	R/W	UART IrDA Control Register	0x0000_0040
UART_ALTCTL **x=0,1,2,3,4,5**	UARTx_BA+0x2C	R/W	UART Alternate Control/Status Register	0x0000_000C
UART_FUNCSEL **x=0,1,2,3,4,5**	UARTx_BA+0x30	R/W	UART Function Select Register	0x0000_0000
UART_LINCTL **x=0,1**	UARTx_BA+0x34	R/W	UART LIN Control Register	0x000C_0000
UART_LINSTS **x=0,1**	UARTx_BA+0x38	R/W	UART LIN Status Register	0x0000_0000

UART_BRCOMP **x=0,1**	UARTx_BA+0x3C	R/W	UART Baud Rate Compensation Register	0x0000_0000
UART_WKCTL **x=0,1,2,3,4,5**	UARTx_BA+0x40	R/W	UART Wake-up Control Register	0x0000_0000
UART_WKSTS **x=0,1,2,3,4,5**	UARTx_BA+0x44	R/W	UART Wake-up Status Register	0x0000_0000
UART_DWKCOMP **x=0,1,2,3,4,5**	UARTx_BA+0x48	R/W	UART Incoming Data Wake-up Compensation Register	0x0000_0000

6.14.7 Register Description

UART Receive/Transmit Buffer Register (UART_DAT)

Register	Offset	R/W	Description	Reset Value
UART_DAT **x=0,1,2,3,4,5**	UARTx_BA+0x00	R/W	UART Receive/Transmit Buffer Register	Undefined

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

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**PARITY**	**PARITY Bit Receive/Transmit Buffer** Write Operation: By writing to this bit, the PARITY bit will be stored in transmitter FIFO. If PBE (UART_LINE[3]) and PSS (UART_LINE[7]) are set, the UART controller will send out this bit follow the DAT (UART_DAT[7:0]) through the UART_TXD. Read Operation: If PBE (UART_LINE[3]) and PSS (UART_LINE[7]) are enabled, the PARITY bit can be read by this bit. **Note:** This bit has effect only when PBE (UART_LINE[3]) and PSS (UART_LINE[7]) are set.
[7:0]	**DAT**	**Data Receive/Transmit Buffer** Write Operation: By writing one byte to this register, the data byte will be stored in transmitter FIFO. The UART controller will send out the data stored in transmitter FIFO top location through the UART_TXD. Read Operation: By reading this register, the UART controller will return an 8-bit data received from receiver FIFO.

UART Interrupt Enable Register (UART_INTEN)

Register	Offset	R/W	Description	Reset Value
UART_INTEN x=0,1,2,3,4,5	UARTx_BA+0x04	R/W	UART Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved	TXENDIEN	Reserved			ABRIEN	Reserved	SWBEIEN
15	14	13	12	11	10	9	8
RXPDMAEN	TXPDMAEN	ATOCTSEN	ATORTSEN	TOCNTEN	Reserved		LINIEN
7	6	5	4	3	2	1	0
Reserved	WKIEN	BUFERRIEN	RXTOIEN	MODEMIEN	RLSIEN	THREIEN	RDAIEN

Bits	Description	
[31:23]	**Reserved**	Reserved.
[22]	**TXENDIEN**	**Transmitter Empty Interrupt Enable Bit** If TXENDIEN (UART_INTEN[22]) is enabled, the Transmitter Empty interrupt TXENDINT (UART_INTSTS[30]) will be generated when TXENDIF (UART_INTSTS[22]) is set (TX FIFO (UART_DAT) is empty and the STOP bit of the last byte has been transmitted). 0 = Transmitter empty interrupt Disabled. 1 = Transmitter empty interrupt Enabled.
[21:19]	**Reserved**	Reserved.
[18]	**ABRIEN**	**Auto-baud Rate Interrupt Enable Bit** 0 = Auto-baud rate interrupt Disabled. 1 = Auto-baud rate interrupt Enabled.
[17]	**Reserved**	Reserved.
[16]	**SWBEIEN**	**Single-wire Bit Error Detection Interrupt Enable Bit** Set this bit, the Single-wire Half Duplex Bit Error Detection Interrupt SWBEINT (UART_INTSTS[24]) is generated when Single-wire Bit Error Detection SWBEIF (UART_INTSTS[16]) is set. 0 = Single-wire Bit Error Detect Inerrupt Disabled. 1 = Single-wire Bit Error Detect Inerrupt Enabled. **Note:** This bit is valid when FUNCSEL (UART_FUNCSEL[2:0]) is set as UART Single-wire mode.
[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 stopped. 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** 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 stopped. 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.
[10:9]	**Reserved**	Reserved.
[8]	**LINIEN**	**LIN Bus Interrupt Enable Bit** 0 = LIN bus interrupt Disabled. 1 = LIN bus interrupt Enabled. **Note:** This bit is used for LIN function mode.
[7]	**Reserved**	Reserved.
[6]	**WKIEN**	**Wake-up Interrupt Enable Bit** 0 = Wake-up Interrupt Disabled. 1 = Wake-up Interrupt Enabled.
[5]	**BUFERRIEN**	**Buffer Error Interrupt Enable Bit** 0 = Buffer error interrupt Disabled. 1 = Buffer error interrupt Enabled.
[4]	**RXTOIEN**	**RX Time-out Interrupt Enable Bit** 0 = RX time-out interrupt Disabled. 1 = RX time-out interrupt Enabled.
[3]	**MODEMIEN**	**Modem Status Interrupt Enable Bit** 0 = Modem status interrupt Disabled. 1 = Modem status interrupt Enabled.
[2]	**RLSIEN**	**Receive Line Status Interrupt Enable Bit** 0 = Receive Line Status interrupt Disabled. 1 = Receive Line Status interrupt Enabled.
[1]	**THREIEN**	**Transmit Holding Register Empty Interrupt Enable Bit** 0 = Transmit holding register empty interrupt Disabled. 1 = Transmit holding register empty interrupt Enabled.

[0]	**RDAIEN**	**Receive Data Available Interrupt Enable Bit** 0 = Receive data available interrupt Disabled. 1 = Receive data available interrupt Enabled.

UART FIFO Control Register (UART_FIFO)

Register	Offset	R/W	Description	Reset Value
UART_FIFO **x=0,1,2,3,4,5**	UARTx_BA+0x08	R/W	UART FIFO Control Register	0x0000_0101

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

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19:16]	**RTSTRGLV**	**nRTS Trigger Level for Auto-flow Control** 0000 = nRTS Trigger Level is 1 byte. 0001 = nRTS Trigger Level is 4 bytes. 0010 = nRTS Trigger Level is 8 bytes. 0011 = nRTS Trigger Level is 14 bytes. Others = Reserved. **Note:** This field is used for auto nRTS flow control.
[15:9]	**Reserved**	Reserved.
[8]	**RXOFF**	**Receiver Disable Bit** The receiver is disabled or not (set 1 to disable receiver). 0 = Receiver Enabled. 1 = Receiver Disabled. **Note:** This bit is used for RS-485 Normal Multi-drop mode. It should be programmed before RS485NMM (UART_ALTCTL[8]) is programmed.
[7:4]	**RFITL**	**RX FIFO Interrupt Trigger Level** When the number of bytes in the receive FIFO equals the RFITL, the RDAIF (UART_INTSTS[0]) will be set (if RDAIEN (UART_INTEN[0]) enabled, and an interrupt will be generated). 0000 = RX FIFO Interrupt Trigger Level is 1 byte. 0001 = RX FIFO Interrupt Trigger Level is 4 bytes. 0010 = RX FIFO Interrupt Trigger Level is 8 bytes. 0011 = RX FIFO Interrupt Trigger Level is 14 bytes. Others = Reserved.
[3]	**Reserved**	Reserved.
[2]	**TXRST**	**TX Field Software Reset** When TXRST (UART_FIFO[2]) is set, all the byte in the transmit FIFO and TX internal state machine are

		cleared. 0 = No effect. 1 = Reset the TX internal state machine and pointers. **Note 1:** This bit will automatically clear at least 3 UART peripheral clock cycles. **Note 2:** Before setting this bit, it should wait for the TXEMPTYF (UART_FIFOSTS[28]) to 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. **Note 1:** This bit will automatically clear at least 3 UART peripheral clock cycles. **Note 2:** Before setting this bit, it should wait for the RXIDLE (UART_FIFOSTS[29]) to be set.
[0]	**Reserved**	Reserved.

UART Line Control Register (UART_LINE)

Register	Offset	R/W	Description	Reset Value
UART_LINE **x=0,1,2,3,4,5**	UARTx_BA+0x0C	R/W	UART Line Control Register	0x0000_0000

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

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**RXDINV**	**RX Data Inverted** 0 = Received data signal inverted Disabled. 1 = Received data signal inverted Enabled. **Note 1:** Before setting this bit, TXRXDIS (UART_FUNCSEL[3]) should be set then wait for TXRXACT (UART_FIFOSTS[31]) to be cleared. When the configuration is done, clear TXRXDIS (UART_FUNCSEL[3]) to activate UART controller. **Note 2:** This bit is valid when FUNCSEL (UART_FUNCSEL[2:0]) is set as UART, LIN or RS485 function.
[8]	**TXDINV**	**TX Data Inverted** 0 = Transmitted data signal inverted Disabled. 1 = Transmitted data signal inverted Enabled. **Note 1:** Before setting this bit, TXRXDIS (UART_FUNCSEL[3]) should be set then wait for TXRXACT (UART_FIFOSTS[31]) to be cleared. When the configuration is done, clear TXRXDIS (UART_FUNCSEL[3]) to activate UART controller. **Note 2:** This bit is valid when FUNCSEL (UART_FUNCSEL[2:0]) is set as UART, LIN or RS485 function.
[7]	**PSS**	**PARITY Bit Source Selection** The PARITY bit can be selected to be generated and checked automatically or by software. 0 = PARITY bit is generated by EPE (UART_LINE[4]) and SPE (UART_LINE[5]) setting and checked automatically. 1 = PARITY bit generated and checked by software. **Note 1:** This bit has effect only when PBE (UART_LINE[3]) is set. **Note 2:** If PSS is 0, the PARITY bit is transmitted and checked automatically. If PSS is 1, the transmitted PARITY bit value can be determined by writing PARITY (UART_DAT[8]) and the PARITY bit can be read by reading PARITY (UART_DAT[8]).
[6]	**BCB**	**Break Control Bit** 0 = Break Control Disabled. 1 = Break Control Enabled. **Note:** When this bit is set to logic 1, the transmitted serial data output (TX) is forced to the Spacing State (logic 0). This bit acts only on TX line and has no effect on the transmitter logic.

[5]	**SPE**	**Stick Parity Enable Bit** 0 = Stick parity Disabled. 1 = Stick parity Enabled. **Note:** If PBE (UART_LINE[3]) and EPE (UART_LINE[4]) are logic 1, the PARITY bit is transmitted and checked as logic 0. If PBE (UART_LINE[3]) is 1 and EPE (UART_LINE[4]) is 0 then the PARITY bit is transmitted and checked as 1.
[4]	**EPE**	**Even Parity Enable Bit** 0 = Odd number of logic 1's is transmitted and checked in each word. 1 = Even number of logic 1's is transmitted and checked in each word. **Note:** This bit has effect only when PBE (UART_LINE[3]) is set.
[3]	**PBE**	**PARITY Bit Enable Bit** 0 = PARITY bit generated Disabled. 1 = PARITY bit generated Enabled. **Note:** PARITY bit is generated on each outgoing character and is checked on each incoming data.
[2]	**NSB**	**Number of "STOP Bit"** 0 = One "STOP bit" is generated in the transmitted data. 1 = When select 5-bit word length, 1.5 "STOP bit" is generated in the transmitted data. When select 6-, 7- and 8-bit word length, 2 "STOP bit" is generated in the transmitted data.
[1:0]	**WLS**	**Word Length Selection** This field sets UART word length. 00 = 5 bits. 01 = 6 bits. 10 = 7 bits. 11 = 8 bits.

UART Modem Control Register (UART_MODEM)

Register	Offset	R/W	Description	Reset Value
UART_MODEM **x=0,1,2,3,4,5**	UARTx_BA+0x10	R/W	UART Modem Control Register	0x0000_0200

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

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13]	**RTSSTS**	**nRTS Pin Status (Read Only)** This bit mirror from nRTS pin output of voltage logic status. 0 = nRTS pin output is low level voltage logic state. 1 = nRTS pin output is high level voltage logic state.
[12:10]	**Reserved**	Reserved.
[9]	**RTSACTLV**	**nRTS Pin Active Level** This bit defines the active level state of nRTS pin output. 0 = nRTS pin output is high level active. 1 = nRTS pin output is low level active. (Default) **Note 1:** Refer to Figure 6.14-13 and Figure 6.14-14 for UART function mode. **Note 2:** Refer to Figure 6.14-24 and Figure 6.14-25 for RS-485 function mode. **Note 3:** Before setting this bit, TXRXDIS (UART_FUNCSEL[3]) should be set then wait for TXRXACT (UART_FIFOSTS[31]) to be cleared. When the configuration is done, clear TXRXDIS (UART_FUNCSEL[3]) to activate UART controller.
[8:2]	**Reserved**	Reserved.
[1]	**RTS**	**nRTS (Request-to-send) Signal Control** This bit is direct control internal nRTS signal active or not, and then drive the nRTS pin output with RTSACTLV bit configuration. 0 = nRTS signal is active. 1 = nRTS signal is inactive. **Note 1:** The nRTS signal control bit is not effective when nRTS auto-flow control is enabled in UART function mode. **Note 2:** The nRTS signal control bit is not effective when RS-485 auto direction mode (AUD) is enabled in RS-485 function mode. **Note 3:** Single-wire mode supports this feature.
[0]	**Reserved**	Reserved.

UART Modem Status Register (UART_MODEMSTS)

Register	Offset	R/W	Description	Reset Value
UART_MODEMSTS **x=0,1,2,3,4,5**	UARTx_BA+0x14	R/W	UART Modem Status Register	0x0000_0110

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

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**CTSACTLV**	**nCTS Pin Active Level** This bit defines the active level state of nCTS pin input. 0 = nCTS pin input is high level active. 1 = nCTS pin input is low level active. (Default) **Note:** Before setting this bit, TXRXDIS (UART_FUNCSEL[3]) should be set then wait for TXRXACT (UART_FIFOSTS[31]) to be cleared. When the configuration is done, clear TXRXDIS (UART_FUNCSEL[3]) to activate UART controller.
[7:5]	**Reserved**	Reserved.
[4]	**CTSSTS**	**nCTS Pin Status (Read Only)** This bit mirror from nCTS pin input of voltage logic status. 0 = nCTS pin input is low level voltage logic state. 1 = nCTS pin input is high level voltage logic state. **Note:** This bit echoes when UART controller peripheral clock is enabled, and nCTS multi-function port is selected.
[3:1]	**Reserved**	Reserved.
[0]	**CTSDETF**	**Detect nCTS State Change Flag** This bit is set whenever nCTS input has change state, and it will generate Modem interrupt to CPU when MODEMIEN (UART_INTEN[3]) is set to 1. 0 = nCTS input has not change state. 1 = nCTS input has change state. **Note:** This bit can be cleared by writing “1” to it.

UART FIFO Status Register (UART_FIFOSTS)

Register	Offset	R/W	Description	Reset Value
UART_FIFOSTS **x=0,1,2,3,4,5**	UARTx_BA+0x18	R/W	UART FIFO Status Register	0xB040_4000

31	30	29	28	27	26	25	24
TXRXACT	Reserved	RXIDLE	TXEMPTYF	Reserved			TXOVIF
23	22	21	20	19	18	17	16
TXFULL	TXEMPTY	TXPTR					
15	14	13	12	11	10	9	8
RXFULL	RXEMPTY	RXPTR					
7	6	5	4	3	2	1	0
Reserved	BIF	FEF	PEF	ADDRDETF	ABRDTOIF	ABRDIF	RXOVIF

Bits	Description	
[31]	**TXRXACT**	**TX and RX Active Status (Read Only)** This bit indicates TX and RX are active or inactive. 0 = TX and RX are inactive. 1 = TX and RX are active. (Default) **Note:** When TXRXDIS (UART_FUNCSEL[3]) is set and both TX and RX are in idle state, this bit is cleared. The UART controller can not transmit or receive data at this moment. Otherwise this bit is set.
[30]	**Reserved**	Reserved.
[29]	**RXIDLE**	**RX Idle Status (Read Only)** This bit is set by hardware when RX is idle. 0 = RX is busy. 1 = RX is idle. (Default)
[28]	**TXEMPTYF**	**Transmitter Empty Flag (Read Only)** This bit is set by hardware when TX FIFO (UART_DAT) is empty and the STOP bit of the last byte has been transmitted. 0 = TX FIFO is not empty or the STOP bit of the last byte has been not transmitted. 1 = TX FIFO is empty and the STOP bit of the last byte has been transmitted. **Note:** This bit is cleared automatically when TX FIFO is not empty or the last byte transmission has not completed.
[27:25]	**Reserved**	Reserved.
[24]	**TXOVIF**	**TX Overflow Error Interrupt Flag** If TX FIFO (UART_DAT) is full, an additional write to UART_DAT will cause this bit to logic 1. 0 = TX FIFO is not overflow. 1 = TX FIFO is overflow. **Note:** This bit can be cleared by writing “1” to it.
[23]	**TXFULL**	**Transmitter FIFO Full (Read Only)** This bit indicates TX FIFO full or not.

		0 = TX FIFO is not full. 1 = TX FIFO is full. **Note:** This bit is set when the number of usage in TX FIFO Buffer is equal to 16, otherwise it is cleared by hardware.
[22]	**TXEMPTY**	**Transmitter FIFO Empty (Read Only)** This bit indicates TX FIFO empty or not. 0 = TX FIFO is not empty. 1 = TX FIFO is empty. **Note:** When the last byte of TX FIFO has been transferred to Transmitter Shift Register, hardware sets this bit high. It will be cleared when writing data into UART_DAT (TX FIFO not empty).
[21:16]	**TXPTR**	**TX FIFO Pointer (Read Only)** This field indicates the TX FIFO Buffer Pointer. When CPU writes one byte into UART_DAT, TXPTR increases one. When one byte of TX FIFO is transferred to Transmitter Shift Register, TXPTR decreases one. The Maximum value shown in TXPTR is 15. When the using level of TX FIFO Buffer equal to 16, the TXFULL bit is set to 1 and TXPTR will show 0. As one byte of TX FIFO is transferred to Transmitter Shift Register, the TXFULL bit is cleared to 0 and TXPTR will show 15.
[15]	**RXFULL**	**Receiver FIFO Full (Read Only)** This bit initiates RX FIFO full or not. 0 = RX FIFO is not full. 1 = RX FIFO is full. **Note:** This bit is set when the number of usage in RX FIFO Buffer is equal to 16, otherwise it is cleared by hardware.
[14]	**RXEMPTY**	**Receiver FIFO Empty (Read Only)** This bit initiate RX FIFO empty or not. 0 = RX FIFO is not empty. 1 = RX FIFO is empty. **Note:** When the last byte of RX FIFO has been read by CPU, hardware sets this bit high. It will be cleared when UART receives any new data.
[13:8]	**RXPTR**	**RX FIFO Pointer (Read Only)** This field indicates the RX FIFO Buffer Pointer. When UART receives one byte from external device, RXPTR increases one. When one byte of RX FIFO is read by CPU, RXPTR decreases one. The Maximum value shown in RXPTR is 15. When the using level of RX FIFO Buffer equal to 16, the RXFULL bit is set to 1 and RXPTR will show 0. As one byte of RX FIFO is read by CPU, the RXFULL bit is cleared to 0 and RXPTR will show 15.
[7]	**Reserved**	Reserved.
[6]	**BIF**	**Break Interrupt Flag** This bit is set to logic 1 whenever the received data input (RX) is held in the “spacing state” (logic 0) for longer than a full word transmission time (that is, the total time of “START bit” + data bits + parity + STOP bits). 0 = No Break interrupt is generated. 1 = Break interrupt is generated. **Note:** This bit can be cleared by writing “1” to it or updated by reading UART_DAT.
[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 or updated by reading UART_DAT.
[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 or updated by reading UART_DAT.
[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’). **Note 1:** This field is used for RS-485 function mode and ADDRDEN (UART_ALTCTL[15]) is set to 1 to enable Address detection mode. **Note 2:** This bit can be cleared by writing “1” to it.
[2]	**ABRDTOIF**	**Auto-baud Rate Detect Time-out Interrupt Flag** This bit is set to logic “1” in Auto-baud Rate Detect mode when the baud rate counter is overflow. 0 = Auto-baud rate counter is underflow. 1 = Auto-baud rate counter is overflow. **Note:** This bit can be cleared by writing “1” to it.
[1]	**ABRDIF**	**Auto-baud Rate Detect Interrupt Flag** This bit is set to logic “1” when auto-baud rate detect function is finished. 0 = Auto-baud rate detect function is not finished. 1 = Auto-baud rate detect function is finished. **Note:** This bit can be cleared by writing “1” to it.
[0]	**RXOVIF**	**RX Overflow Error Interrupt Flag** This bit is set when RX FIFO overflow. If the number of bytes of received data is greater than RX_FIFO (UART_DAT) size 16 bytes, this bit will be set. 0 = RX FIFO is not overflow. 1 = RX FIFO is overflow. **Note:** This bit can be cleared by writing “1” to it.

UART Interrupt Status Register (UART_INTSTS)

Register	Offset	R/W	Description	Reset Value
UART_INTSTS **x=0,1,2,3,4,5**	UARTx_BA+0x1C	R/W	UART Interrupt Status Register	0x0040_0002

31	30	29	28	27	26	25	24
ABRINT	TXENDINT	HWBUFEINT	HWTOINT	HWMODINT	HWRLSINT	Reserved	SWBEINT
23	22	21	20	19	18	17	16
Reserved	TXENDIF	HWBUFEIF	HWTOIF	HWMODIF	HWRLSIF	Reserved	SWBEIF
15	14	13	12	11	10	9	8
LININT	WKINT	BUFERRINT	RXTOINT	MODEMINT	RLSINT	THREINT	RDAINT
7	6	5	4	3	2	1	0
LINIF	WKIF	BUFERRIF	RXTOIF	MODEMIF	RLSIF	THREIF	RDAIF

Bits	Description	
[31]	**ABRINT**	**Auto-baud Rate Interrupt Indicator (Read Only)** This bit is set if ABRIEN (UART_INTEN[18]) and ABRIF (UART_ALTCTL[17]) are both set to 1. 0 = No Auto-baud Rate interrupt is generated. 1 = The Auto-baud Rate interrupt is generated.
[30]	**TXENDINT**	**Transmitter Empty Interrupt Indicator (Read Only)** This bit is set if TXENDIEN (UART_INTEN[22]) and TXENDIF (UART_INTSTS[22]) are both set to 1. 0 = No Transmitter Empty interrupt is generated. 1 = Transmitter Empty interrupt is generated.
[29]	**HWBUFEINT**	**PDMA Mode Buffer Error Interrupt Indicator (Read Only)** This bit is set if BUFERRIEN (UART_INTEN[5]) and HWBUFEIF (UART_INTSTS[21]) are both set to 1. 0 = No buffer error interrupt is generated in PDMA mode. 1 = Buffer error interrupt is generated in PDMA mode.
[28]	**HWTOINT**	**PDMA Mode RX Time-out Interrupt Indicator (Read Only)** This bit is set if RXTOIEN (UART_INTEN[4]) and HWTOIF (UART_INTSTS[20]) are both set to 1. 0 = No RX time-out interrupt is generated in PDMA mode. 1 = RX time-out interrupt is generated in PDMA mode.
[27]	**HWMODINT**	**PDMA Mode MODEM Status Interrupt Indicator (Read Only)** This bit is set if MODEMIEN (UART_INTEN[3]) and HWMODIF (UART_INTSTS[19]) are both set to 1. 0 = No Modem interrupt is generated 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]	**Reserved**	Reserved.

[24]	**SWBEINT**	**Single-wire Bit Error Detect Interrupt Indicator (Read Only)** This bit is set if SWBEIEN (UART_INTEN[16]) and SWBEIF (UART_INTSTS[16]) are both set to 1. 0 = No Single-wire Bit Error Detection Interrupt generated. 1 = Single-wire Bit Error Detection Interrupt generated.
[23]	**Reserved**	Reserved.
[22]	**TXENDIF**	**Transmitter Empty Interrupt Flag** This bit is set when TX FIFO (UART_DAT) is empty and the STOP bit of the last byte has been transmitted (TXEMPTYF (UART_FIFOSTS[28]) is set). If TXENDIEN (UART_INTEN[22]) is enabled, the Transmitter Empty interrupt will be generated. 0 = No transmitter empty interrupt flag is generated. 1 = Transmitter empty interrupt flag is generated. **Note:** This bit is cleared automatically when TX FIFO is not empty or the last byte transmission has not completed.
[21]	**HWBUFEIF**	**PDMA Mode Buffer Error Interrupt Flag (Read Only)** This bit is set when the TX or RX FIFO overflows (TXOVIF (UART_FIFOSTS[24]) or RXOVIF (UART_FIFOSTS[0]) is set). When BUFERRIF (UART_INTSTS[5]) is set, the transfer maybe is not correct. If BUFERRIEN (UART_INTEN[5]) is enabled, the buffer error interrupt will be generated. 0 = No buffer error interrupt flag is generated in PDMA mode. 1 = Buffer error interrupt flag is generated in PDMA mode. **Note:** This bit is cleared when both TXOVIF (UART_FIFOSTS[24]]) and RXOVIF (UART_FIFOSTS[0]) are cleared.
[20]	**HWTOIF**	**PDMA Mode RX Time-out Interrupt Flag (Read Only)** This bit is set when the RX FIFO is not empty and no activities occurred in the RX FIFO and the time-out counter equal to TOIC (UART_TOUT[7:0]). If RXTOIEN (UART_INTEN[4]) is enabled, the RX time-out interrupt will be generated. 0 = No RX time-out interrupt flag is generated in PDMA mode. 1 = RX time-out interrupt flag is generated in PDMA mode. **Note:** This bit is read only and user can read UART_DAT (RX is in active) to clear it.
[19]	**HWMODIF**	**PDMA Mode MODEM Interrupt Flag (Read Only)** This bit is set when the nCTS pin has state change (CTSDETF (UART_MODEMSTS[0] =1)). If MODEMIEN (UART_INTEN[3]) is enabled, the Modem interrupt will be generated. 0 = No Modem interrupt flag is generated in PDMA mode. 1 = Modem interrupt flag is generated in PDMA mode. **Note:** This bit is read only and reset to 0 when the bit CTSDETF (UART_MODEMSTS[0]) is cleared by writing 1 on CTSDETF (UART_MODEMSTS[0]).
[18]	**HWRLSIF**	**PDMA Mode Receive Line Status Flag (Read Only)** This bit is set when the RX receive data have parity error, frame error or break error (at least one of 3 bits, BIF (UART_FIFOSTS[6]), FEF (UART_FIFOSTS[5]) and PEF (UART_FIFOSTS[4]) is set). If RLSIEN (UART_INTEN[2]) is enabled, the RLS interrupt will be generated. 0 = No RLS interrupt flag is generated in PDMA mode. 1 = RLS interrupt flag is generated in PDMA mode. **Note 1:** In RS-485 function mode, this field include “receiver detect any address byte received address byte character (bit9 = ‘1’) bit". **Note 2:** 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. **Note 3:** 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]	**Reserved**	Reserved.
[16]	**SWBEIF**	**Single-wire Bit Error Detection Interrupt Flag**

		This bit is set when the single wire bus state not equals to UART controller TX state in Single-wire mode. 0 = No single-wire bit error detection interrupt flag is generated. 1 = Single-wire bit error detection interrupt flag is generated. **Note 1:** This bit is active when FUNCSEL (UART_FUNCSEL[2:0]) is set as UART Single-wire mode. **Note 2:** This bit can be cleared by writing "1" to it.
[15]	**LININT**	**LIN Bus Interrupt Indicator (Read Only)** This bit is set if LINIEN (UART_INTEN[8]) and LINIF (UART_INTSTS[7]) are both set to 1. 0 = No LIN Bus interrupt is generated. 1 = The LIN Bus interrupt is generated.
[14]	**WKINT**	**UART Wake-up Interrupt Indicator (Read Only)** This bit is set if WKIEN (UART_INTEN[6]) and WKIF (UART_INTSTS[6]) are both set to 1. 0 = No UART wake-up interrupt is generated. 1 = UART wake-up interrupt is generated.
[13]	**BUFERRINT**	**Buffer Error Interrupt Indicator (Read Only)** This bit is set if BUFERRIEN (UART_INTEN[5]) and BUFERRIF (UART_ INTSTS[5]) are both set to 1. 0 = No buffer error interrupt is generated. 1 = Buffer error interrupt is generated.
[12]	**RXTOINT**	**RX Time-out Interrupt Indicator (Read Only)** This bit is set if RXTOIEN (UART_INTEN[4]) and RXTOIF (UART_INTSTS[4]) are both set to 1. 0 = No RX time-out interrupt is generated. 1 = RX time-out interrupt is generated.
[11]	**MODEMINT**	**MODEM Status Interrupt Indicator (Read Only)** This bit is set if MODEMIEN (UART_INTEN[3]) and MODEMIF (UART_INTSTS[3]) are both set to 1 0 = No Modem interrupt is generated. 1 = Modem interrupt is generated..
[10]	**RLSINT**	**Receive Line Status Interrupt Indicator (Read Only)** This bit is set if RLSIEN (UART_INTEN[2]) and RLSIF (UART_INTSTS[2]) are both set to 1. 0 = No RLS interrupt is generated. 1 = RLS interrupt is generated.
[9]	**THREINT**	**Transmit Holding Register Empty Interrupt Indicator (Read Only)** This bit is set if THREIEN (UART_INTEN[1]) and THREIF (UART_INTSTS[1]) are both set to 1. 0 = No THRE interrupt is generated. 1 = THRE interrupt is generated.
[8]	**RDAINT**	**Receive Data Available Interrupt Indicator (Read Only)** This bit is set if RDAIEN (UART_INTEN[0]) and RDAIF (UART_INTSTS[0]) are both set to 1. 0 = No RDA interrupt is generated. 1 = RDA interrupt is generated.
[7]	**LINIF**	**LIN Bus Interrupt Flag** This bit is set when LIN slave header detect (SLVHDETF (UART_LINSTS[0] =1)), LIN break detect (BRKDETF (UART_LINSTS[8]=1)), bit error detect (BITEF (UART_LINSTS[9]=1)), LIN slave ID parity error (SLVIDPEF (UART_LINSTS[2] = 1)) or LIN slave header error detect (SLVHEF (UART_LINSTS[1])). If LINIEN (UART_INTEN[8]) is enabled the LIN interrupt will be generated. 0 = None of SLVHDETF, BRKDETF, BITEF, SLVIDPEF and SLVHEF is generated. 1 = At least one of SLVHDETF, BRKDETF, BITEF, SLVIDPEF and SLVHEF is generated. **Note:** This bit is cleared when SLVHDETF (UART_LINSTS[0]), BRKDETF (UART_LINSTS[8]), BITEF (UART_LINSTS[9]), SLVIDPEF (UART_LINSTS[2]) and SLVHEF (UART_LINSTS[1]) all are cleared and software writing '1' to LINIF (UART_INTSTS[7]).

[6]	**WKIF**	**UART Wake-up Interrupt Flag (Read Only)** This bit is set when TOUTWKF (UART_WKSTS[4]), RS485WKF (UART_WKSTS[3]), RFRTWKF (UART_WKSTS[2]), DATWKF (UART_WKSTS[1]) or CTSWKF (UART_WKSTS[0]) is set to 1. 0 = No UART wake-up interrupt flag is generated. 1 = UART wake-up interrupt flag is generated. **Note:** This bit is cleared if all of TOUTWKF, RS485WKF, RFRTWKF, DATWKF and CTSWKF are cleared to 0 by writing 1 to the corresponding interrupt flag.
[5]	**BUFERRIF**	**Buffer Error Interrupt Flag (Read Only)** This bit is set when the TX FIFO or RX FIFO overflows (TXOVIF (UART_FIFOSTS[24]) or RXOVIF (UART_FIFOSTS[0]) is set). When BUFERRIF (UART_INTSTS[5]) is set, the transfer is not correct. If BUFERRIEN (UART_INTEN[5]) is enabled, the buffer error interrupt will be generated. 0 = No buffer error interrupt flag is generated. 1 = Buffer error interrupt flag is generated. **Note:** This bit is cleared if both of RXOVIF (UART_FIFOSTS[0]) and TXOVIF (UART_FIFOSTS[24]) are cleared to 0 by writing 1 to RXOVIF (UART_FIFOSTS[0]) and TXOVIF (UART_FIFOSTS[24]).
[4]	**RXTOIF**	**RX Time-out Interrupt Flag (Read Only)** This bit is set when the RX FIFO is not empty 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. **Note 1:** 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. **Note 2:** 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. **Note 3:** In RS-485 function mode, this bit is read only and reset to 0 when all bits of BIF (UART_FIFOSTS[6]), FEF (UART_FIFOSTS[5]), PEF (UART_FIFOSTS[4]) and ADDRDETF (UART_FIFOSTS[3]) are cleared.
[1]	**THREIF**	**Transmit Holding Register Empty Interrupt Flag (Read Only)** This bit is set when the last data of TX FIFO is transferred to Transmitter Shift Register. If THREIEN (UART_INTEN[1]) is enabled, the THRE interrupt will be generated. 0 = No THRE interrupt flag is generated. 1 = THRE interrupt flag is generated. **Note:** This bit is read only and will be cleared when writing data into UART_DAT (TX FIFO is not empty).
[0]	**RDAIF**	**Receive Data Available Interrupt Flag (Read Only)** When the number of bytes in the RX FIFO equals the RFITL then the RDAIF (UART_INTSTS[0]) will be set. If RDAIEN (UART_INTEN[0]) is enabled, the RDA interrupt will be generated. 0 = No RDA interrupt flag is generated.

		1 = RDA interrupt flag is generated. **Note:** This bit is read only and will be cleared when the number of unread bytes of RX FIFO drops below the threshold level (RFITL (UART_FIFO[7:4]).

UART Time-out Register (UART_TOUT)

Register	Offset	R/W	Description	Reset Value
UART_TOUT **x=0,1,2,3,4,5**	UARTx_BA+0x20	R/W	UART Time-out Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**DLY**	**TX Delay Time Value** This field is used to program 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 x=0,1,2,3,4,5	UARTx_BA+0x24	R/W	UART Baud Rate Divider Register	0x0F00_0000

31	30	29	28	27	26	25	24
Reserved		BAUDM1	BAUDM0	EDIVM1			
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
BRD							
7	6	5	4	3	2	1	0
BRD							

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29]	**BAUDM1**	**BAUD Rate Mode Selection Bit 1** This bit is baud rate mode selection bit 1. UART provides three baud rate calculation modes. This bit combines with BAUDM0 (UART_BAUD[28]) to select baud rate calculation mode. The detail description is shown in Table 6.14-4. **Note:** In IrDA mode must be operated in mode 0.
[28]	**BAUDM0**	**BAUD Rate Mode Selection Bit 0** This bit is baud rate mode selection bit 0. UART provides three baud rate calculation modes. This bit combines with BAUDM1 (UART_BAUD[29]) to select baud rate calculation mode. The detail description is shown in Table 6.14-4.
[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 6.14-4.
[23:16]	**Reserved**	Reserved.
[15:0]	**BRD**	**Baud Rate Divider** The field indicates the baud rate divider. This filed is used in baud rate calculation. The detail description is shown in Table 6.14-4.

UART IrDA Control Register (UART_IRDA)

Register	Offset	R/W	Description	Reset Value
UART_IRDA **x=0,1,2,3,4,5**	UARTx_BA+0x28	R/W	UART IrDA Control Register	0x0000_0040

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

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**RXINV**	**IrDA Inverse Receive Input Signal** 0 = None inverse receiving input signal. 1 = Inverse receiving input signal. (Default) **Note 1:** Before setting this bit, TXRXDIS (UART_FUNCSEL[3]) should be set then wait for TXRXACT (UART_FIFOSTS[31]) to be cleared. When the configuration is done, clear TXRXDIS (UART_FUNCSEL[3]) to activate UART controller. **Note 2:** This bit is valid when FUNCSEL (UART_FUNCSEL[2:0]) is set as IrDA function.
[5]	**TXINV**	**IrDA Inverse Transmitting Output Signal** 0 = None inverse transmitting signal. (Default). 1 = Inverse transmitting output signal. **Note 1:** Before setting this bit, TXRXDIS (UART_FUNCSEL[3]) should be set then wait for TXRXACT (UART_FIFOSTS[31]) to be cleared. When the configuration is done, clear TXRXDIS (UART_FUNCSEL[3]) to activate UART controller. **Note 2:** This bit is valid when FUNCSEL (UART_FUNCSEL[2:0]) is set as IrDA function.
[4:2]	**Reserved**	Reserved.
[1]	**TXEN**	**IrDA Receiver/Transmitter Selection Enable Bit** 0 = IrDA Transmitter Disabled and Receiver Enabled. (Default) 1 = IrDA Transmitter Enabled and Receiver Disabled.
[0]	**Reserved**	Reserved.

UART Alternate Control/Status Register (UART_ALTCTL)

Register	Offset	R/W	Description	Reset Value
UART_ALTCTL **x=0,1,2,3,4,5**	UARTx_BA+0x2C	R/W	UART Alternate Control/Status Register	0x0000_000C

31	30	29	28	27	26	25	24
ADDRMV							
23	22	21	20	19	18	17	16
Reserved			ABRDBITS		ABRDEN	ABRIF	Reserved
15	14	13	12	11	10	9	8
ADDRDEN	Reserved				RS485AUD	RS485AAD	RS485NMM
7	6	5	4	3	2	1	0
LINTXEN	LINRXEN	Reserved		BRKFL			

Bits	Description	
[31:24]	**ADDRMV**	**Address Match Value** This field contains the RS-485 address match values. **Note:** This field is used for RS-485 auto address detection mode.
[23:21]	**Reserved**	Reserved.
[20:19]	**ABRDBITS**	**Auto-baud Rate Detect Bit Length** 00 = 1-bit time from START bit to the 1st rising edge. The input pattern shall be 0x01. 01 = 2-bit time from START bit to the 1st rising edge. The input pattern shall be 0x02. 10 = 4-bit time from START bit to the 1st rising edge. The input pattern shall be 0x08. 11 = 8-bit time from START bit to the 1st rising edge. The input pattern shall be 0x80. **Note :** The calculation of bit number includes the START bit.
[18]	**ABRDEN**	**Auto-baud Rate Detect Enable Bit** 0 = Auto-baud rate detect function Disabled. 1 = Auto-baud rate detect function Enabled. **Note :** This bit is cleared automatically after auto-baud detection is finished.
[17]	**ABRIF**	**Auto-baud Rate Interrupt Flag (Read Only)** This bit is set when auto-baud rate detection function finished or the auto-baud rate counter was overflow and if ABRIEN (UART_INTEN[18]) is set then the auto-baud rate interrupt will be generated. 0 = No auto-baud rate interrupt flag is generated. 1 = Auto-baud rate interrupt flag is generated. **Note:** This bit is read only, but it can be cleared by writing “1” to ABRDTOIF (UART_FIFOSTS[2]) and ABRDIF (UART_FIFOSTS[1]).
[16]	**Reserved**	Reserved.
[15]	**ADDRDEN**	**RS-485 Address Detection Enable Bit** This bit is used to enable RS-485 Address Detection mode. 0 = Address detection mode Disabled. 1 = Address detection mode Enabled.

		Note: This bit is used for RS-485 any operation mode.
[14:11]	**Reserved**	Reserved.
[10]	**RS485AUD**	**RS-485 Auto Direction Function (AUD)** 0 = RS-485 Auto Direction Operation function (AUD) Disabled. 1 = RS-485 Auto Direction Operation function (AUD) Enabled. **Note:** It can be active with RS-485_AAD or RS-485_NMM operation mode.
[9]	**RS485AAD**	**RS-485 Auto Address Detection Operation Mode (AAD)** 0 = RS-485 Auto Address Detection Operation mode (AAD) Disabled. 1 = RS-485 Auto Address Detection Operation mode (AAD) Enabled. **Note:** It cannot be active with RS-485_NMM operation mode.
[8]	**RS485NMM**	**RS-485 Normal Multi-drop Operation Mode (NMM)** 0 = RS-485 Normal Multi-drop Operation mode (NMM) Disabled. 1 = RS-485 Normal Multi-drop Operation mode (NMM) Enabled. **Note:** It cannot be active with RS-485_AAD operation mode.
[7]	**LINTXEN**	**LIN TX Break Mode Enable Bit** 0 = LIN TX Break mode Disabled. 1 = LIN TX Break mode Enabled. **Note:** When TX break field transfer operation finished, this bit will be cleared automatically.
[6]	**LINRXEN**	**LIN RX Enable Bit** 0 = LIN RX mode Disabled. 1 = LIN RX mode Enabled.
[5:4]	**Reserved**	Reserved.
[3:0]	**BRKFL**	**UART LIN Break Field Length** This field indicates a 4-bit LIN TX break field count. **Note 1:** This break field length is BRKFL + 1. **Note 2:** According to LIN spec, the reset value is 0xC (break field length = 13).

UART Function Select Register (UART_FUNCSEL)

Register	Offset	R/W	Description	Reset Value
UART_FUNCSEL **x=0,1,2,3,4,5**	UARTx_BA+0x30	R/W	UART Function Select Register	0x0000_0000

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

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**DGE**	**Deglitch Enable Bit** 0 = Deglitch Disabled. 1 = Deglitch Enabled. **Note:** When this bit is set to logic 1, any pulse width less than about 300 ns will be considered a glitch and will be removed in the serial data input (RX). This bit acts only on RX line and has no effect on the transmitter logic.
[5:4]	**Reserved**	Reserved.
[3]	**TXRXDIS**	**TX and RX Disable Bit** Setting this bit can disable TX and RX. 0 = TX and RX Enabled. 1 = TX and RX Disabled. **Note:** The TX and RX will not be disabled immediately when this bit is set. The TX and RX compelet current task before disable TX and RX. When TX and RX disable, the TXRXACT (UART_FIFOSTS[31]) is cleared.
[2:0]	**FUNCSEL**	**Function Select** 000 = UART function. 001 = LIN function. 010 = IrDA function. 011 = RS-485 function. 100 = UART Single-wire function. Others = Reserved.

UART LIN Control Register (UART_LINCTL)

Register	Offset	R/W	Description	Reset Value
UART_LINCTL x=0,1	UARTx_BA+0x34	R/W	UART LIN Control Register	0x000C_0000

31	30	29	28	27	26	25	24
PID							
23	22	21	20	19	18	17	16
HSEL		BSL		BRKFL			
15	14	13	12	11	10	9	8
Reserved			BITERREN	LINRXOFF	BRKDETEN	IDPEN	SENDH
7	6	5	4	3	2	1	0
Reserved			MUTE	SLVDUEN	SLVAREN	SLVHDEN	SLVEN

Bits	Description	
[31:24]	**PID**	**LIN PID Bits** This field contains the LIN frame ID value in LIN function mode, and the frame ID parity can be generated by software or hardware depends on IDPEN (UART_LINCTL[9]) = 1. If the parity generated by hardware, user fill ID0~ID5 (PID[29:24]), hardware will calculate P0 (PID[30]) and P1 (PID[31]), otherwise user must filled frame ID and parity in this field. **Note 1:** User can fill any 8-bit value to this field and the bit 24 indicates ID0 (LSB first). **Note 2:** This field can be used for LIN master mode or slave mode.
[23:22]	**HSEL**	**LIN Header Select** 00 = The LIN header includes “break field”. 01 = The LIN header includes “break field” and “sync field”. 10 = The LIN header includes “break field”, “sync field” and “frame ID field”. 11 = Reserved. **Note:** This bit is used to master mode for LIN to send header field (SENDH (UART_LINCTL[8]) = 1) or used to slave to indicate exit from mute mode condition (MUTE (UART_LINCTL[4] = 1).
[21:20]	**BSL**	**LIN Break/Sync Delimiter Length** 00 = The LIN break/sync delimiter length is 1-bit time. 01 = The LIN break/sync delimiter length is 2-bit time. 10 = The LIN break/sync delimiter length is 3-bit time. 11 = The LIN break/sync delimiter length is 4-bit time. **Note:** This bit used for LIN master to sending header field.
[19:16]	**BRKFL**	**LIN Break Field Length** This field indicates a 4-bit LIN TX break field count. **Note 1:** These registers are shadow registers of BRKFL (UART_ALTCTL[3:0]). User can read/write it by setting BRKFL (UART_ALTCTL[3:0]) or BRKFL (UART_LINCTL[19:16]). **Note 2:** This break field length is BRKFL + 1. **Note 3:** According to LIN spec, the reset value is 12 (break field length = 13).
[16:15]	**Reserved**	Reserved.

[12]	**BITERREN**	**Bit Error Detect Enable Bit** 0 = Bit error detection function Disabled. 1 = Bit error detection function Enabled. **Note:** In LIN function mode, when occur bit error, the BITEF (UART_LINSTS[9]) flag will be asserted. If the LINIEN (UART_INTEN[8]) = 1, an interrupt will be generated.
[11]	**LINRXOFF**	**LIN Receiver Disable Bit** If the receiver is enabled (LINRXOFF (UART_LINCTL[11]) = 0), all received byte data will be accepted and stored in the RX FIFO, and if the receiver is disabled (LINRXOFF (UART_LINCTL[11] = 1), all received byte data will be ignore. 0 = LIN receiver Enabled. 1 = LIN receiver Disabled. **Note:** This bit is only valid when operating in LIN function mode (FUNCSEL (UART_FUNCSEL[2:0]) = 001).
[10]	**BRKDETEN**	**LIN Break Detection Enable Bit** When detect consecutive dominant greater than 11 bits, and are followed by a delimiter character, the BRKDETF (UART_LINSTS[8]) flag is set at the end of break field. If the LINIEN (UART_INTEN[8])=1, an interrupt will be generated. 0 = LIN break detection Disabled. 1 = LIN break detection Enabled.
[9]	**IDPEN**	**LIN ID Parity Enable Bit** 0 = LIN frame ID parity Disabled. 1 = LIN frame ID parity Enabled. **Note 1:** This bit can be used for LIN master to sending header field (SENDH (UART_LINCTL[8])) = 1 and HSEL (UART_LINCTL[23:22]) = 10 or be used for enable LIN slave received frame ID parity checked. **Note 2:** This bit is only used when the operation header transmitter is in HSEL (UART_LINCTL[23:22]) = 10.
[8]	**SENDH**	**LIN TX Send Header Enable Bit** The LIN TX header can be “break field” or “break and sync field” or “break, sync and frame ID field”, it is depend on setting HSEL (UART_LINCTL[23:22]). 0 = Send LIN TX header Disabled. 1 = Send LIN TX header Enabled. **Note 1:** This bit is shadow bit of LINTXEN (UART_ALTCTL[7]); user can read/write it by setting LINTXEN (UART_ALTCTL[7]) or SENDH (UART_LINCTL[8]). **Note 2:** When transmitter header field (it may be “break” or “break + sync” or “break + sync + frame ID” selected by HSEL (UART_LINCTL[23:22]) field) transfer operation finished, this bit will be cleared automatically.
[7:5]	**Reserved**	Reserved.
[4]	**MUTE**	**LIN Mute Mode Enable Bit** 0 = LIN mute mode Disabled. 1 = LIN mute mode Enabled. **Note:** The exit from mute mode condition and each control and interactions of this field are explained in 6.14.5.10 (LIN slave mode).
[3]	**SLVDUEN**	**LIN Slave Divider Update Method Enable Bit** 0 = UART_BAUD updated is written by software (if no automatic resynchronization update occurs at the same time). 1 = UART_BAUD is updated at the next received character. User must set the bit before checksum reception. **Note 1:** This bit only is valid in LIN slave mode (SLVEN (UART_LINCTL[0]) = 1). **Note 2:** This bit used for LIN Slave Automatic Resynchronization mode. (for Non-Automatic Resynchronization mode, this bit should be kept cleared) **Note 3:** The control and interactions of this field are explained in 6.14.5.10 (Slave mode with automatic

		resynchronization).
[2]	**SLVAREN**	**LIN Slave Automatic Resynchronization Mode Enable Bit** 0 = LIN automatic resynchronization Disabled. 1 = LIN automatic resynchronization Enabled. **Note 1:** This bit only is valid in LIN slave mode (SLVEN (UART_LINCTL[0]) = 1). **Note 2:** When operation in Automatic Resynchronization mode, the baud rate setting must be mode2 (BAUDM1 (UART_BAUD[29]) and BAUDM0 (UART_BAUD[28]) must be 1). **Note 3:** The control and interactions of this field are explained in 6.14.5.10 (Slave mode with automatic resynchronization).
[1]	**SLVHDEN**	**LIN Slave Header Detection Enable Bit** 0 = LIN slave header detection Disabled. 1 = LIN slave header detection Enabled. **Note 1:** This bit is only valid in LIN slave mode (SLVEN (UART_LINCTL[0]) = 1). **Note 2:** In LIN function mode, when detect header field (break + sync + frame ID), SLVHDETF (UART_LINSTS[0]) flag will be asserted. If the LINIEN (UART_INTEN[8]) = 1, an interrupt will be generated.
[0]	**SLVEN**	**LIN Slave Mode Enable Bit** 0 = LIN slave mode Disabled. 1 = LIN slave mode Enabled.

UART LIN Status Register (UART_LINSTS)

Register	Offset	R/W	Description	Reset Value
UART_LINSTS **x=0,1**	UARTx_BA+0x38	R/W	UART LIN Status Register	0x0000_0000

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

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**BITEF**	**Bit Error Detect Status Flag** At TX transfer state, hardware will monitor the bus state, if the input pin (UART_RXD) state not equals to the output pin (UART_TXD) state, BITEF (UART_LINSTS[9]) will be set. When occur bit error, if the LINIEN (UART_INTEN[8]) = 1, an interrupt will be generated. 0 = Bit error not detected. 1 = Bit error detected. **Note 1:** This bit can be cleared by writing 1 to it. **Note 2:** This bit is only valid when enable bit error detection function (BITERREN (UART_LINCTL[12]) = 1).
[8]	**BRKDETF**	**LIN Break Detection Flag** This bit is set by hardware when a break is detected and be cleared by writing 1 to it through software. 0 = LIN break not detected. 1 = LIN break detected. **Note 1:** This bit can be cleared by writing 1 to it. **Note 2:** This bit is only valid when LIN break detection function is enabled (BRKDETEN (UART_LINCTL[10]) =1).
[7:4]	**Reserved**	Reserved.
[3]	**SLVSYNCF**	**LIN Slave Sync Field** This bit indicates that the LIN sync field is being analyzed in Automatic Resynchronization mode. When the receiver header have some error been detect, user must reset the internal circuit to re-search new frame header by writing 1 to this bit. 0 = The current character is not at LIN sync state. 1 = The current character is at LIN sync state. **Note 1:** This bit is only valid in LIN Slave mode (SLVEN (UART_LINCTL[0]) = 1). **Note 2:** This bit can be cleared by writing 1 to it. **Note 3:** When writing 1 to it, hardware will reload the initial baud rate and re-search a new frame header.

[2]	**SLVIDPEF**	**LIN Slave ID Parity Error Flag** This bit is set by hardware when receipted frame ID parity is not correct. 0 = No active. 1 = Receipted frame ID parity is not correct. **Note 1:** This bit can be cleared by writing 1 to it. **Note 2:** This bit is only valid in LIN slave mode (SLVEN (UART_LINCTL[0])= 1) and enable LIN frame ID parity check function IDPEN (UART_LINCTL[9]).
[1]	**SLVHEF**	**LIN Slave Header Error Flag** This bit is set by hardware when a LIN header error is detected in LIN slave mode and be cleared by writing 1 to it. The header errors include “break delimiter is too short (less than 0.5 bit time)”, “frame error in sync field or Identifier field”, “sync field data is not 0x55 in Non-Automatic Resynchronization mode”, “sync field deviation error with Automatic Resynchronization mode”, “sync field measure time-out with Automatic Resynchronization mode” and “LIN header reception time-out”. 0 = LIN header error not detected. 1 = LIN header error detected. **Note 1:** This bit can be cleared by writing 1 to it. **Note 2:** This bit is only valid when UART is operated in LIN slave mode (SLVEN (UART_LINCTL[0]) = 1) and enables LIN slave header detection function (SLVHDEN (UART_LINCTL[1])).
[0]	**SLVHDETF**	**LIN Slave Header Detection Flag** This bit is set by hardware when a LIN header is detected in LIN slave mode and be cleared by writing 1 to it. 0 = LIN header not detected. 1 = LIN header detected (break + sync + frame ID). **Note 1:** This bit can be cleared by writing 1 to it. **Note 2:** This bit is only valid in LIN slave mode (SLVEN (UART_LINCTL[0]) = 1) and enable LIN slave header detection function (SLVHDEN (UART_LINCTL[1])). **Note 3:** When enable ID parity check IDPEN (UART_LINCTL[9]), if hardware detect complete header (“break + sync + frame ID”), the SLVHDETF will be set whether the frame ID correct or not.

UART Baud Rate Compensation Register (UART_BRCOMP)

Register	Offset	R/W	Description	Reset Value
UART_BRCOMP x=0,1	UARTx_BA+0x3C	R/W	UART Baud Rate Compensation Register	0x0000_0000

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

Bits	Description	
[31]	**BRCOMPDEC**	**Baud Rate Compensation Decrease** 0 = Positive (increase one module clock) compensation for each compensated bit. 1 = Negative (decrease one module clock) compensation for each compensated bit.
[30:9]	**Reserved**	Reserved.
[8:0]	**BRCOMP**	**Baud Rate Compensation Patten** These 9-bits are used to define the relative bit is compensated or not. BRCOMP[7:0] is used to define the compensation of DAT (UART_DAT[7:0]) and BRCOM[8] is used to define PARITY (UART_DAT[8]).

UART Wake-up Control Register (UART_WKCTL)

Register	Offset	R/W	Description	Reset Value
UART_WKCTL **x=0,1,2,3,4,5**	UARTx_BA+0x40	R/W	UART Wake-up Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**WKTOUTEN**	**WKRS485EN**	**WKRFRTEN**	**WKDATEN**	**WKCTSEN**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**WKTOUTEN**	**Received Data FIFO Reached Threshold Time-out Wake-up Enable Bit** 0 = Received Data FIFO reached threshold time-out wake-up system function Disabled. 1 = Received Data FIFO reached threshold time-out wake-up system function Enabled. **Note 1:** When the system is in Power-down mode, Received Data FIFO reached threshold time-out will wake up system from Power-down mode. **Note 2:** It is suggested 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. **Note 1:** When the system is in.Power-down mode, RS-485 Address Match will wake-up system from Power-down mode. **Note 2:** This bit is used for RS-485 Auto Address Detection (AAD) mode in RS-485 function mode and ADDRDEN (UART_ALTCTL[15]) is set to 1.
[2]	**WKRFRTEN**	**Received Data FIFO Reached Threshold Wake-up Enable Bit** 0 = Received Data FIFO reached threshold wake-up system function Disabled. 1 = Received Data FIFO reached threshold wake-up system function Enabled. **Note:** When the system is in Power-down mode, Received Data FIFO reached threshold will wake-up system from Power-down mode.
[1]	**WKDATEN**	**Incoming Data Wake-up Enable Bit** 0 = Incoming data wake-up system function Disabled. 1 = Incoming data wake-up system function Enabled. **Note:**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. **Note:**When the system is in Power-down mode, an external.nCTS change will wake up system from

		Power-down mode.

UART Wake-up Status Register (UART_WKSTS)

Register	Offset	R/W	Description	Reset Value
UART_WKSTS **x=0,1,2,3,4,5**	UARTx_BA+0x44	R/W	UART Wake-up Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			TOUTWKF	RS485WKF	RFRTWKF	DATWKF	CTSWKF

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**TOUTWKF**	**Received Data FIFO Threshold Time-out Wake-up Flag** This bit is set if chip wake-up from power-down state by Received Data FIFO Threshold Time-out wake-up. 0 = Chip stays in power-down state. 1 = Chip wake-up from power-down state by Received Data FIFO reached threshold time-out. **Note 1:** If WKTOUTEN (UART_WKCTL[4]) is enabled, the Received Data FIFO reached threshold time-out wake-up cause this bit is set to '1'. **Note 2:** 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. **Note 1:** If WKRS485EN (UART_WKCTL[3]) is enabled, the RS-485 Address Match (AAD mode) wake-up cause this bit is set to '1'. **Note 2:** This bit can be cleared by writing '1' to it.
[2]	**RFRTWKF**	**Received Data FIFO Reached Threshold Wake-up Flag** This bit is set if chip wake-up from power-down state by Received Data FIFO reached threshold wake-up. 0 = Chip stays in power-down state. 1 = Chip wake-up from power-down state by Received Data FIFO Reached Threshold wake-up. **Note 1:** If WKRFRTEN (UART_WKCTL[2]) is enabled, the Received Data FIFO Reached Threshold wake-up cause this bit is set to '1'. **Note 2:** This bit can be cleared by writing '1' to it.
[1]	**DATWKF**	**Incoming Data Wake-up Flag** This bit is set if chip wake-up from power-down state by data wake-up. 0 = Chip stays in power-down state. 1 = Chip wake-up from power-down state by Incoming Data wake-up.

		Note 1: If WKDATEN (UART_WKCTL[1]) is enabled, the Incoming Data wake-up cause this bit is set to '1'. **Note 2:** 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. **Note 1:** If WKCTSEN (UART_WKCTL[0]) is enabled, the nCTS wake-up cause this bit is set to '1'. **Note 2:** 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 x=0,1,2,3,4,5	UARTx_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.
[15:0]	**STCOMP**	**START Bit Compensation Value** These bits field indicate how many clock cycle selected by UART_CLK do the UART controller can get the 1st bit (START bit) when the device is woken up from Power-down mode. **Note:** It is valid only when WKDATEN (UART_WKCTL[1]) is set.

6.15 Serial Peripheral Interface (SPI)

6.15.1 Overview

The Serial Peripheral Interface (SPI) applies to synchronous serial data communication and allows full duplex transfer. Devices communicate in Master/Slave mode with the 4-wire bi-direction interface. The chip contains up to two sets of SPI controllers performing a serial-to-parallel conversion on data received from a peripheral device, and a parallel-to-serial conversion on data transmitted to a peripheral device. Each SPI controller can be configured as a master or a slave device and supports the PDMA function to access the data buffer. Each SPI controller also supports I^2S mode to connect external audio CODEC.

6.15.2 Features

- SPI Mode
  - Up to two sets of SPI controllers
  - Supports Master or Slave mode operation
  - Master mode up to 24 MHz
  - Slave mode up to 24 MHz when SPI master device supports adjustment function of RX data sampling clock
  - Slave mode up to 18 MHz when SPI master device does not support adjustment function of RX data sampling clock
  - Configurable bit length of a transaction word from 4 to 32-bit
  - Provides separate 4-level of 32-bit (or 8-level of 16-bit) transmit and receive FIFO buffers which depends on SPI setting of data width
  - 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
  - Supports one data channel half-duplex transfer
  - Supports receive-only mode
- I^2S Mode
  - 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

6.15.3 Block Diagram

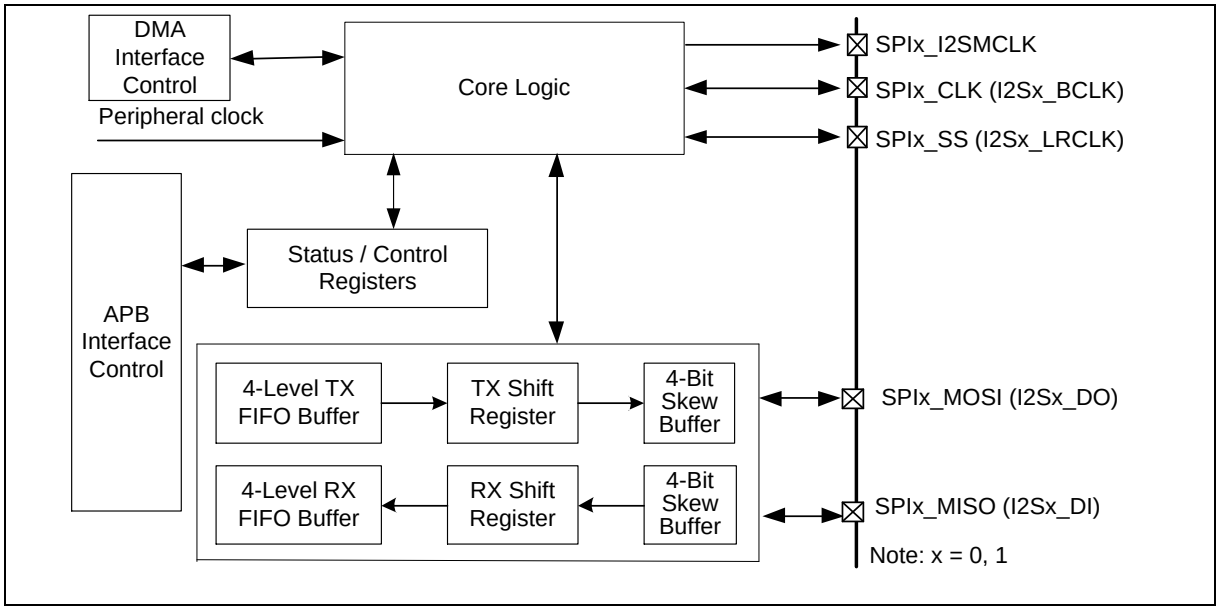


Figure 6.15-1 SPI Block Diagram

TX FIFO Buffer:

The transmit FIFO buffer is a 4-level depth, 32-bit wide, first-in, first-out register buffer. The data can be written to the transmit FIFO buffer in advance through software by writing the SPIx_TX register. In SPI mode, the transmit FIFO will be configured as 8-level while data length is set as 4~16 bits.

RX FIFO Buffer:

The receive FIFO buffer is also a 4-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 SPIx_RX register by software. In SPI mode, the receive FIFO will be configured as 8-level while data length is set as 4~16 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.

6.15.4 Basic Configuration

6.15.4.1 SPI0 Basic Configuration

- Clock Source Configuration
  - Select the source of SPI0 peripheral clock on SPI0SEL (CLK_CLKSEL2[5:4]).
  - Enable SPI0 peripheral clock in SPI0CKEN (CLK_APBCLK0[13]).

- Reset Configuration
  - Reset SPI0 controller in SPI0RST (SYS_IPRST1[13]).
- SPI1 Basic Configuration
- Clock Source Configuration
  - Select the source of SPI1 peripheral clock on SPI1SEL (CLK_CLKSEL2[7:6]).
  - Enable SPI1 peripheral clock in SPI1CKEN (CLK_APBCLK0[14]).
- Reset Configuration
  - Reset SPI1 controller in SPI1RST (SYS_IPRST1[14]).
- SPI/I²S (SPI0~SPI1) Interface Controller Pin description is shown as follows:

Pin	SPI Mode	I²S Mode
SPIx_SS	SPI slave selection pin	I²S left/right channel synchronization clock pin (I2Sx_LRCLK)
SPIx_CLK	SPI clock pin	I²S bit clock pin (I2Sx_BCLK)
SPIx_MISO	SPI master input or slave output pin	I²S data input pin (I2Sx_DI)
SPIx_MOSI	SPI master output or slave input pin	I²S data output pin (I2Sx_DO)
SPIx_I2SMCLK	Not available	I²S Master clock output pin

Table 6.15-1 SPI/I²S Interface Controller Pin Description (SPI0~SPI1)

6.15.5 Functional Description

6.15.5.1 Terminology

SPI Peripheral Clock and SPI Bus Clock

The SPI controller needs the peripheral clock to drive the SPI logic unit to perform the data transfer. The peripheral clock rate is determined by the settings of clock divisor (SPIx_CLKDIV) and the clock source which can be HXT, PLL, PCLK or HIRC. SPIxSEL of CLK_CLKSEL2 register determines the clock source of the peripheral clock. The DIVIDER (SPIx_CLKDIV[8:0]) setting determines the divisor of the clock rate calculation.

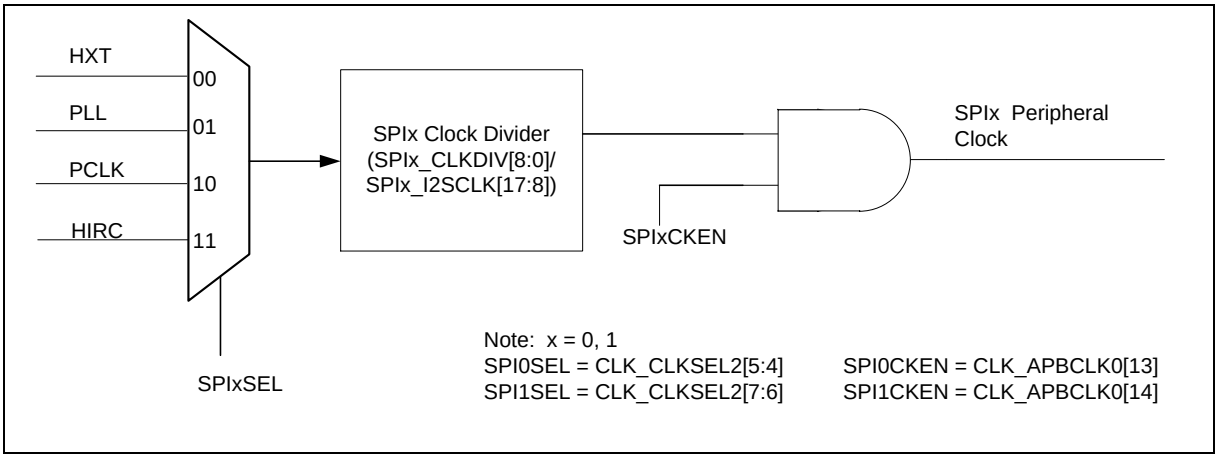


Figure 6.15-2 SPI Peripheral Clock

In Master mode, the frequency of the SPI bus clock is equal to the peripheral clock rate. In general, the SPI bus clock is denoted as SPI clock. In Slave mode, the SPI bus clock is provided by a master device. The frequency of SPI peripheral clock cannot be faster than the system clock rate regardless

of Master or Slave mode. If the clock source of peripheral clock is not system clock, the frequency of SPI peripheral clock shall be slower than the system clock frequency regardless of Master or Slave mode.

In I^2S mode, the peripheral clock rate is equal to I^2S bit clock rate determined by SPIx_I2SCLK register.

Master/Slave mode

The SPI controllers can be set as Master or Slave mode by setting the SLAVE (SPIx_CTL[18]) to communicate with the off-chip SPI slave or master device. The HALFDPX (SPIx_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.

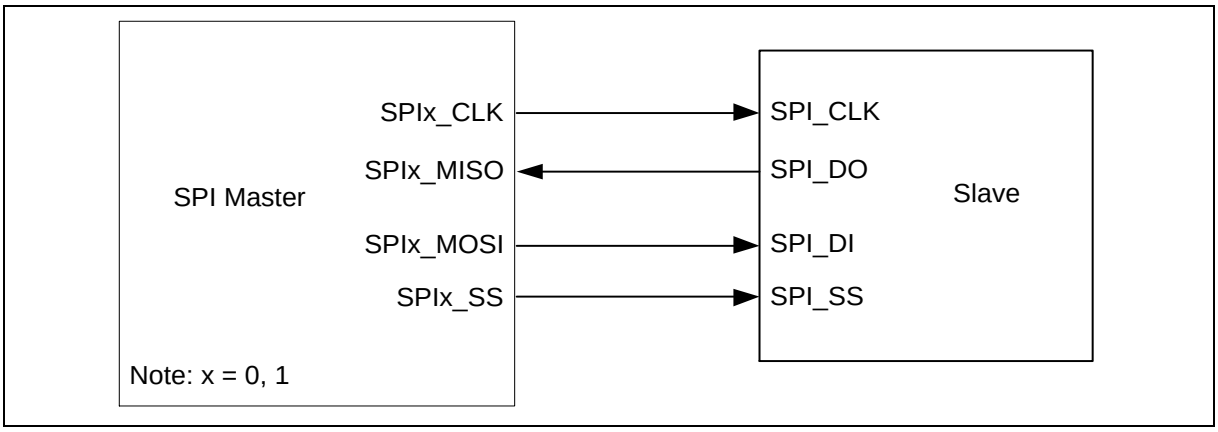


Figure 6.15-3 SPI Full-Duplex Master Mode Application Block Diagram

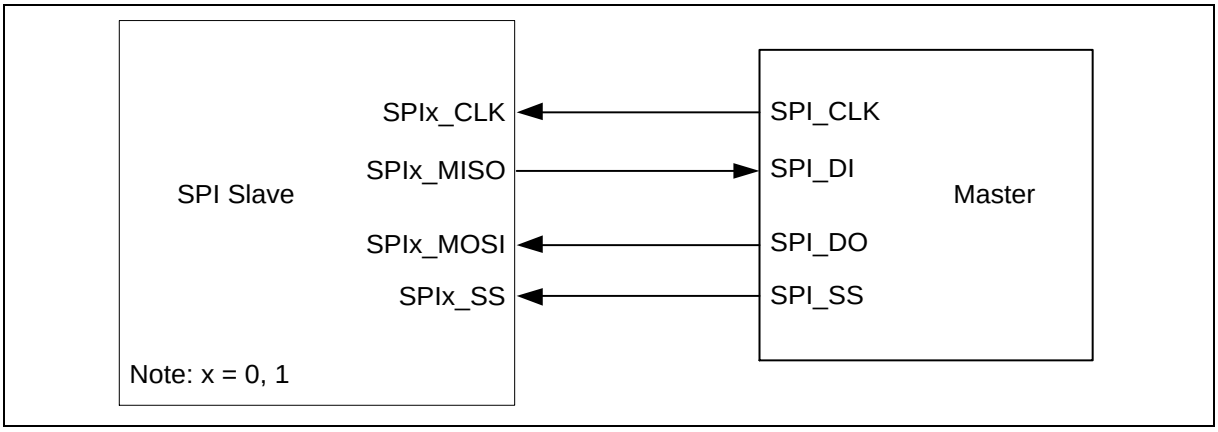


Figure 6.15-4 SPI Full-Duplex Slave Mode Application Block Diagram

Slave Selection

In Master mode, the SPI controller can drive off-chip slave device through the slave select output pin SPIx_SS. In Slave mode, the off-chip master device drives the slave selection signal from the SPIx_SS input port to this SPI controller. The duration between the slave select active edge and the first SPI clock input shall over 3 SPI peripheral clock cycles of slave.

In Master/Slave mode, the active state of slave selection signal can be programmed to low or high active in SSACTPOL (SPIx_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.

In Slave mode, the SPIx_MISO signal type is controlled by the SPIx_SS pin. If the SPIx_SS pin is inactive state, the SPIx_MISO is set to tri-state to avoid interference with other Slave devices. If the SPIx_SS pin is active state, the SPIx_MISO is set to output mode for data transfer.

Timing Condition

The CLKPOL (SPIx_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 (SPIx_CTL[2]) defines the data transmitted out either on negative edge or on positive edge of SPI clock. RXNEG (SPIx_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 (SPIx_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 (SPIx_CTL[12:8]), the unit transfer interrupt flag UNITIF (SPIx_STATUS[1]) will be set to 1.

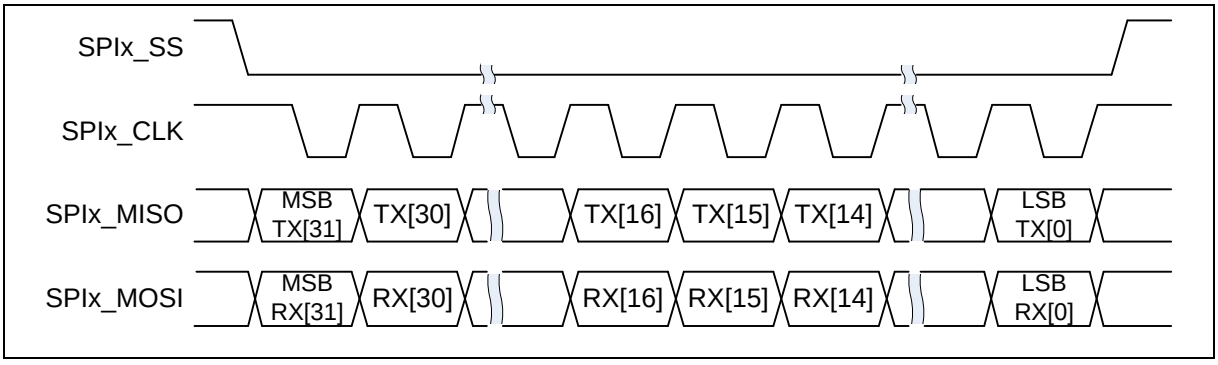


Figure 6.15-5 32-bit in One Transaction

LSB/MSB First

LSB (SPIx_CTL[13]) defines the bit transfer sequence in a transaction. If the LSB (SPIx_CTL[13]) is set to 1, the transfer sequence is LSB first. The bit 0 will be transferred firstly. If the LSB (SPIx_CTL[13]) is cleared to 0, the transfer sequence is MSB first.

Suspend Interval

SUSPITV (SPIx_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).

6.15.5.2 Automatic Slave Selection

In Master mode, if AUTOSS (SPIx_SSCTL[3]) is set, the slave selection signal will be generated automatically and output to the SPIx_SS pin according to whether SS (SPIx_SSCTL[0]) is enabled or not. The slave selection signal will be set to active state by the SPI controller when the SPI data transfer is started by writing to FIFO. It will be set to inactive state when SPI bus is idle. If SPI bus is not idle, i.e. TX FIFO, TX shift register or TX skew buffer is not empty, the slave selection signal will be set to inactive state between transactions if the value of SUSPITV (SPIx_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 (SPIx_SSCTL[2]).

The duration between the slave selection signal active edge and the first SPI bus clock edge is 1 SPI bus clock cycle and the duration between the last SPI bus clock and the slave selection signal inactive edge is 1.5 SPI bus clock cycle.

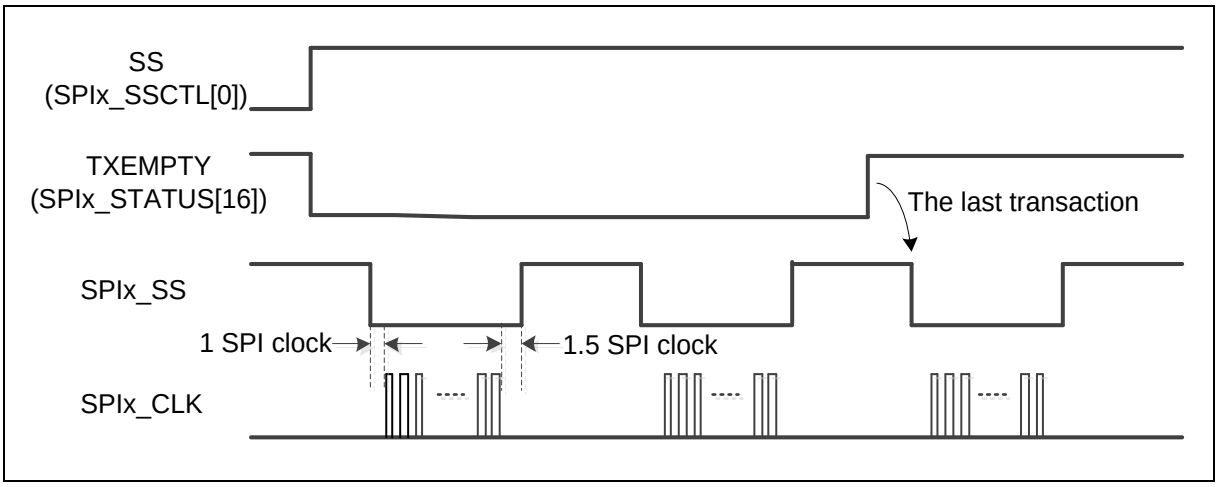


Figure 6.15-6 Automatic Slave Selection (SSACTPOL = 0, SUSPITV > 0x2)

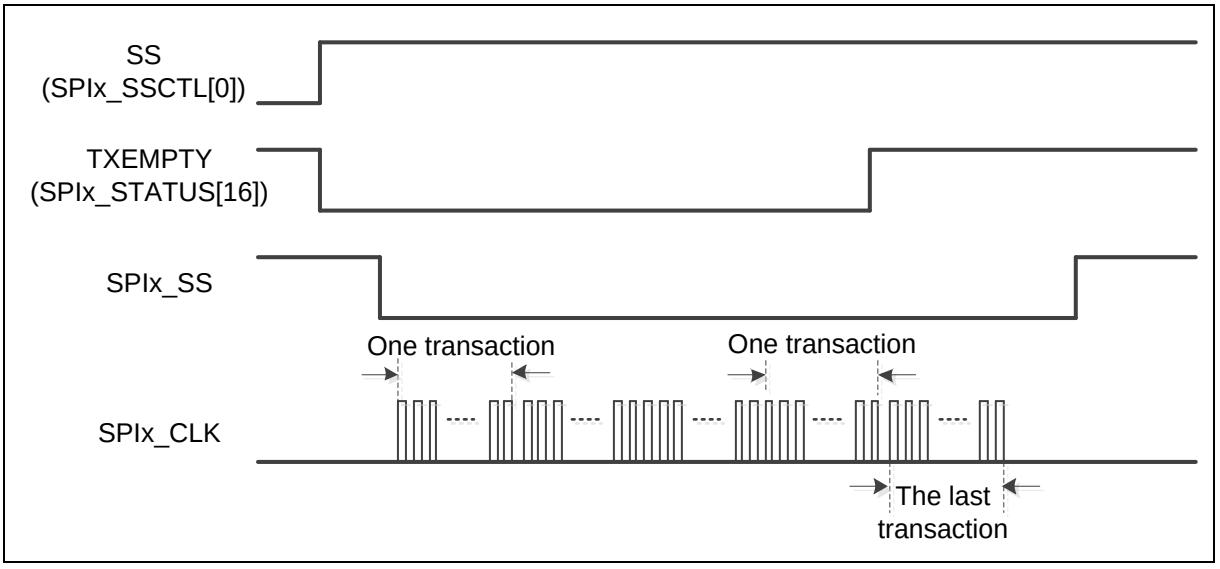


Figure 6.15-7 Automatic Slave Selection (SSACTPOL = 0, SUSPITV < 0x3)

6.15.5.3 Byte Reorder and Suspend Function

When the transfer is set as MSB first (LSB = 0) and the REORDER (SPIx_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.

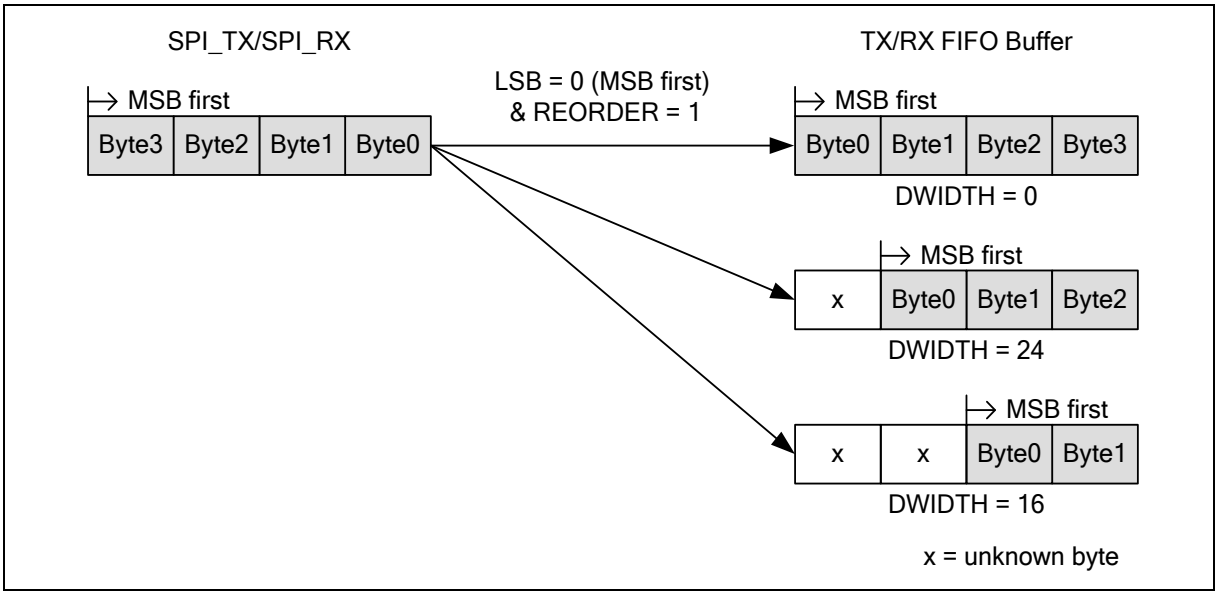


Figure 6.15-8 Byte Reorder Function

In Master mode, if REORDER (SPIx_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 (SPIx_CTL[7:4]).

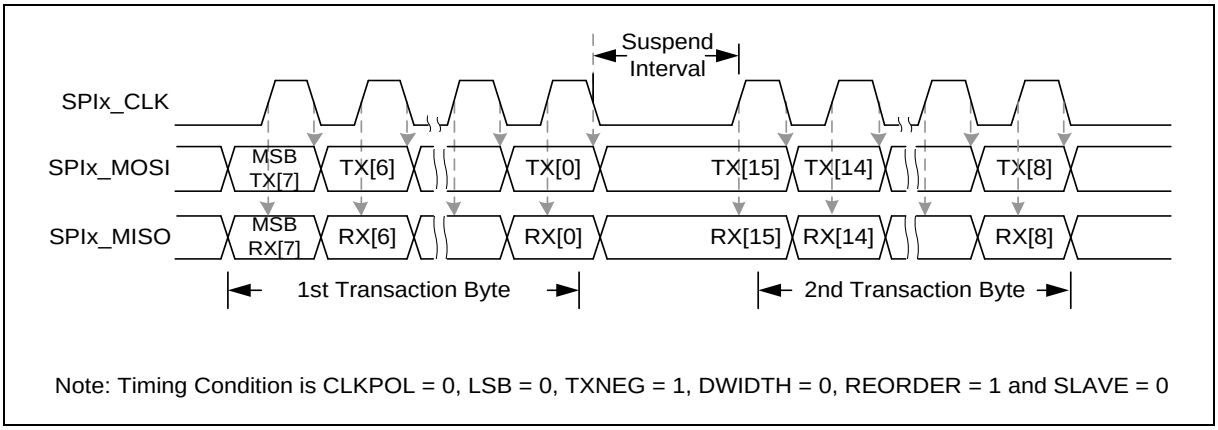


Figure 6.15-9 Timing Waveform for Byte Suspend

6.15.5.4 Half-Duplex Communication

The SPI controller can communicate in half-duplex mode by setting HALFDPX (SPIx_CTL[14]) bit. In half-duplex mode, there is only one data line for receiving or transmitting data direction which is defined by DATDIR (SPIx_CTL[20]). In half-duplex configuration, the SPIx_MISO pin is free for other applications and it can be configured as GPIO. Enabling or disabling the control bit HALFDPX (SPIx_CTL[14]) will produce TXFBCLR (SPIx_FIFOCTL[9]) and RXFBCLR (SPIx_FIFOCTL[8]) at the same time automatically.

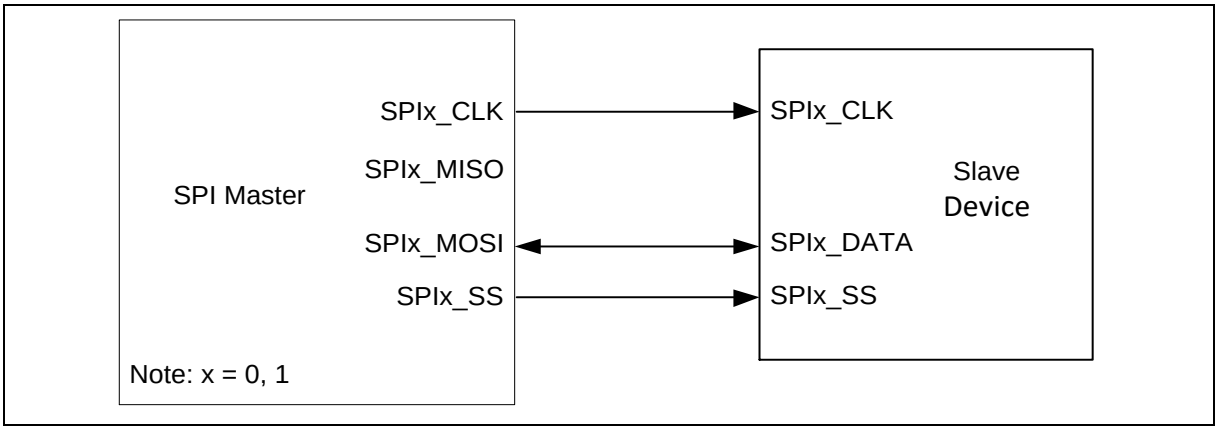


Figure 6.15-10 SPI Half-Duplex Master Mode Application Block Diagram

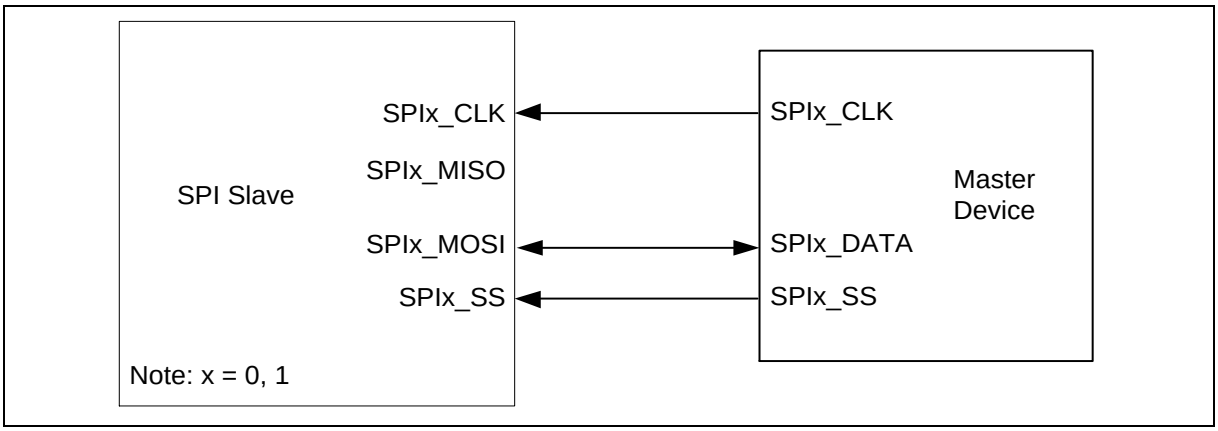


Figure 6.15-11 SPI Half-Duplex Slave Mode Application Block Diagram

6.15.5.5 Receive-Only Mode

In SPI Master device, it can communicate in receive-only mode by setting RXONLY (SPIx_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 (SPIx_SSCTL[3]) is enabled in receive-only mode, SPI Master will keep activating the slave select signal.

The remaining SPIx_MOSI pin of SPI Master device is not used for communication and can be configured as GPIO. The status BUSY (SPIx_STATUS[0]) will be asserted in receive-only mode due to the generation of SPI bus clock. Entering this mode will produce the TXFBCLR (SPIx_FIFOCTL[9]) and RXFBCLR (SPIx_FIFOCTL[8]) at the same time automatically. When user enables this mode, the output SPI bus clock will be sent out after 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.

When user sets RXONLY (SPIx_CTL[15]) enabled, SPI RX data with data bit width of DWIDTH (SPIx_CTL[12:8]) will be received into RX FIFO and SPI clock will be sent to SPI slave device until RX FIFO is full.

For data bit width of 8~16 bits, the SPI master will send SPI output clock to SPI slave and receive RX data when RX FIFO counter RXCNT (SPIx_STATUS[27:24]) is less than or equal to 6.

For data bit width of 17~32 bits, the SPI master will send SPI output clock to SPI slave and receive RX data when RX FIFO counter RXCNT (SPIx_STATUS[27:24]) is less than or equal to 2.

6.15.5.6 Slave 3-Wire Mode

When SLV3WIRE (SPIx_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 (SPIx_SSCTL[4]) only takes effect in Slave mode. Only three pins, SPIx_CLK, SPIx_MISO, and SPIx_MOSI, are required to communicate with a SPI master. The SPIx_SS pin can be configured as a GPIO. When the SLV3WIRE (SPIx_SSCTL[4]) is set to 1, the SPI slave will be ready to transmit/receive data after the SPIEN (SPIx_CTL[0]) is set to 1.

6.15.5.7 PDMA Transfer Function

SPI controller supports PDMA transfer function.

When TXPDMAEN (SPIx_PDMACTL[0]) is set to 1, the controller will issue request to PDMA controller to start the PDMA transmission process automatically.

When RXPDMAEN (SPIx_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.

6.15.5.8 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 (SPIx_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 (SPIx_STATUS[16]) will be set to 1. Note that the TXEMPTY (SPIx_STATUS[16]) flag is set to 1 while the last transaction is still in progress. In Master mode, the BUSY (SPIx_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 (SPIx_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 (SPIx_STATUS[8]) and RXFULL (SPIx_STATUS[9]), to indicate the current status of RX FIFO buffer.

The transmitting and receiving threshold can be configured by setting TXTH (SPIx_FIFOCTL[30:28]) and RXTH (SPIx_FIFOCTL[26:24]). When the count of valid data stored in transmit FIFO buffer is less than or equal to TXTH (SPIx_FIFOCTL[30:28]) setting, TXTHIF (SPIx_STATUS[18]) will be set to 1. When the count of valid data stored in receive FIFO buffer is larger than RXTH (SPIx_FIFOCTL[26:24]) setting, RXTHIF (SPIx_STATUS[10]) will be set to 1.

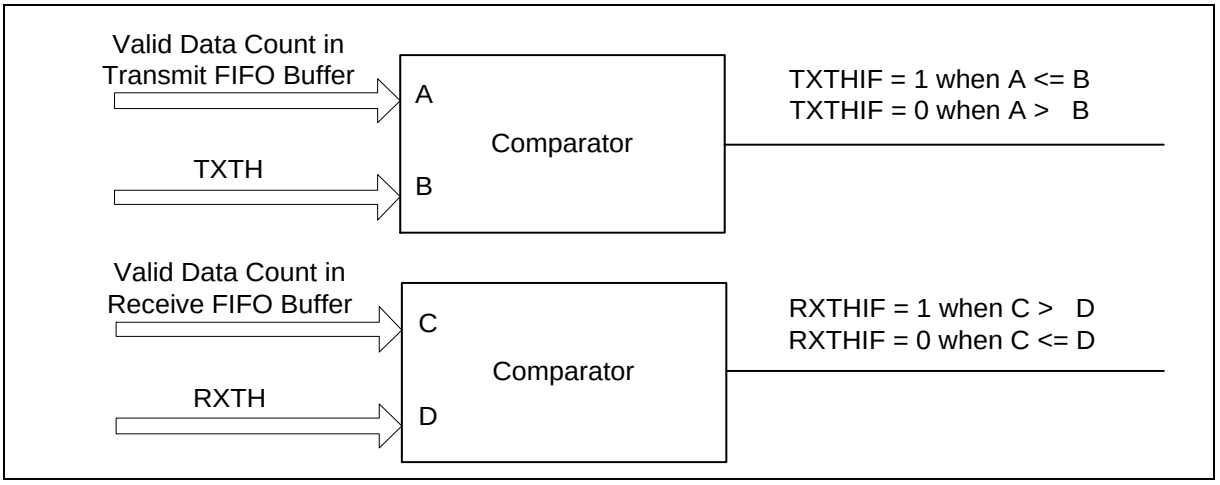


Figure 6.15-12 FIFO Threshold Comparator

In Master mode, when the first datum is written to the SPIx_TX register, the TXEMPTY flag (SPIx_STATUS[16]) will be cleared to 0. The transmission will start after 1 PCLK clock cycles and 6 peripheral clock cycles. User can write the next data into SPIx_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 (SPIx_CTL[7:4]). If the SUSPITV (SPIx_CTL[7:4]) equals 0, SPI controller can perform continuous transfer. User can write data into SPIx_TX register as long as the TXFULL (SPIx_STATUS[17]) is 0.

The example 1 of Figure 6.15-13 indicates the updated condition of TXEMPTY (SPIx_STATUS[16]) and the relationship among the FIFO buffer, shift register and the skew buffer for 4~16 bits of data length. The TXEMPTY (SPIx_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 (SPIx_STATUS[16]) will be 1. The Data0 in shift register will be shifted into skew buffer by bit for transmission until the transfer is done.

The Example 2 of Figure 6.15-13 indicates the updated condition of TXFULL (SPIx_STATUS[17]) when there are 8 data in the FIFO buffer and the next data of Data9 is not written into the FIFO buffer when the TXFULL = 1.

The example 1 of Figure 6.15-14 indicates the updated condition of TXEMPTY (SPIx_STATUS[16]) and the relationship among the FIFO buffer, shift register and the skew buffer for 17~32 bits of data length.

The Example 2 of Figure 6.15-14 indicates the updated condition of TXFULL (SPIx_STATUS[17]) when there are 4 data in the FIFO buffer and the next data of Data5 is not written into the FIFO buffer when the TXFULL = 1.

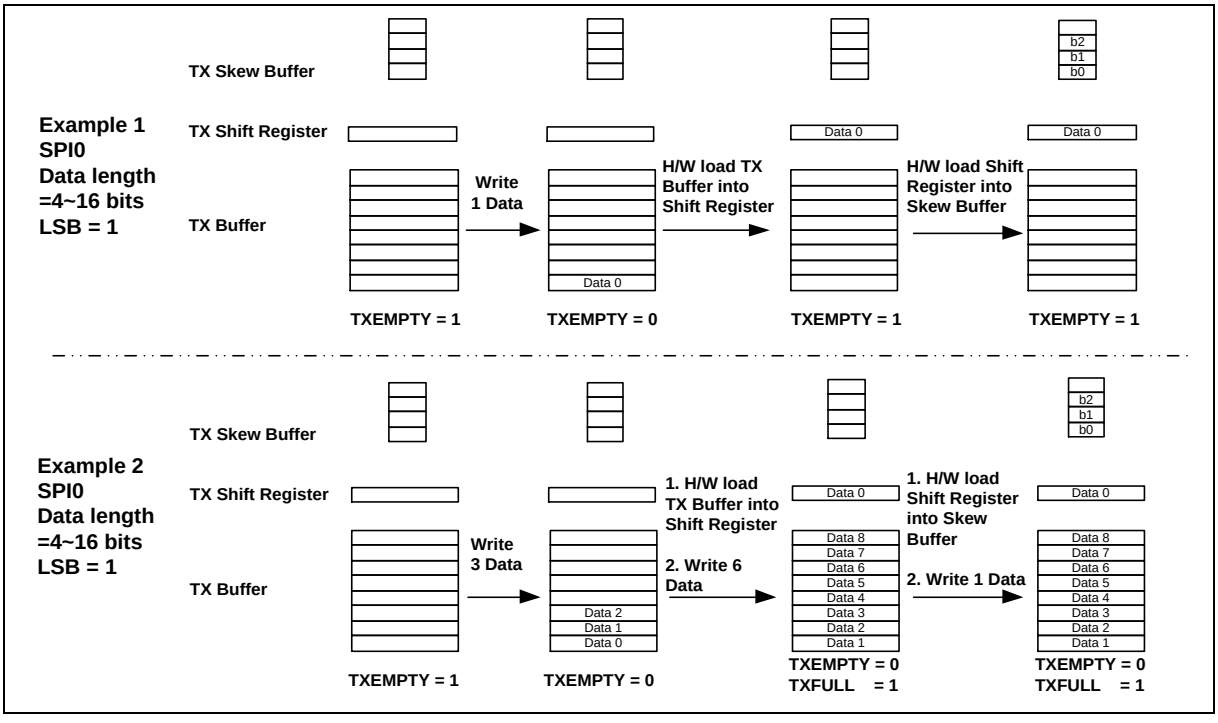


Figure 6.15-13 Transmit FIFO Buffer Example for 4~16 bits of data length

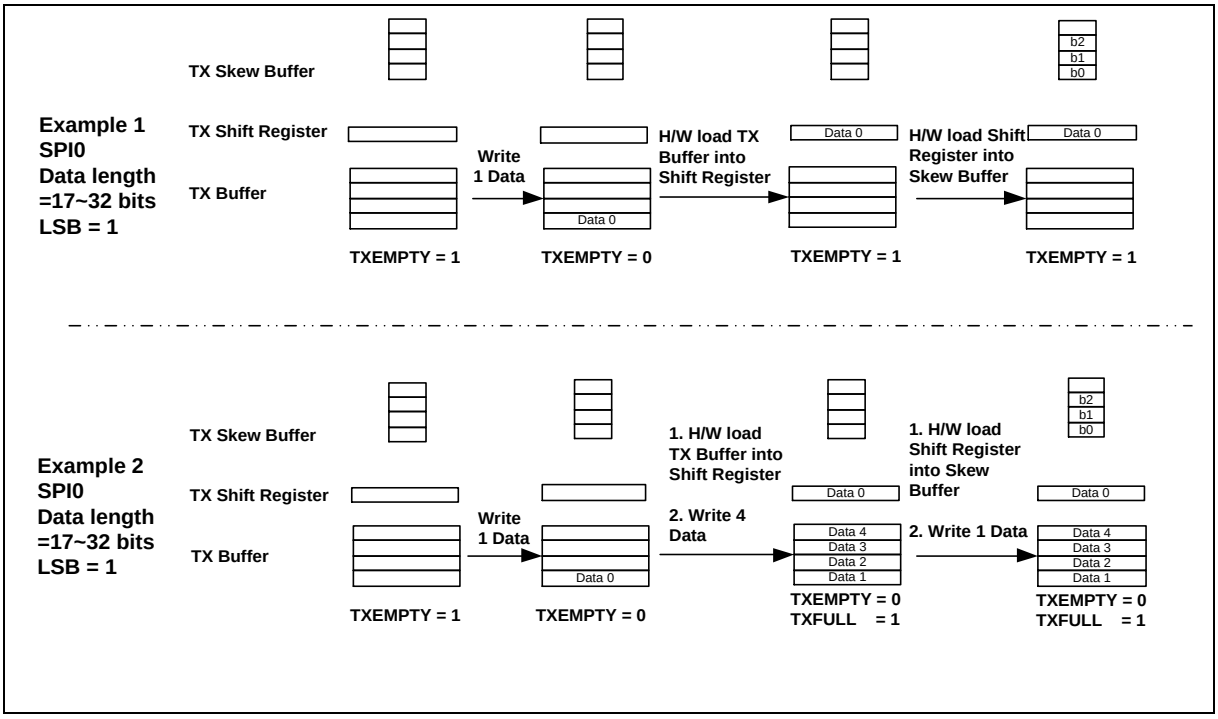


Figure 6.15-14 Transmit FIFO Buffer Example for 17~32 Bits of Data Length

The subsequent transactions will be triggered automatically if the transmitted data are updated in time. If the SPIx_TX register is not updated after all data transfer are done, the transfer will stop.

In Master mode, during receiving operation, the serial data are received from SPIx_MISO 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 (SPIx_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 (SPIx_CTL[12:8]).

The RXEMPTY (SPIx_STATUS[8]) will be cleared to 0 while the receive FIFO buffer contains unread data (see the Example 1 of Receive FIFO Buffer Example in Figure 6.15-15). The received data can be read by software from SPIx_RX register as long as the RXEMPTY (SPIx_STATUS[8]) is 0. If the receive FIFO buffer contains 8 unread data, the RXFULL (SPIx_STATUS[9]) will be set to 1 (see the Example 2 of Receive FIFO Buffer Example in Figure 6.15-15).

The RXEMPTY (SPIx_STATUS[8]) will be cleared to 0 while the receive FIFO buffer contains unread data (see the Example 1 of Receive FIFO Buffer Example in Figure 6.15-16). The received data can be read by software from SPIx_RX register as long as the RXEMPTY (SPIx_STATUS[8]) is 0. If the receive FIFO buffer contains 4 unread data, the RXFULL (SPIx_STATUS[9]) will be set to 1 (see the Example 2 of Receive FIFO Buffer Example in Figure 6.15-16).

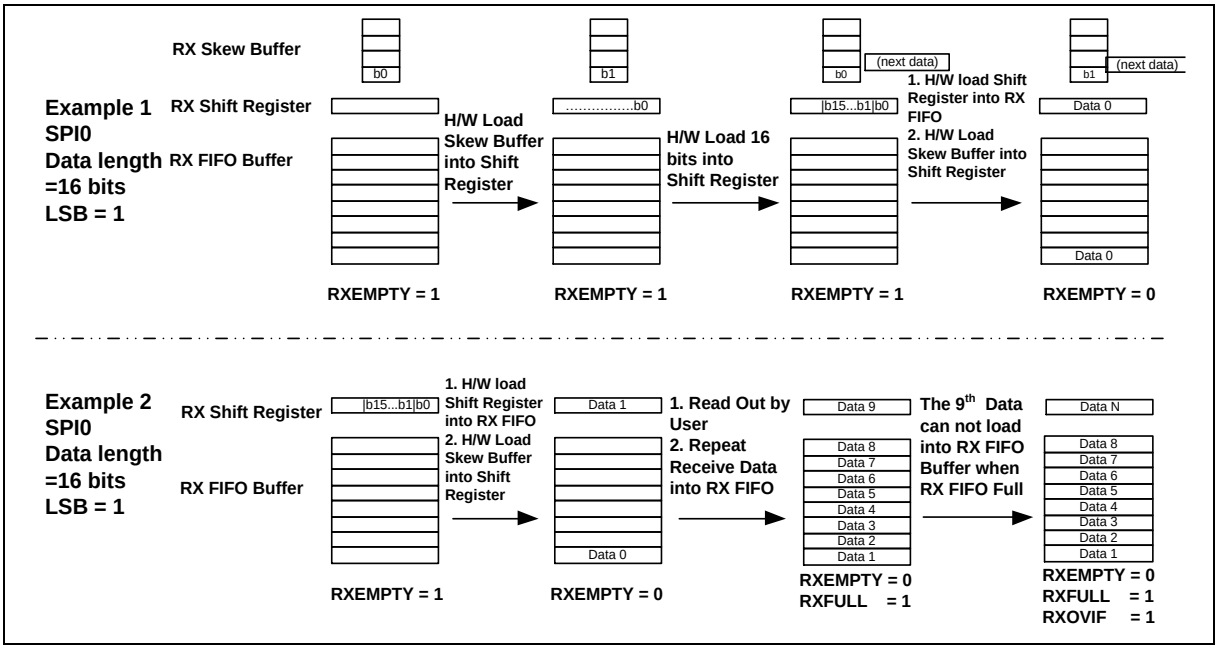


Figure 6.15-15 Receive FIFO Buffer Example for 16 bits of Data Length

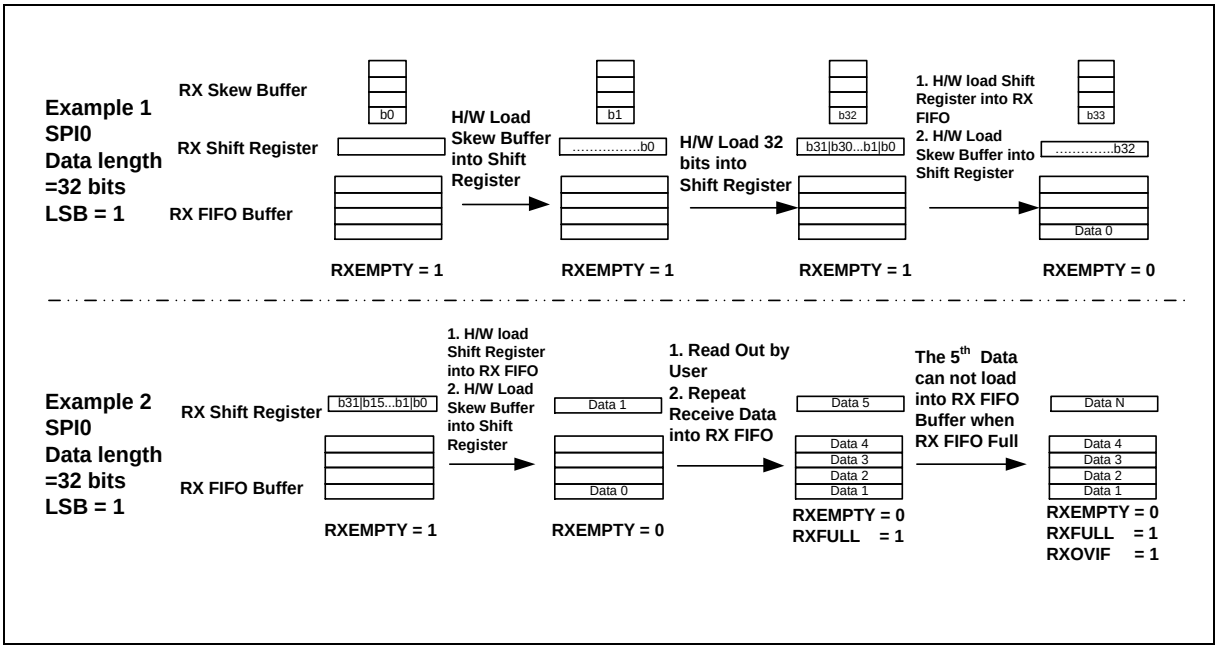


Figure 6.15-16 Receive FIFO Buffer Example for 32 Bits of Data Length

In Slave mode, during transmission operation, when data is written to the SPIx_TX register by software, the data will be loaded into transmit FIFO buffer and the TXEMPTY (SPIx_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 SPIx_TX register as long as the TXFULL (SPIx_STATUS[17]) is 0. After all data have been drawn out by the SPI transmission logic unit and the SPIx_TX register is not updated by software, the TXEMPTY (SPIx_STATUS[16]) will be set to 1.

If there is no data written to the SPIx_TX register, the transmit underflow interrupt flag, TXUFIF (SPIx_STATUS[19]) will be set to 1 when the slave selection signal is active. The output data will be held by TXUFPOL (SPIx_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 (SPIx_STATUS[7]), will be set to 1 as SPIx_SS goes to inactive state.

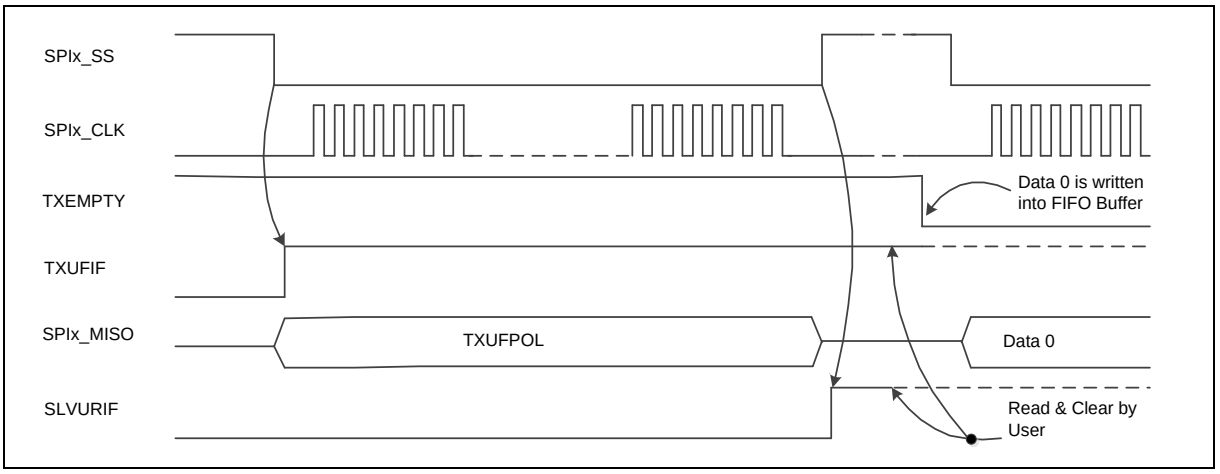


Figure 6.15-17 TX Underflow Event and Slave Under Run Event

In SPI 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 (SPIx_FIFOCTL[6]) because there is TX underflow event. The written data will be transmitted in the next transfer.

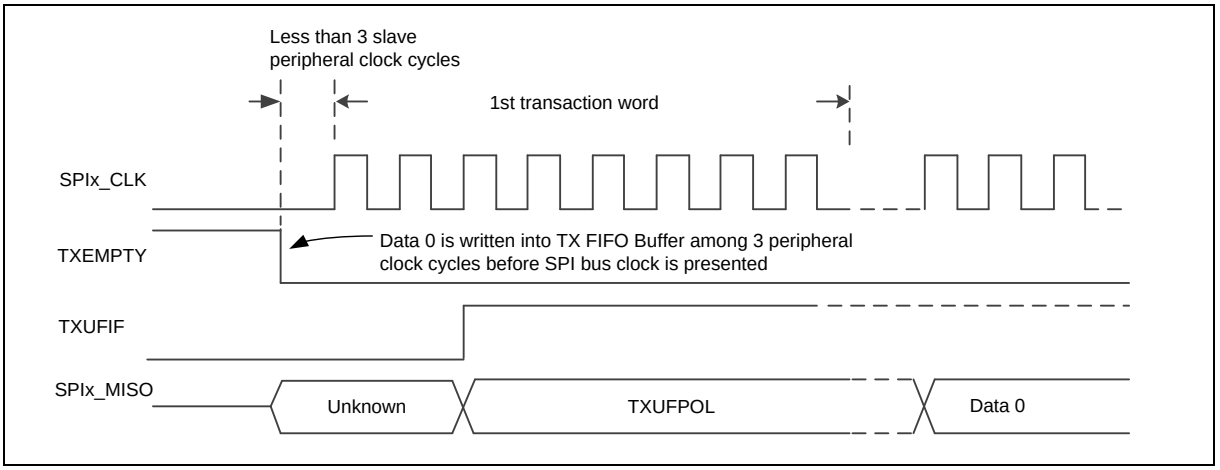


Figure 6.15-18 TX Underflow Event (SPI Slave 3-Wire Mode Enabled)

In Slave mode, during receiving operation, the serial data is received from SPIx_MOSI pin and stored to SPIx_RX register. The reception mechanism is similar to Master mode reception operation. If the receive FIFO buffer contains 4 unread data, the RXFULL (SPIx_STATUS[9]) will be set to 1 and the RXOVIF (SPIx_STATUS[11]) will be set to 1 if there is more serial data received from SPIx_MOSI and follow-up data will be dropped.

If the receive bit count mismatch with the DWIDTH (SPIx_CTL[12:8]) when the slave selection line goes to inactive state, the SLVBEIF (SPIx_STATUS[6]) will be set to 1 in SPI Slave mode. RX data will be written into RX FIFO and SLVBENUM (SPIx_STATUS2[29:24]) will be updated when SLVBERX (SPIx_FIFOCTL[10]) is enabled and SPI slave bit count error event happened. RX data will be dropped and SLVBENUM (SPIx_STATUS2[29:24]) will be fixed to 0x0 when the control register SLVBERX (SPIx_FIFOCTL[10]) is disabled and SPI slave bit count error event happened. The status register SLVBENUM (SPIx_STATUS2[29:24]) indicates that effective bit number of uncompleted RX data when SPI slave bit count error happened. In Figure 6.15-19, the timing diagram of SPI slave bit count error and SLVBENUM is shown, and SLVBENUM will be updated to 0x8 when SPI slave device

receives 8 bits data and DWIDTH (SPIx_CTL[12:8]) sets to 16 bits at SPI bit count error event. The SPI slave bit count error event (SLVBEIF) will generates RX FIFO write operation pulse to write RX data from RX shift register to RX FIFO, and RX FIFO count (RXCNT) will update to 1.

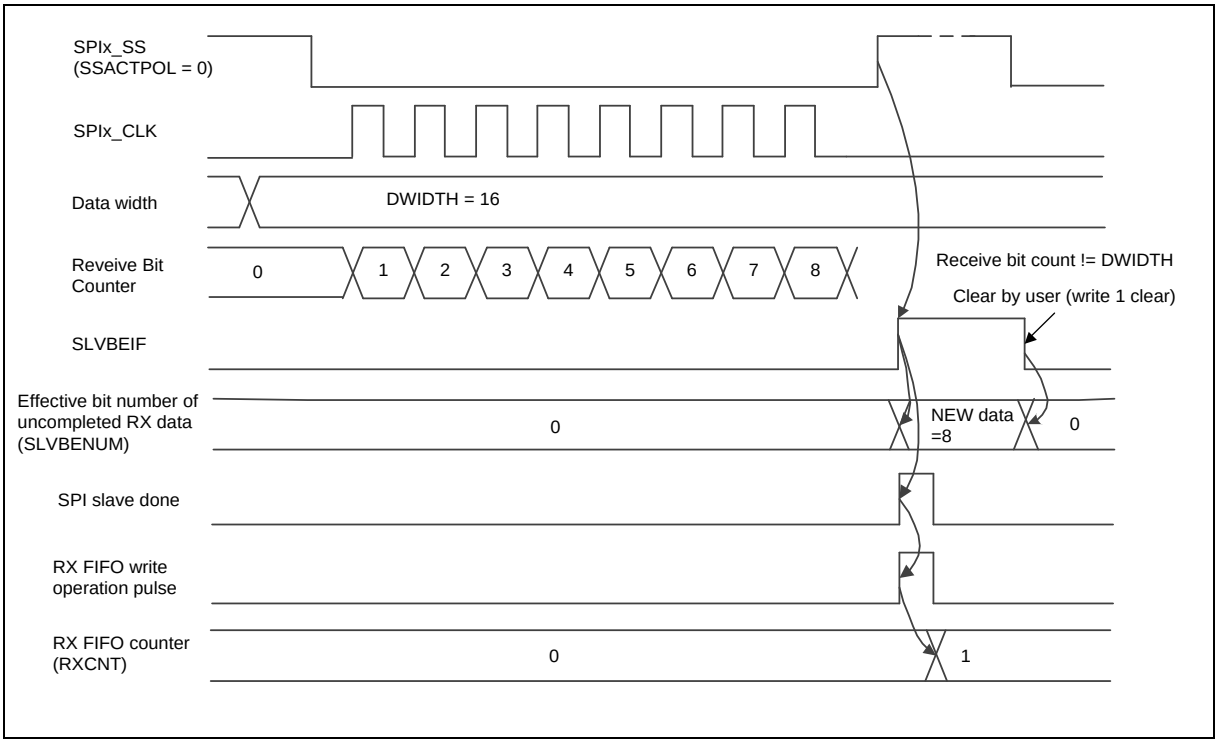


Figure 6.15-19 Slave Mode Bit Count Error and Effective Bit Number of Uncompleted RX Data

A receive time-out function is built 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 (SPIx_STATUS[12]) will be set to 1. When the receive FIFO is read by user, the time-out status will be cleared automatically.

6.15.5.9 Interrupt

- SPI unit transfer interrupt

  As the SPI controller finishes a unit transfer, the unit transfer interrupt flag UNITIF (SPIx_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 (SPIx_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 (SPIx_STATUS[2]) and SSINAIF (SPIx_STATUS[3]), will be set to 1 when the SPIEN (SPIx_CTL[0]) and SLAVE (SPIx_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 (SPIx_SSCTL[13]) or SSACTIEN (SPIx_SSCTL[12]), are set to 1.

- Slave bit count error interrupt

  In Slave mode, if the transmit/receive bit count mismatch with the DWIDTH (SPIx_CTL[12:8]) when the slave selection line goes to inactive state, the SLVBEIF (SPIx_STATUS[6]) will be set to 1. The uncompleted transaction will be dropped from TX shift register. When the control register SLVBERX (SPIx_FIFOCTL[10]) is disabled and SPI slave bit count error event happened, the uncompleted transaction data will be

dropped from RX shift registers. When control register SLVBERX (SPIx_FIFOCTL[10]) is enabled and SPI slave bit count error event happened, the uncompleted transaction data will be written from RX shift registers into RX FIFO. The SPI controller will issue an interrupt if the SLVBEIEN (SPIx_SSCTL[8]) is set to 1.

Note: If the slave selection signal is active but there is no any serial clock input, the SLVBEIF (SPIx_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 SPIx_TX register, the TXUFIF (SPIx_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 (SPIx_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 (SPIx_STATUS[7]) will be set to 1 when SPIx_SS goes to inactive state. The SPI controller will issue a TX under run interrupt if the SLVURIEN (SPIx_SSCTL[9]) is set to 1.

  Note: In Slave 3-Wire mode, the slave selection signal is considered active all the time so that user shall poll the TXUFIF (SPIx_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 unread data for 17~32-bits of data length, the RXFULL (SPIx_STATUS[9]) will be set to 1 and the RXOVIF (SPIx_STATUS[11]) will be set to 1 if there is more serial data received from SPI bus and follow-up data will be dropped. The SPI controller will issue an interrupt if the RXOVIEN (SPIx_FIFOCTL[5]) is set to 1.

  If the receive FIFO buffer contains 8 unread data for 4~16-bits of data length, the RXFULL (SPIx_STATUS[9]) will be set to 1 and the RXOVIF (SPIx_STATUS[11]) will be set to 1 if there is more serial data received from SPI bus and follow-up data will be dropped.The SPI controller will issue an interrupt if the RXOVIEN (SPIx_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 (SPIx_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 (SPIx_FIFOCTL[30:28]), the transmit FIFO interrupt flag TXTHIF (SPIx_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 (SPIx_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 (SPIx_FIFOCTL[26:24]), the receive FIFO interrupt flag RXTHIF (SPIx_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 (SPIx_FIFOCTL[2]), is set to 1.

6.15.5.10 I²S Mode

The SPI0~SPI1 controllers support I²S mode with PCM mode A, PCM mode B, MSB justified and I²S data format. The bit count of an audio channel is determined by WDWIDTH (SPIx_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 I²S 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.

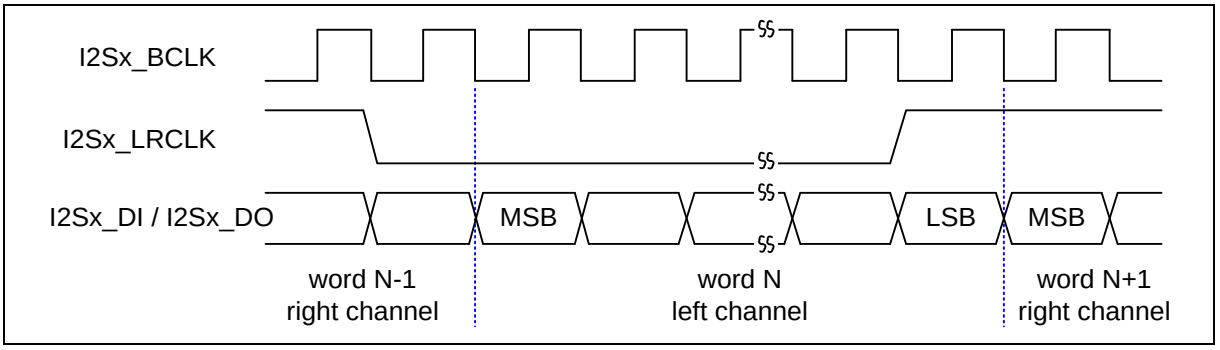


Figure 6.15-20 I²S Data Format Timing Diagram

In MSB justified data format, the MSB is sent and latched on the first clock of an audio channel.

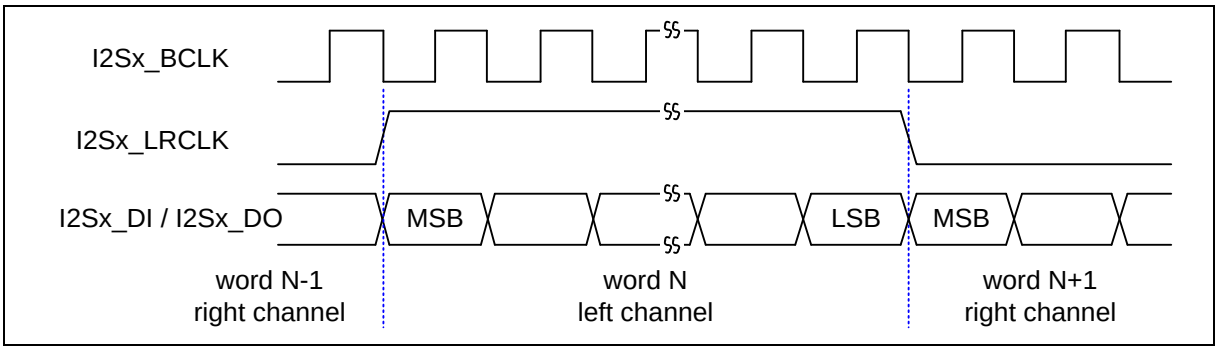


Figure 6.15-21 MSB Justified Data Format Timing Diagram

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.

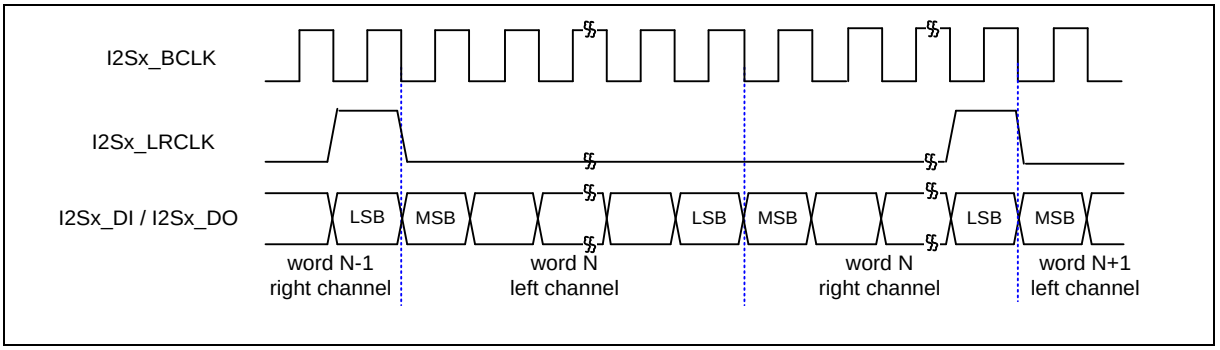


Figure 6.15-22 PCM Mode A Timing Diagram

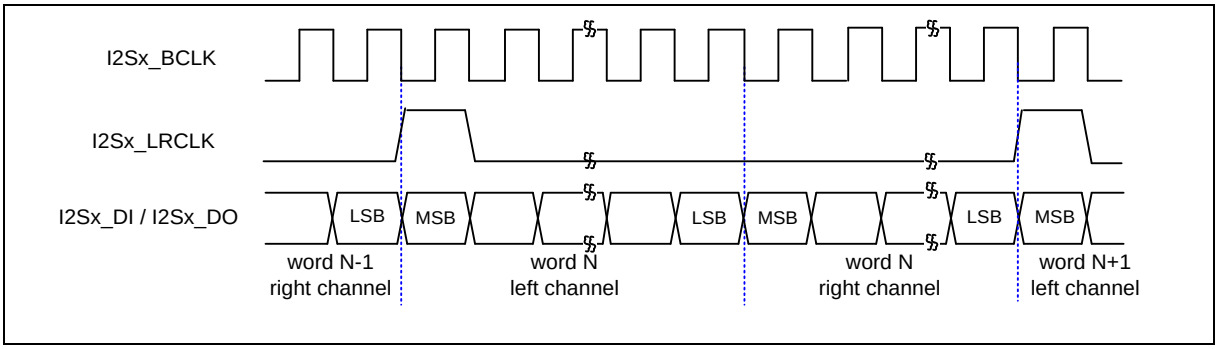


Figure 6.15-23 PCM Mode B Timing Diagram

6.15.5.11 I²S Mode FIFO Operation

Sound transmission includes stereo and mono modes. The number of data bits in each mode can also be set and adjusted as 8-bits, 16-bits, 24-bits or 32-bits. Figure 6.15-24 shows the order in which the data content of various sound modes are placed in the FIFO buffer.

In monaural mode, both left and right channels transmit the same data. During reception, the RXLCH (SPIx_I2SCTL[23]) bit determines whether the content to be received comes from left or right channel data.

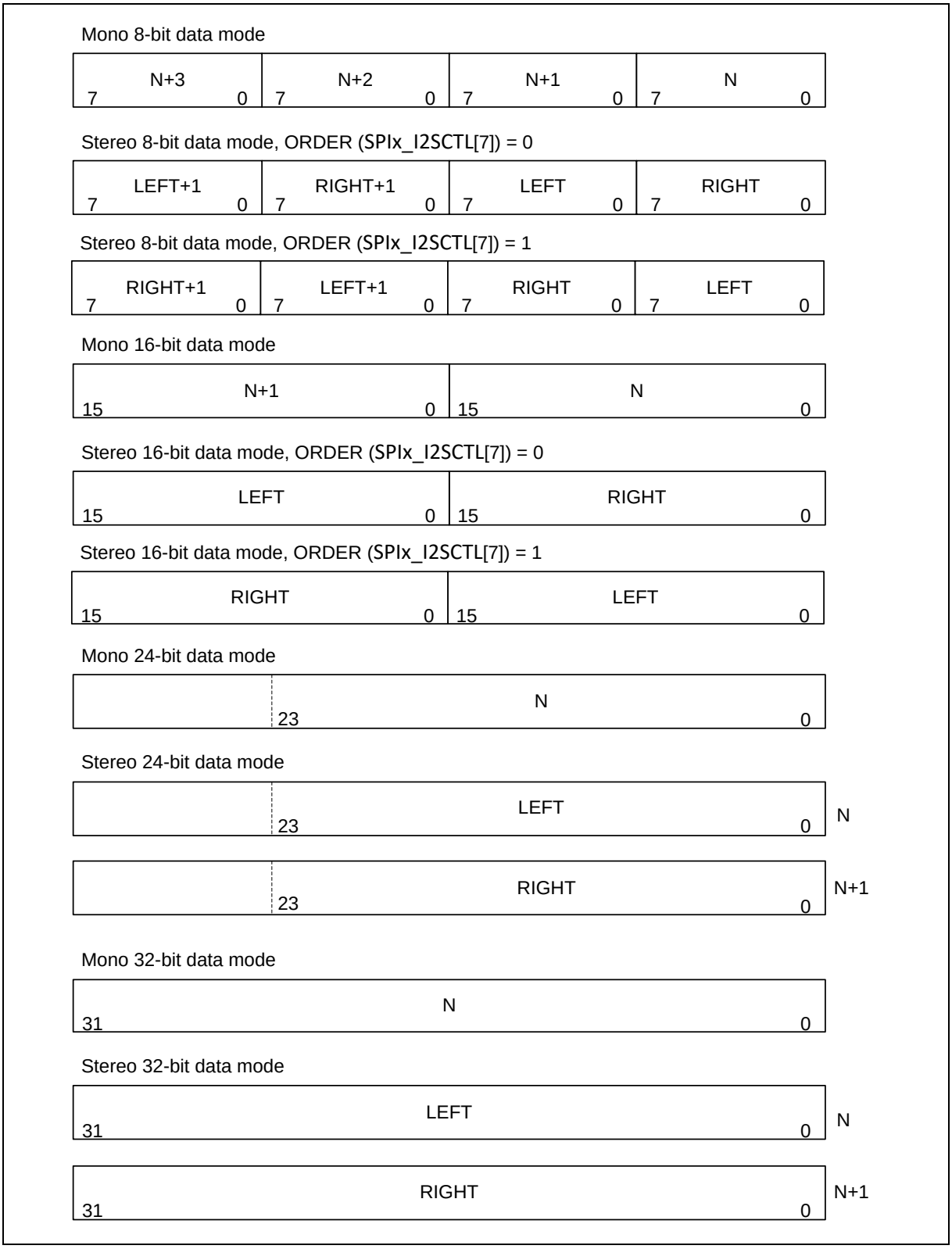


Figure 6.15-24 FIFO Contents for Various I²S Modes

6.15.5.12 Dummy Data Number for I²S / PCM Master mode and Monaural Mode

Before I²S / PCM master starts to send TX data to an external slave device, user sets control registers

I2SEN (SPIx_I2SCTL[0]) enable, TXEN (SPIx_I2SCTL[1]) enable, and write TX data to TX FIFO. After master sends dummy data (data with zero value) to an external slave device, master will send TX FIFO data to an external slave device. Table 6.15-2 shows the number of dummy data for monaural mode, and the unit of dummy data number is L channel + R channel.

Data Width (SPIx_I2SCTL[5:4])	Pre-transmit Dummy Data (Unit = L Channel + R Channel)
8 bits	0
16 bits	0
24 bits	1
32 bits	1

Table 6.15-2 Dummy Data Number for I²S / PCM Master Mode and Monaural Mode

6.15.5.13 Dummy Data Number for I²S / PCM Master mode and Stereo Mode

Before I²S / PCM master starts to send TX data to an external slave device, user sets control registers I2SEN (SPIx_I2SCTL[0]) enabled, TXEN (SPIx_I2SCTL[1]) enabled, and writes TX data to TX FIFO. After master sends dummy data (data with zero value) to an external slave device, master will send TX FIFO data to an external slave device. Table 6.15-3 shows the number of dummy data for stereo mode, and the unit of dummy data number is L channel + R channel.

Data Width (SPIx_I2SCTL[5:4])	Pre-transmit Dummy Data (Unit = L Channel + R Channel)
8 bits	0
16 bits	0
24 bits	1
32 bits	1

Table 6.15-3 Dummy Data Number for I²S / PCM Master Mode and Stereo Mode

6.15.5.14 Dummy Data Number for I²S Slave mode and Monaural Mode

Before I²S / PCM slave starts to send TX data to an external master device, user sets control registers I2SEN (SPIx_I2SCTL[0]) enable, TXEN (SPIx_I2SCTL[1]) enable, SLAVE mode (SPIx_I2SCTL[8]), and write TX data to TX FIFO. After slave sends dummy data (data with zero value) to an external master device, slave will send TX FIFO data to an external master device. Table 6.15-4 shows the number of dummy data for I²S slave monaural mode, and the unit of dummy data number is L channel + R channel.

Data Width (SPIx_I2SCTL[5:4])	Pre-transmit Dummy Data (Unit = L Channel + R Channel)
8 bits	3
16 bits	2
24 bits	2
32 bits	2

Table 6.15-4 Dummy Data Number for I²S Slave Mode and Monaural Mode

6.15.5.15 Dummy Data Number for PCM Slave Mode and Monaural Mode

Before I²S / PCM slave starts to send TX data to an external master device, user sets control registers I2SEN (SPIx_I2SCTL[0]) enable, TXEN (SPIx_I2SCTL[1]) enable, SLAVE mode (SPIx_I2SCTL[8]), and write TX data to TX FIFO. After slave sends dummy data (data with zero value) to an external

master device, slave will send TX FIFO data to an external master device. Table 6.15-5 shows the number of dummy data for PCM slave monaural mode, and the unit of dummy data number is L channel + R channel.

Data Width (SPIx_I2SCTL[5:4])	Pre-transmit Dummy Data (Unit = L Channel + R Channel)
8 bits	2
16 bits	1
24 bits	1
32 bits	1

Table 6.15-5 Dummy Data Number for PCM Slave Mode and Monaural Mode

6.15.5.16 Dummy Data Number for I²S Slave mode and Stereo Mode

Before I²S / PCM slave starts to send TX data to an external master device, user sets control registers I2SEN (SPIx_I2SCTL[0]) enable, TXEN (SPIx_I2SCTL[1]) enable, SLAVE mode (SPIx_I2SCTL[8]), and write TX data to TX FIFO. After slave sends dummy data (data with zero value) to an external master device, slave will send TX FIFO data to an external master device. Table 6.15-6 shows number of dummy data for I²S slave stereo mode, and the unit of dummy data number is L channel + R channel.

Data Width (SPIx_I2SCTL[5:4])	Pre-transmit Dummy Data (Unit = L Channel + R Channel)
8 bits	3
16 bits	2
24 bits	2
32 bits	2

Table 6.15-6 Dummy Data Number for I²S Slave Mode and Stereo Mode

6.15.5.17 Dummy Data Number for PCM Slave mode and Stereo Mode

Before I²S / PCM slave starts to send TX data to an external master device, user sets control registers I2SEN (SPIx_I2SCTL[0]) enable, TXEN (SPIx_I2SCTL[1]) enable, SLAVE mode (SPIx_I2SCTL[8]), and write TX data to TX FIFO. After slave sends dummy data (data with zero value) to an external master device, slave will send TX FIFO data to an external master device. Table 6.15-7 shows number of dummy data for PCM slave stereo mode, and the unit of dummy data number is L channel + R channel.

Data Width (SPIx_I2SCTL[5:4])	Pre-transmit Dummy Data (Unit = L Channel + R Channel)
8 bits	2
16 bits	1
24 bits	1
32 bits	1

Table 6.15-7 Dummy Data Number for PCM Slave Mode and Stereo Mode

6.15.6 Timing Diagram

The active state of slave selection signal can be defined by setting the SSACTPOL (SPIx_SSCTL[2]). The SPI clock which is in idle state can be configured as high or low state by setting the CLKPOL (SPIx_CTL[3]). It also provides the bit length of a transaction word in DWIDTH (SPIx_CTL[12:8]), and

transmitting/receiving data from MSB or LSB first in LSB (SPIx_CTL[13]). User can also select which edge of SPI clock to transmit/receive data in TXNEG/RXNEG (SPIx_CTL[2:1]). Four SPI timing diagrams for master/slave operations and the related settings are shown below.

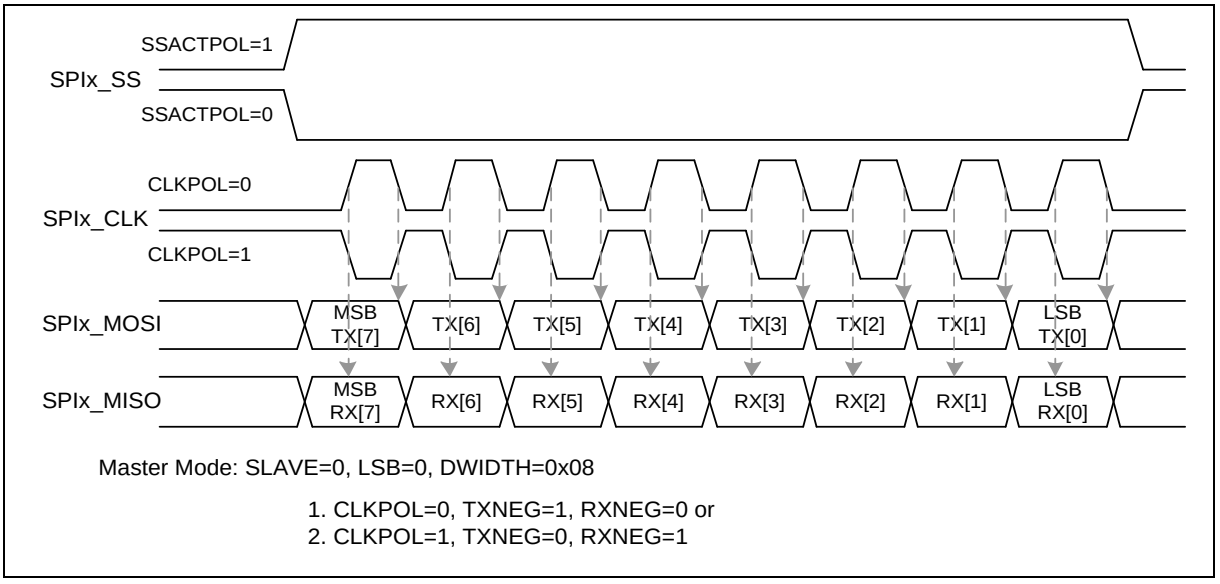


Figure 6.15-25 SPI Timing in Master Mode

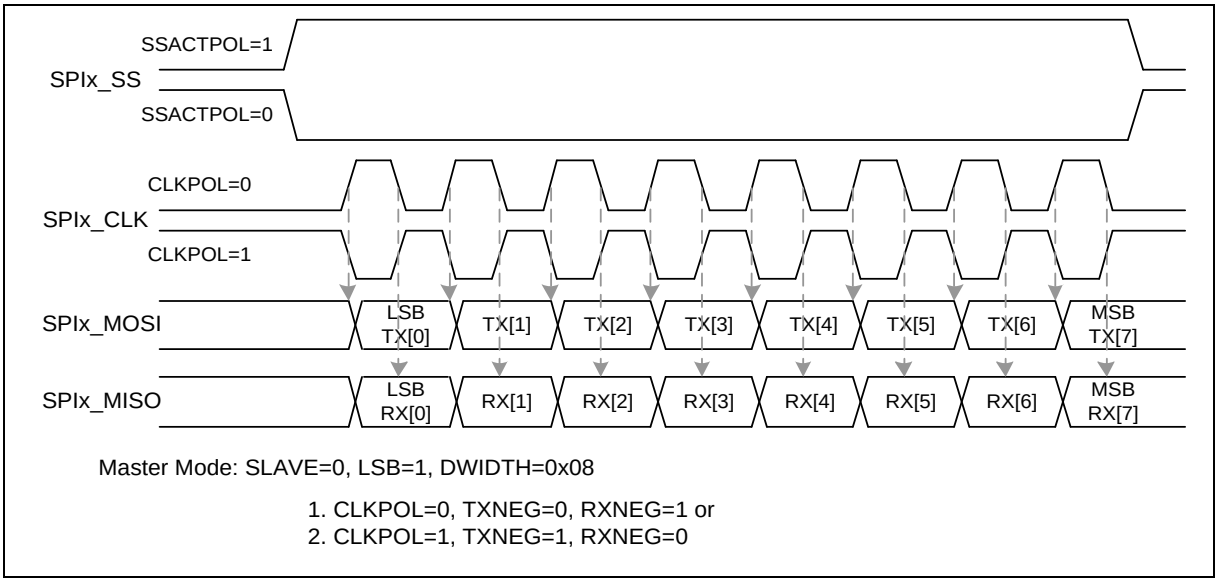


Figure 6.15-26 SPI Timing in Master Mode (Alternate Phase of SPIx_CLK)

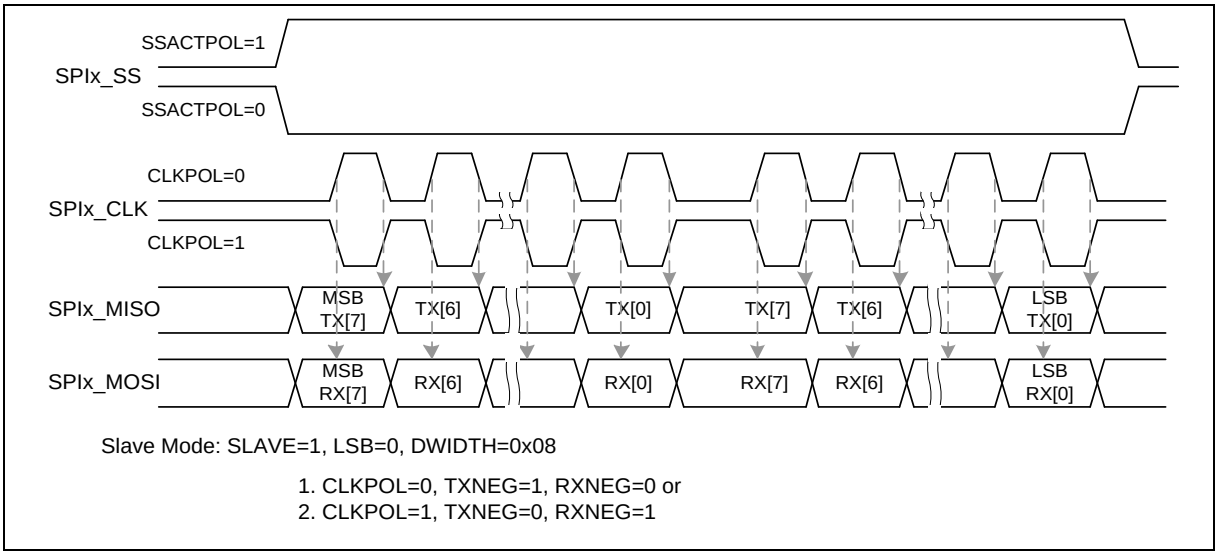


Figure 6.15-27 SPI Timing in Slave Mode

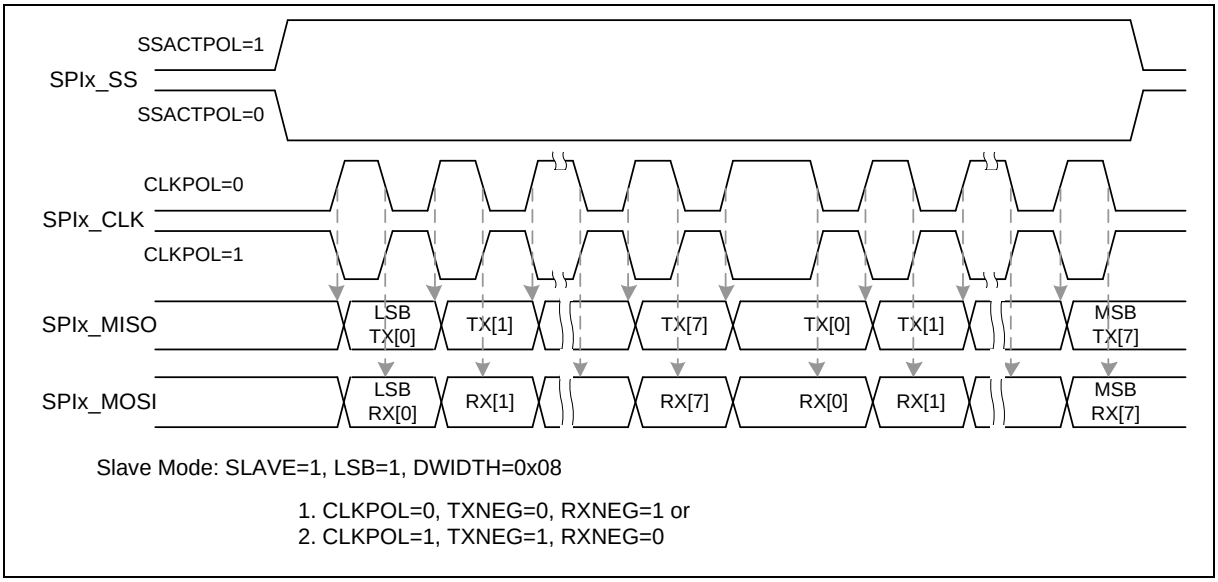


Figure 6.15-28 SPI Timing in Slave Mode (Alternate Phase of SPIx_CLK)

6.15.7 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 (SPIx_CLKDIV [8:0]) to determine the output frequency of SPI clock.
2. Write the SPIx_SSCTL register a proper value for the related settings of Master mode:
   1) Clear AUTOSS (SPIx_SSCTL[3]) to 0 to disable the Automatic Slave Selection function.
   2) Configure slave selection signal as active low by clearing SSACTPOL (SPIx_SSCTL[2]) to 0.
   3) Enable slave selection signal by setting SS (SPIx_SSCTL[0]) to 1 to activate the off-chip slave device.
3. Write the related settings into the SPIx_CTL register to control the SPI master actions.
   1) Configure this SPI controller as master device by setting SLAVE (SPIx_CTL[18]) to 0.
   2) Force the SPI clock idle state at low by clearing CLKPOL (SPIx_CTL[3]) to 0.
   3) Select data transmitted on negative edge of SPI bus clock by setting TXNEG (SPIx_CTL[2]) to 1.
   4) Select data latched on positive edge of SPI bus clock by clearing RXNEG (SPIx_CTL[1]) to 0.
   5) Set the bit length of a transaction as 8-bit in DWIDTH bit field (SPIx_CTL[12:8] = 0x08).
   6) Set MSB transfer first by clearing LSB (SPIx_CTL[13]) to 0.
4. Set SPIEN (SPIx_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 SPIx_TX register.
6. Waiting for SPI interrupt if the UNITIEN (SPIx_CTL[17]) is set to 1, or just polling the unit transfer interrupt flag UNITIF (SPIx_STATUS[1]).
7. Read out the received one byte data from SPIx_RX register.
8. Go to 5) to continue another data transfer or set SS (SPIx_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 SPIx_SSCTL register a proper value for the related settings of Slave mode.
2. Select high level for the input of slave selection signal by setting SSACTPOL (SPIx_SSCTL[2]) to 1.
3. Write the related settings into the SPIx_CTL register to control this SPI slave actions

1) Set the SPI controller as slave device by setting SLAVE (SPIx_CTL[18]) to 1.
2) Select the SPI clock idle state at high by setting CLKPOL (SPIx_CTL[3]) to 1.
3) Select data transmitted on negative edge of SPI bus clock by setting TXNEG (SPIx_CTL[2]) to 1.
4) Select data latched on positive edge of SPI bus clock by clearing RXNEG (SPIx_CTL[1]) to 0.
5) Set the bit length of a transaction as 8-bit in DWIDTH bit field (SPIx_CTL[12:8] = 0x08).

4. Set LSB transfer first by setting LSB (SPIx_CTL[13]) to 1.
5. Set the SPIEN (SPIx_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.
6. 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 SPIx_TX register.
7. 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 SPIx_TX register does not need to be updated by software.
8. Waiting for SPI interrupt if the UNITIEN (SPIx_CTL[17]) is set to 1, or just polling the unit transfer interrupt flag UNITIF (SPIx_STATUS[1]).
9. Read out the received one byte data from SPIx_RX register.
10. Go to 7 to continue another data transfer or stop data transfer.

6.15.8 Register Map

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

Register	Offset	R/W	Description	Reset Value
SPI Base Address: **SPIx_BA = 0x4006_1000 + (0x0000_1000 * x)** **x=0, 1**				
SPIx_CTL	SPIx_BA+0x00	R/W	SPI Control Register	0x0000_0034
SPIx_CLKDIV	SPIx_BA+0x04	R/W	SPI Clock Divider Register	0x0000_0000
SPIx_SSCTL	SPIx_BA+0x08	R/W	SPI Slave Select Control Register	0x0000_0000
SPIx_PDMACTL	SPIx_BA+0x0C	R/W	SPI PDMA Control Register	0x0000_0000
SPIx_FIFOCTL	SPIx_BA+0x10	R/W	SPI FIFO Control Register	0x2200_0000
SPIx_STATUS	SPIx_BA+0x14	R/W	SPI Status Register	0x0005_0110
SPIx_STATUS2	SPIx_BA+0x18	R	SPI Status2 Register	0x0000_0000
SPIx_TX	SPIx_BA+0x20	W	SPI Data Transmit Register	0x0000_0000
SPIx_RX	SPIx_BA+0x30	R	SPI Data Receive Register	0x0000_0000
SPIx_I2SCTL	SPIx_BA+0x60	R/W	I^2S Control Register	0x0000_0000
SPIx_I2SCLK	SPIx_BA+0x64	R/W	I^2S Clock Divider Control Register	0x0000_0000
SPIx_I2SSTS	SPIx_BA+0x68	R/W	I^2S Status Register	0x0005_0100

6.15.9 Register Description

SPI Control Register (SPIx_CTL)

Register	Offset	R/W	Description	Reset Value
SPIx_CTL	SPIx_BA+0x00	R/W	SPI Control Register	0x0000_0034

Note: Not supported in I²S mode.

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			DATDIR	REORDER	SLAVE	UNITIEN	Reserved
15	14	13	12	11	10	9	8
RXONLY	HALFDPX	LSB	DWIDTH				
7	6	5	4	3	2	1	0
SUSPITV				CLKPOL	TXNEG	RXNEG	SPIEN

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20]	**DATDIR**	**Data Port Direction Control** This bit is used to select the data input/output direction in half-duplex transfer 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.
[15]	**RXONLY**	**Receive-only Mode Enable Bit** 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 (SPIx_CTL[20]) can be used to set the data direction in half-duplex transfer. 0 = SPI operates in full-duplex transfer.

		1 = SPI operates in half-duplex transfer.
[13]	**LSB**	**Send LSB First** 0 = The MSB, which bit of transmit/receive register depends on the setting of DWIDTH, is transmitted/received first. 1 = The LSB, bit 0 of the SPI TX register, is sent first to the SPI data output pin, and the first bit received from the SPI data input pin will be put in the LSB position of the RX register (bit 0 of SPI_RX).
[12:8]	**DWIDTH**	**Data Width** This field specifies how many bits can be transmitted / received in one transaction. The minimum bit length is 4 bits and can up to 32 bits. DWIDTH = 0x04 …. 4 bits. DWIDTH = 0x05 …. 5 bits. …… DWIDTH = 0x1F …. 31 bits. DWIDTH = 0x00 …. 32 bits. **Note:** 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.
[7:4]	**SUSPITV**	**Suspend Interval** 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 + 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. **Note:** Master Mode only.
[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 SPIx_CTL, SPIx_CLKDIV, SPIx_SSCTL and SPIx_FIFOCTL registers, user shall clear the SPIEN (SPIx_CTL[0]) and confirm the SPIENSTS (SPIx_STATUS[15]) is 0.

SPI Clock Divider Register (SPIx_CLKDIV)

Register	Offset	R/W	Description	Reset Value
SPIx_CLKDIV	SPIx_BA+0x04	R/W	SPI 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.
[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 1:** Not supported in I^2S mode. **Note 2:** The time interval must be larger than or equal to 8 peripheral clock cycles between releasing SPI IP software reset and setting this clock divider register.

Note: DIVIDER should be set carefully because the peripheral clock frequency must be slower than or equal to system frequency.

SPI Slave Select Control Register (SPIx_SSCTL)

Register	Offset	R/W	Description	Reset Value
SPIx_SSCTL	SPIx_BA+0x08	R/W	SPI Slave Select Control Register	0x0000_0000

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							
15	14	13	12	11	10	9	8
Reserved		SSINAIEN	SSACTIEN	Reserved		SLVURIEN	SLVBEIEN
7	6	5	4	3	2	1	0
Reserved			SLV3WIRE	AUTOSS	SSACTPOL	Reserved	SS

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13]	**SSINAIEN**	**Slave Select Inactive Interrupt Enable Bit** 0 = Slave select inactive interrupt Disabled. 1 = Slave select inactive interrupt Enabled.
[12]	**SSACTIEN**	**Slave Select Active Interrupt Enable Bit** 0 = Slave select active interrupt Disabled. 1 = Slave select active interrupt Enabled.
[11:10]	**Reserved**	Reserved.
[9]	**SLVURIEN**	**Slave Mode TX Under Run Interrupt Enable Bit** 0 = Slave mode TX under run interrupt Disabled. 1 = Slave mode TX under run interrupt Enabled.
[8]	**SLVBEIEN**	**Slave Mode Bit Count Error Interrupt Enable Bit** 0 = Slave mode bit count error interrupt Disabled. 1 = Slave mode bit count error interrupt Enabled.
[7:5]	**Reserved**	Reserved.
[4]	**SLV3WIRE**	**Slave 3-wire Mode Enable Bit** In Slave 3-wire mode, the SPI controller can work with 3-wire interface including SPIx_CLK, SPIx_MISO and SPIx_MOSI pins. 0 = 4-wire bi-direction interface. 1 = 3-wire bi-direction interface. **Note 1:** The value of this register equals to control register SLAVE (SPIx_I2SCTL[8]) when I^2S mode is enabled. **Note 2:** SPI Slave Mode only.
[3]	**AUTOSS**	**Automatic Slave Selection Function Enable Bit** 0 = Automatic slave selection function Disabled. Slave selection signal will be asserted/de-asserted according to SS (SPIx_SSCTL[0]). 1 = Automatic slave selection function Enabled.

		Note: Master Mode only.
[2]	**SSACTPOL**	**Slave Selection Active Polarity** This bit defines the active polarity of slave selection signal (SPIx_SS). 0 = The slave selection signal SPIx_SS is active low. 1 = The slave selection signal SPIx_SS is active high.
[1]	**Reserved**	Reserved.
[0]	**SS**	**Slave Selection Control** If AUTOSS bit is cleared to 0, 0 = set the SPIx_SS line to inactive state. 1 = set the SPIx_SS line to active state. If the AUTOSS bit is set to 1, 0 = Keep the SPIx_SS line at inactive state. 1 = SPIx_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 SPIx_SS is specified in SSACTPOL (SPIx_SSCTL[2]). **Note:** Master Mode only.

SPI PDMA Control Register (SPIx_PDMACTL)

Register	Offset	R/W	Description	Reset Value
SPIx_PDMACTL	SPIx_BA+0x0C	R/W	SPI PDMA Control Register	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**PDMARST**	**PDMA Reset** 0 = No effect. 1 = Reset the PDMA control logic of the SPI controller. This bit will be automatically cleared to 0.
[1]	**RXPDMAEN**	**Receive PDMA Enable Bit** 0 = 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. **Note1:** 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. **Note2:** In SPI Master mode with full duplex transfer, if both TX and RX PDMA functions are enabled, TX PDMA function cannot be disabled prior to RX PDMA function. User can disable RX PDMA function firstly or disable both functions simultaneously.

SPI FIFO Control Register (SPIx_FIFOCTL)

Register	Offset	R/W	Description	Reset Value
SPIx_FIFOCTL	SPIx_BA+0x10	R/W	SPI FIFO Control Register	0x2200_0000

31	30	29	28	27	26	25	24
Reserved	TXTH			Reserved	RXTH		
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					SLVBERX	TXFBCLR	RXFBCLR
7	6	5	4	3	2	1	0
TXUFIEN	TXUFPOL	RXOVIEN	RXTOIEN	TXTHIEN	RXTHIEN	TXRST	RXRST

Bits	Description	
[31]	**Reserved**	Reserved.
[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. The MSB of this bit field is only meaningful while SPI mode 4~16 bits of data length.
[27]	**Reserved**	Reserved.
[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. The MSB of this bit field is only meaningful while SPI mode 4~16 bits of data length.
[23:11]	**Reserved**	Reserved.
[10]	**SLVBERX**	**RX FIFO Write Data Enable Bit When Slave Mode Bit Count Error** 0 = Uncompleted RX data will be dropped from RX FIFO when bit count error event happen in SPI slave mode. 1 = Uncompleted RX data will be written into RX FIFO when bit count error event happen in SPI slave mode. User can read SLVBENUM (SPIx_STATUS2[29:24]) to know that the effective bit number of uncompleted RX data when SPI slave bit count error happened. **Note:** SPI Slave Mode only.
[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 (SPIx_STATUS[19]) will be set to 1. This bit is used to enable the TX underflow interrupt. 0 = Slave TX underflow interrupt Disabled. 1 = Slave TX underflow interrupt Enabled.
[6]	**TXUFPOL**	**TX Underflow Data Polarity** 0 = The SPI data out is keep 0 if there is TX underflow event in Slave mode. 1 = The SPI data out is keep 1 if there is TX underflow event in Slave mode. **Note 1:** The TX underflow event occurs if there is no any data in TX FIFO when the slave selection signal is active. **Note 2:** This bit should be set as 0 in I^2S mode. **Note 3:** When TX underflow event occurs, SPIx_MISO pin state will be determined by this setting even though TX FIFO is not empty afterward. Data stored in TX FIFO will be sent through SPIx_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 (SPIx_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 (SPIx_STATUS[23]) to check if reset is accomplished or not.

SPI Status Register (SPIx_STATUS)

Register	Offset	R/W	Description	Reset Value
SPIx_STATUS	SPIx_BA+0x14	R/W	SPI Status Register	0x0005_0110

Note: Not supported in I²S mode.

31	30	29	28	27	26	25	24
TXCNT				RXCNT			
23	22	21	20	19	18	17	16
TXRXRST	Reserved			TXUFIF	TXTHIF	TXFULL	TXEMPTY
15	14	13	12	11	10	9	8
SPIENSTS	Reserved		RXTOIF	RXOVIF	RXTHIF	RXFULL	RXEMPTY
7	6	5	4	3	2	1	0
SLVURIF	SLVBEIF	Reserved	SSLINE	SSINAIF	SSACTIF	UNITIF	BUSY

Bits	Description	
[31:28]	**TXCNT**	**Transmit FIFO Data Count (Read Only)** This bit field indicates the valid data count of transmit FIFO buffer.
[27:24]	**RXCNT**	**Receive FIFO Data Count (Read Only)** This bit field indicates the valid data count of receive FIFO buffer.
[23]	**TXRXRST**	**TX or RX Reset Status (Read Only)** 0 = The reset function of TXRST or RXRST is done. 1 = Doing the reset function of TXRST or RXRST. **Note:** Both the reset operations of TXRST and RXRST need 3 system clock cycles + 2 peripheral clock cycles. User can check the status of this bit to monitor the reset function is doing or done.
[22:20]	**Reserved**	Reserved.
[19]	**TXUFIF**	**TX Underflow Interrupt Flag** When the TX underflow event occurs, this bit will be set to 1, the state of data output pin depends on the setting of TXUFPOL. 0 = No effect. 1 = No data in Transmit FIFO and TX shift register when the slave selection signal is active. **Note 1:** This bit will be cleared by writing 1 to it. **Note 2:** If reset slave's transmission circuit when slave selection signal is active, this flag will be set to 1 after 2 peripheral clock cycles + 3 system clock cycles since the reset operation is done.
[18]	**TXTHIF**	**Transmit FIFO Threshold Interrupt Flag (Read Only)** 0 = The valid data count within the transmit FIFO buffer is larger than the setting value of TXTH. 1 = The valid data count within the transmit FIFO buffer is less than or equal to the setting value of TXTH.
[17]	**TXFULL**	**Transmit FIFO Buffer Full Indicator (Read Only)** 0 = Transmit FIFO buffer is not full. 1 = Transmit FIFO buffer is full.
[16]	**TXEMPTY**	**Transmit FIFO Buffer Empty Indicator (Read Only)** 0 = Transmit FIFO buffer is not empty.

		1 = Transmit FIFO buffer is empty.
[15]	**SPIENSTS**	**SPI Enable Status (Read Only)** 0 = SPI controller Disabled. 1 = SPI controller Enabled. **Note:** The SPI peripheral clock is asynchronous with the system clock. In order to make sure the SPI control logic is disabled, this bit indicates the real status of SPI controller.
[14:13]	**Reserved**	Reserved.
[12]	**RXTOIF**	**Receive Time-out Interrupt Flag** 0 = No receive FIFO time-out event. 1 = Receive FIFO buffer is not empty and no read operation on receive FIFO buffer over 64 SPI 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 occurred. **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 occurred. **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.
[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 (SPIx_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 lists the bus busy conditions: a. SPIEN (SPIx_CTL[0]) = 1 and TXEMPTY = 0. b. For SPI Master mode, SPIEN (SPIx_CTL[0]) = 1 and TXEMPTY = 1 but the current transaction is not finished yet. c. For SPI Master mode, SPIEN (SPIx_CTL[0]) = 1 and RXONLY = 1. d. For SPI Slave mode, SPIEN (SPIx_CTL[0]) = 1 and there is serial clock input into the SPI core logic when slave select is active. **e.** For SPI Slave mode, SPIEN (SPIx_CTL[0]) = 1 and the transmit buffer or transmit shift register is not empty even if the slave select is inactive. **Note:** By applications, this SPI bus flag should be used with other status registers in SPIx_STATUS such as TXCNT, RXCNT, TXTHIF, TXFULL, TXEMPTY, RXTHIF, RXFULL, RXEMPTY, and UNITIF. Therefore the SPI transfer done events of TX/RX operations can be obtained at correct timing point.

SPI Status2 Register (SPIx_STATUS2)

Register	Offset	R/W	Description	Reset Value
SPIx_STATUS2	SPIx_BA+0x18	R	SPI Status2 Register	0x0000_0000

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

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29:24]	**SLVBENUM**	**Effective Bit Number of Uncompleted RX Data** This status register indicates that effective bit number of uncompleted RX data when SLVBERX (SPIx_FIFOCTL[10]) is enabled and RX bit count error event happen in SPI slave mode. This status register will be fixed to 0x0 when SLVBERX (SPIx_FIFOCTL[10]) is disabled. **Note 1:** This register will be cleared to 0x0 when user write 0x1 to SLVBEIF (SPIx_STATUS[6]). **Note 2:** SPI Slave Mode only.
[23:0]	**Reserved**	Reserved.

SPI Data Transmit Register (SPIx_TX)

Register	Offset	R/W	Description	Reset Value
SPIx_TX	SPIx_BA+0x20	W	SPI Data Transmit Register	0x0000_0000

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

Bits		Description
[31:0]	**TX**	**Data Transmit Register** The data transmit registers pass through the transmitted data into the 4-level transmit FIFO buffers. The number of valid bits depends on the setting of DWIDTH (SPIx_CTL[12:8]) in SPI mode or WDWIDTH (SPIx_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 (SPIx_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 (SPIx_RX)

Register	Offset	R/W	Description	Reset Value
SPIx_RX	SPIx_BA+0x30	R	SPI Data Receive Register	0x0000_0000

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

Bits	Description	
[31:0]	RX	**Data Receive Register (Read Only)** 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 (SPIx_STATUS[8] or SPIx_I2SSTS[8]) is not set to 1, the receive FIFO buffers can be accessed through software by reading this register.

I²S Control Register (SPIx_I2SCTL)

Register	Offset	R/W	Description	Reset Value
SPIx_I2SCTL	SPIx_BA+0x60	R/W	I²S Control Register	0x0000_0000

Note: Not supported in SPI mode.

31	30	29	28	27	26	25	24
SLVERRIEN	Reserved	FORMAT		Reserved		LZCIEN	RZCIEN
23	22	21	20	19	18	17	16
RXLCH	Reserved					LZCEN	RZCEN
15	14	13	12	11	10	9	8
MCLKEN	Reserved						SLAVE
7	6	5	4	3	2	1	0
ORDER	MONO	WDWIDTH		MUTE	RXEN	TXEN	I2SEN

Bits	Description	
[31]	**SLVERRIEN**	**Bit Number Error Interrupt Enable Bit for Slave Mode** Interrupt occurs if this bit is set to 1 and bit number error event occurs. 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[30]	**Reserved**	Reserved.
[29:28]	**FORMAT**	**Data Format Selection** 00 = I²S data format. 01 = MSB justified data format. 10 = PCM mode A. 11 = PCM mode B.
[27:26]	**Reserved**	Reserved.
[25]	**LZCIEN**	**Left Channel Zero Cross Interrupt Enable Bit** Interrupt occurs if this bit is set to 1 and left channel zero cross event occurs. 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[24]	**RZCIEN**	**Right Channel Zero Cross Interrupt Enable Bit** Interrupt occurs if this bit is set to 1 and right channel zero cross event occurs. 0 = Interrupt Disabled. 1 = Interrupt Enabled.
[23]	**RXLCH**	**Receive Left Channel Enable Bit** When monaural format is selected (MONO = 1), I²S controller will receive right channel data if RXLCH is set to 0, and receive left channel data if RXLCH is set to 1. 0 = Receive right channel data in Mono mode. 1 = Receive left channel data in Mono mode.
[22:18]	**Reserved**	Reserved.

[17]	**LZCEN**	**Left Channel Zero Cross Detection Enable Bit** If this bit is set to 1, when sign bits of the current left channel TX data and the next left channel TX data change or next TX shift data bits are all 0 then I^2S TX output data of the left channel is 0 and LZCIF flag in SPIx_I2SSTS register is set to 1. This function is only available in transmit operation. 0 = Left channel zero cross detection Disabled. 1 = Left channel zero cross detection Enabled.
[16]	**RZCEN**	**Right Channel Zero Cross Detection Enable Bit** If this bit is set to 1, when sign bits of the current right channel TX data and the next right channel TX data change or next TX shift data bits are all 0 then I^2S TX output data of the right channel is 0 and RZCIF flag in SPIx_I2SSTS register is set to 1. This function is only available in transmit operation. 0 = Right channel zero cross detection Disabled. 1 = Right channel zero cross detection Enabled.
[15]	**MCLKEN**	**Master Clock Enable Bit** If MCLKEN is set to 1, I^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.
[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 this chip 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**	**I²S Controller Enable Bit** 0 = I²S mode Disabled. 1 = I²S mode Enabled. **Note 1:** If enabling this bit, I2Sx_BCLK will start to output in Master mode. **Note 2:** Before changing the configurations of SPIx_I2SCTL, SPIx_I2SCLK, and SPIx_FIFOCTL registers, user shall clear the I2SEN (SPIx_I2SCTL[0]) and confirm the I2SENSTS (SPIx_I2SSTS[15]) is 0.

I²S Clock Divider Control Register (SPIx_I2SCLK)

Register	Offset	R/W	Description	Reset Value
SPIx_I2SCLK	SPIx_BA+0x64	R/W	I²S Clock Divider Control Register	0x0000_0000

Note: Not supported in SPI mode.

31	30	29	28	27	26	25	24
Reserved						I2SSLAVE	I2SMODE
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	
[31:26]	**Reserved**	Reserved.
[25]	**I2SSLAVE**	**I²S Clock Divider Number Selection for I²S Slave Mode and I²S Master Mode** User sets I2SSLAVE to set frequency of peripheral clock of I²S master mode and I²S slave mode when BCLKDIV (SPIx_I2SCLK[17:8]) is set. I2SSLAVE needs to be set before I2SEN (SPIx_I2SCTL[0]) is enabled. 0 = The frequency of peripheral clock sets to I²S master mode. 1 = The frequency of peripheral clock sets to I²S slave mode.
[24]	**I2SMODE**	**I²S Clock Divider Number Selection for I²S Mode and SPI Mode** User sets I2SMODE to set frequency of peripheral clock of I²S mode or SPI mode when BCLKDIV (SPIx_I2SCLK[17:8]) or DIVIDER (SPIx_CLKDIV[8:0]) are set. I2SMODE needs to be set before I2SEN (SPIx_I2SCTL[0]) or SPIEN (SPIx_CTL[0]) is enabled. 0 = The frequency of peripheral clock sets to SPI mode. 1 = The frequency of peripheral clock sets to I²S mode.
[23:18]	**Reserved**	Reserved.

[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: $f_{BCLK} = \frac{f_{i2s\_clock\_src}}{2 \times (BCLKDIV + 1)}$ 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 $f_{i2s\_clock\_src} \div \left(\frac{BCLKDIV}{2} + 1\right)$. The peripheral clock frequency in I²S Slave mode must be equal to or faster than 6 times of input bit clock. **Note:** The time interval must be larger than or equal to 8 peripheral clock cycles between releasing SPI IP software reset and setting this clock divider register.
[7]	**Reserved**	Reserved.
[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,. $f_{MCLK} = \frac{f_{i2s\_clock\_src}}{2 \times MCLKDIV}$ 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.

Note: BCLKDIV should be set carefully because the peripheral clock frequency must be slower than or equal to system frequency.

I²S Status Register (SPIx_I2SSTS)

Register	Offset	R/W	Description	Reset Value
SPIx_I2SSTS	SPIx_BA+0x68	R/W	I²S 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	SLVERRIF	LZCIF	RZCIF	TXUFIF	TXTHIF	TXFULL	TXEMPTY
15	14	13	12	11	10	9	8
I2SENSTS	Reserved		RXTOIF	RXOVIF	RXTHIF	RXFULL	RXEMPTY
7	6	5	4	3	2	1	0
Reserved			RIGHT	Reserved			

Bits	Description	
[31]	**Reserved**	Reserved.
[30:28]	**TXCNT**	**Transmit FIFO Data Count (Read Only)** This bit field indicates the valid data count of transmit FIFO buffer.
[27]	**Reserved**	Reserved.
[26:24]	**RXCNT**	**Receive FIFO Data Count (Read Only)** This bit field indicates the valid data count of receive FIFO buffer.
[23]	**TXRXRST**	**TX or RX Reset Status (Read Only)** 0 = The reset function of TXRST or RXRST is done. 1 = Doing the reset function of TXRST or RXRST. **Note:** Both the reset operations of TXRST and RXRST need 3 system clock cycles + 2 peripheral clock cycles. User can check the status of this bit to monitor the reset function is doing or done.
[22]	**SLVERRIF**	**Bit Number Error Interrupt Flag for Slave Mode** 0 = No bit number error event occurred. 1 = Bit number error event occurred. **Note:** This bit will be cleared by writing 1 to it.
[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 = SPI/I^{2}S control logic Disabled. 1 = SPI/I^{2}S control logic 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.
[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.
[4]	**RIGHT**	**Right Channel (Read Only)** This bit indicates the current transmit data is belong to which channel. 0 = Left channel. 1 = Right channel.
[3:0]	**Reserved**	Reserved.

6.16 I²C Serial Interface Controller (I²C)

6.16.1 Overview

I²C is a two-wire, bi-directional serial bus that provides a simple and efficient method of data exchange between devices. The I²C standard is a true multi-master bus including collision detection and arbitration that prevents data corruption if two or more masters attempt to control the bus simultaneously.

There are two sets of I²C controllers which support Power-down wake-up function.

6.16.2 Features

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

- Supports up to two I²C ports
- Master/Slave mode
- Bidirectional data transfer between masters and slaves
- Multi-master bus (no central master)
- 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 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 overflow
- Programmable clocks allow for versatile rate control
- Supports 7-bit addressing and 10-bit addressing mode
- Supports multiple address recognition (four slave address with mask option)
- Supports Power-down wake-up function
- Supports PDMA with one buffer capability
- Supports setup/hold time programmable
- Supports Bus Management (SM/PM compatible) function

6.16.3 Block Diagram

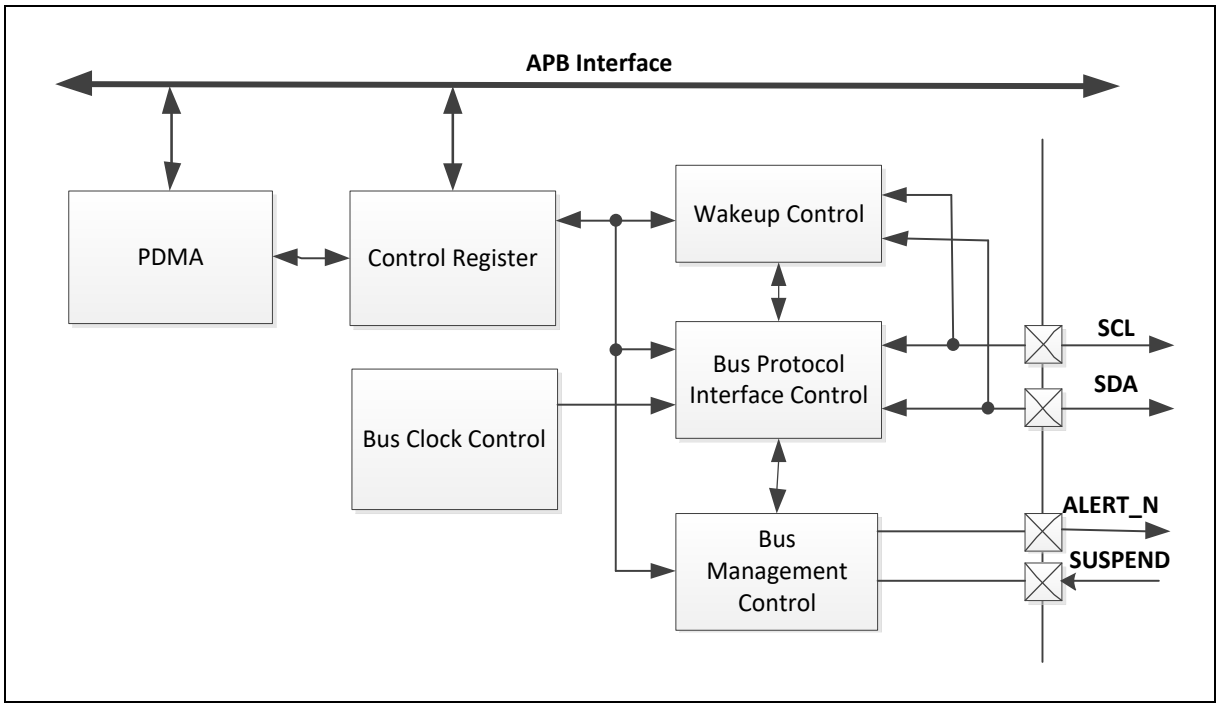


Figure 6.16-1 I²C Controller Block Diagram

6.16.4 Basic Configuration

6.16.4.1 I²C0 Basic Configuration

- Clock source Configuration
  - Enable I²C0 peripheral clock in I2C0CKEN (CLK_APBCLK0[8]).
- Reset Configuration
  - Reset I²C0 controller in I2C0RST (SYS_IPRST1[8]).

6.16.4.2 I²C1 Basic Configuration

- Clock Source Configuration
  - Enable I²C1 peripheral clock in I2C1CKEN (CLK_APBCLK0[9]).
- Reset Configuration
  - Reset I²C1 controller in I2C1RST (SYS_IPRST1[9]).

6.16.5 Functional Description

On I²C bus, data is transferred between a Master and a Slave. Data bits transfer on the SCL and SDA lines are synchronously on a byte-by-byte basis. Each data byte is 8-bit long. There is one SCL clock pulse for each data bit with the MSB being transmitted first, and an acknowledge bit follows each transferred byte. Each bit is sampled during the high period of SCL; therefore, the SDA line may be changed only during the low period of SCL and must be held stable during the high period of SCL. A transition on the SDA line while SCL is high is interpreted as a command (START or STOP). Please refer to Figure 6.16-2 for more detailed I²C BUS Timing.

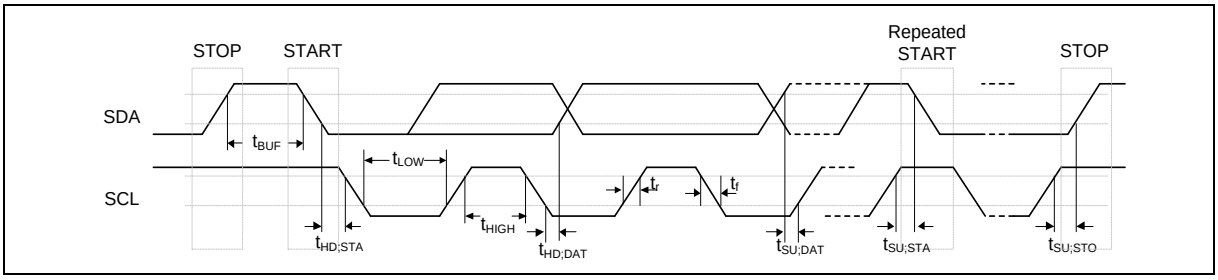


Figure 6.16-2 I²C Bus Timing

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_CTL0 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.

6.16.5.1 I²C Protocol

Figure 6.16-3 shows the typical I²C protocol. Normally, a standard communication consists of four parts:

- START or Repeated START signal generation
- Slave address and R/W bit transfer
- Data transfer
- STOP signal generation

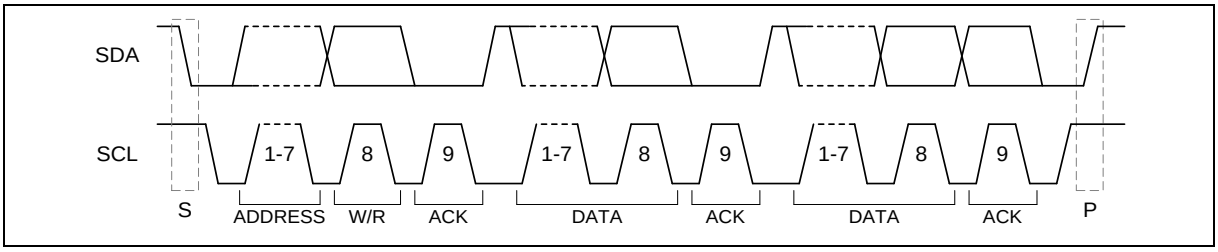


Figure 6.16-3 I²C Protocol

- START or Repeated START signal

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

  After having sent the address byte (address and read/write bit), the master may send any number of bytes followed by a stop condition. Instead of sending the stop condition it is also allowed to send another start condition again followed by an address (and of course including a read/write bit) and more data. The start condition is called as Repeat START (Sr). This is defined recursively allowing any number of start conditions to be sent. The purpose of this is to allow combined write/read operations to one or more devices without releasing the bus and thus with the guarantee that the operation is not interrupted. The controller uses this method to communicate with another slave or the same slave in a different transfer direction (e.g. from writing to a device to reading from a device) without releasing the bus.

Note: Repeat START (Sr) is sent immediately when I²C triggers Repeat START signal. It should wait 0.5 I²C clock cycle to trigger Repeat START signal. For example, I²C 100 kHz => 5 uS delay, 400 kHz => 1.25 uS.

- STOP signal

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

  Figure 6.16-4 shows the waveform of START, Repeat START and STOP.

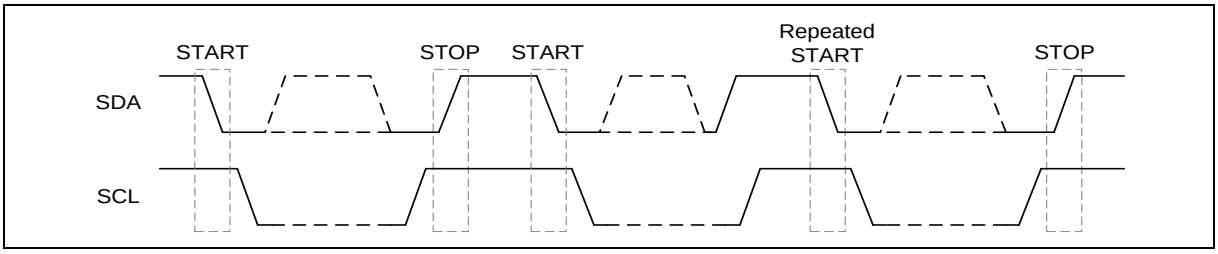


Figure 6.16-4 START and STOP Conditions

- Slave Address Transfer

  After a (Repeated) START condition, the master sends a slave address to identify the target device of the communication. The start address can comprise one or two address bytes (for 7-bit or for 10-bit addressing schemes). After an address byte, a slave sensitive to the transmitted address has to acknowledge the reception.

  Therefore, the slave's address can be programmed in the device, where it is compared to the received address. In case of a match, the slave answers with an acknowledge (SDA = 0). Slaves that are not targeted answer with a non-acknowledge (SDA = 1). In addition to the match of the programmed address, another address byte value has to be answered with an acknowledge if the slave is capable to handle the corresponding requests.

- 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 6.16-5 and Figure 6.16-6 shows the waveform of bit transfer and acknowledge.

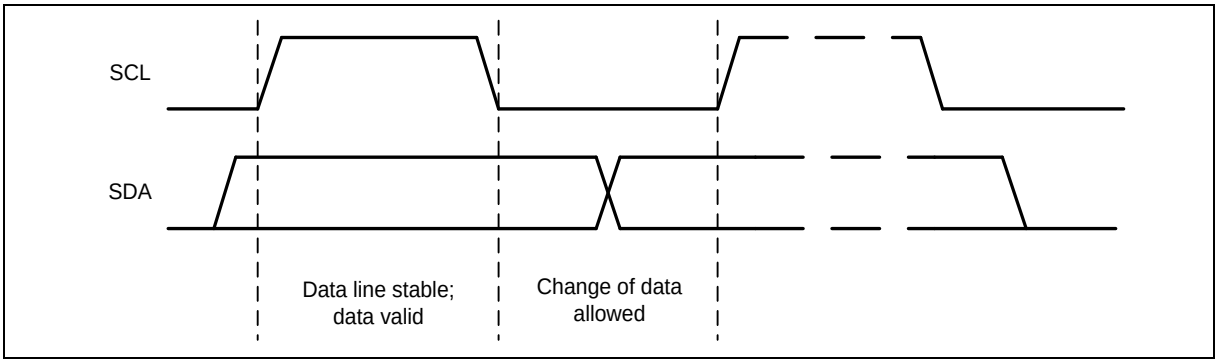


Figure 6.16-5 Bit Transfer on the I²C Bus

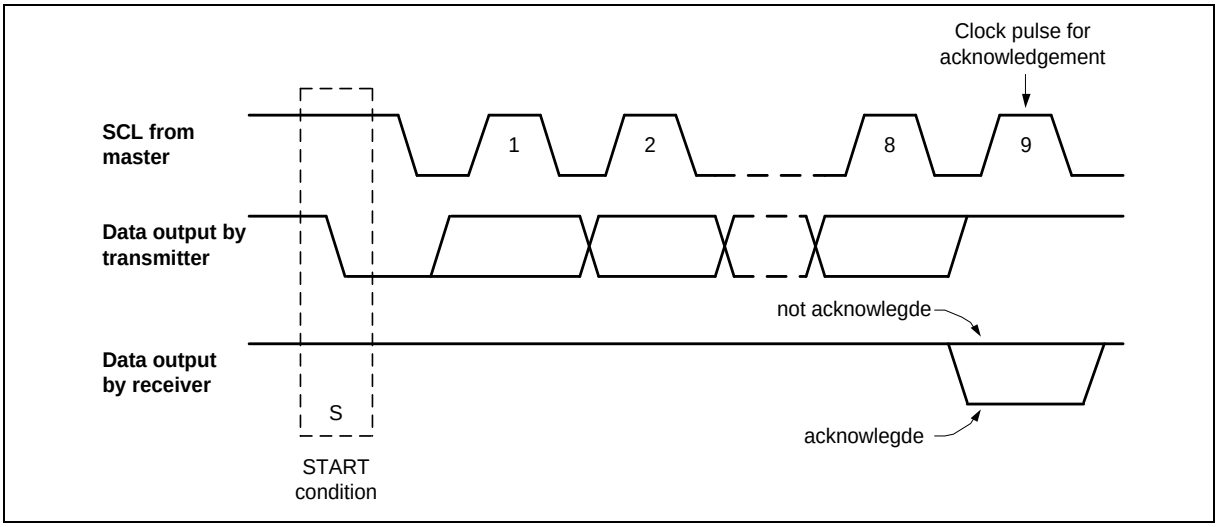


Figure 6.16-6 Acknowledge on the I²C Bus

- Data transfer on I²C bus

  Figure 6.16-7 shows a master transmits data to slave by 7-bit. A master addresses a slave with a 7-bit address and 1-bit write index to denote that the master wants to transmit data to the slave. The master keeps transmitting data after the slave returns acknowledge to the master.

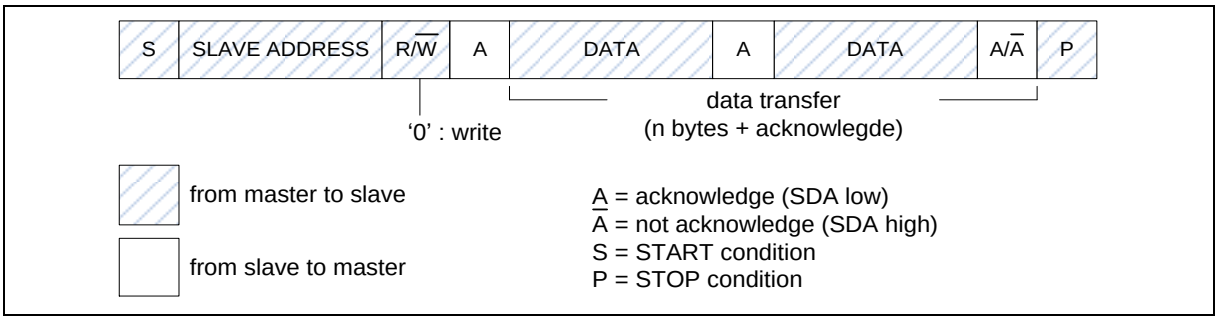


Figure 6.16-7 Master Transmits Data to Slave by 7-bit

Figure 6.16-8 shows a master read data from slave by 7-bit. A master addresses a slave with a 7-bit address and 1-bit read index to denote that the master wants to read data from the slave. The slave will start transmitting data after the slave returns acknowledge to the master.

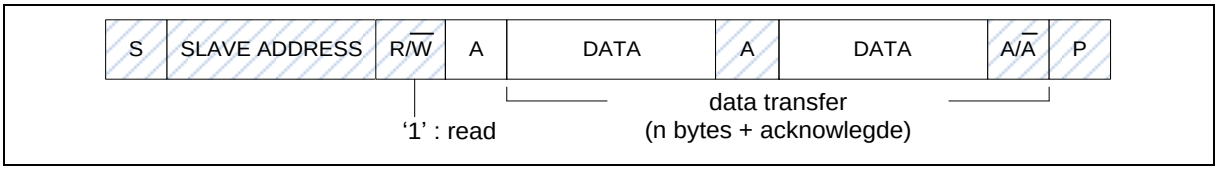


Figure 6.16-8 Master Reads Data from Slave by 7-bit

Figure 6.16-9 shows a master transmits data to slave by 10-bit. A master addresses a slave with a 10-bit address. First byte contains 10-bit address indicator (5'b11110) and 2-bit address with write index, second byte contains 8-bit address. The master keeps transmitting data after the second byte end. Note that 7-bit and 10-bit address device can work on the same bus.

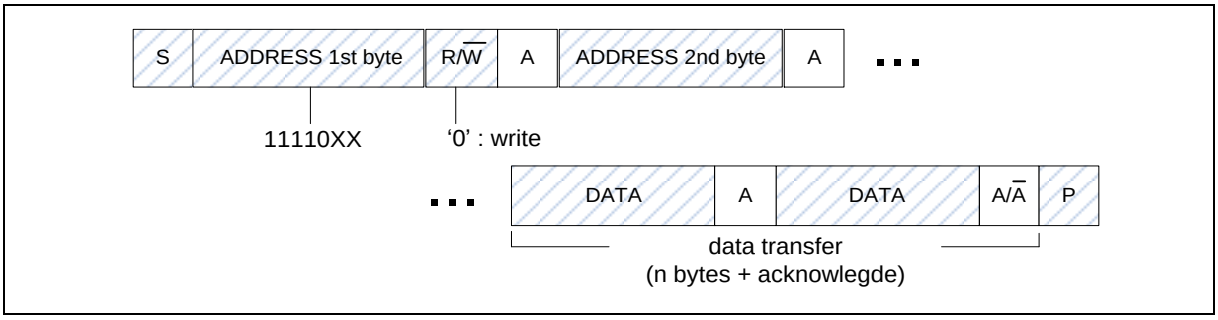


Figure 6.16-9 Master Transmits Data to Slave by 10-bit

Figure 6.16-10 shows a master read data from slave by 10-bit. A master addresses a slave with a 10-bit address. First mater transmits 10-bit address to slave, after that master transmits first byte with read index. The slave will start transmitting data after the first byte with read index.

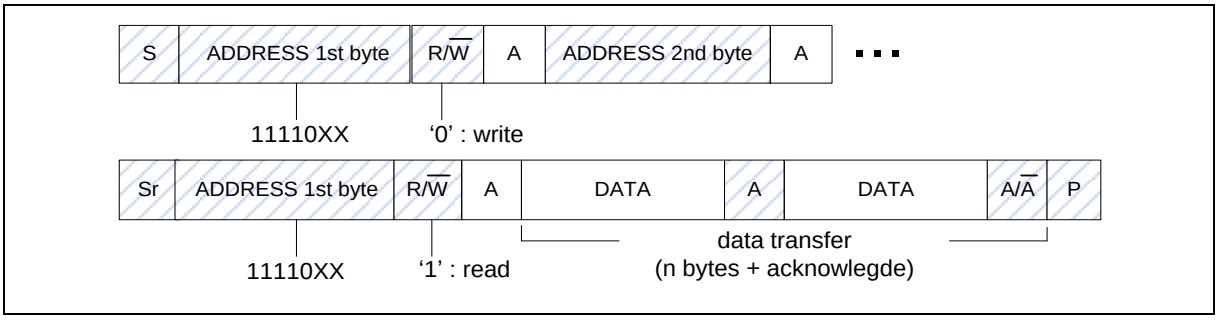


Figure 6.16-10 Master Reads Data from Slave by 10-bit

6.16.5.2 Operation Modes

The on-chip I²C ports support three operation modes, Master, Slave, and General Call Mode.

In a given application, I²C port may operate as a master or as a slave. In Slave mode, the I²C port hardware looks for its own slave address and the general call address. If one of these addresses is detected, and if the slave is willing to receive or transmit data from/to master(by setting the AA(I2C_CTL0[2]) bit), acknowledge pulse will be transmitted out on the 9th clock, hence an interrupt is requested on both master and slave devices if interrupt is enabled. When the microcontroller wishes to become the bus master, hardware waits until the bus is free before entering Master mode so that a possible slave action is not be interrupted. If bus arbitration is lost in Master mode, I²C port switches to Slave mode immediately and can detect its own slave address in the same serial transfer.

To control the I²C bus transfer in each mode, user needs to set I2C_CTL0, I2C_DAT registers according to current status code of I2C_STATUS0 register. In other words, for each I²C bus action, user needs to check current status by I2C_STATUS0 register, and then set I2C_CTL0, I2C_DAT registers to take bus action. Finally, check the response status by I2C_STATUS0.

The bits, STA, STO and AA in I2C_CTL0 register are used to control the next state of the I²C hardware after SI flag of I2C_CTL0 [3] register is cleared. Upon completion of the new action, a new status code will be updated in I2C_STATUS0 register and the SI flag of I2C_CTL0 register will be set. But the SI flag will not be set when I²C STOP. If the I²C interrupt control bit INTEN (I2C_CTL0 [7]) is set, appropriate action or software branch of the new status code can be performed in the Interrupt service routine.

Figure 6.16-11 shows the current I²C status code is 0x08, and then set I2C_DATA=SLA+W and (STA,STO,SI,AA) = (0,0,1,x) to send the address to I²C bus. If a slave on the bus matches the address and response ACK, the I2C_STATUS0 will be updated by status code 0x18.

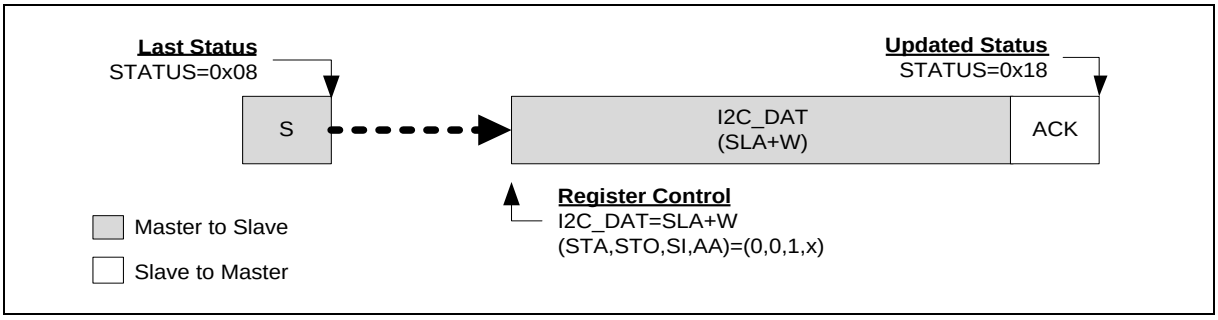


Figure 6.16-11 Control I²C Bus according to the Current I²C Status

Master Mode

In Figure 6.16-12 and Figure 6.16-13, all possible protocols for I²C master are shown. User needs to follow proper path of the flow to implement required I²C protocol.

In other words, user can send a START signal to bus and I²C will be in Master Transmitter (MT) mode (Figure 6.16-12) or Master receiver (MR) mode (Figure 6.16-13) after START signal has been sent successfully and new status code would be 0x08. Followed by START signal, user can send slave address, read/write bit, data and Repeat START, STOP to perform I²C protocol.

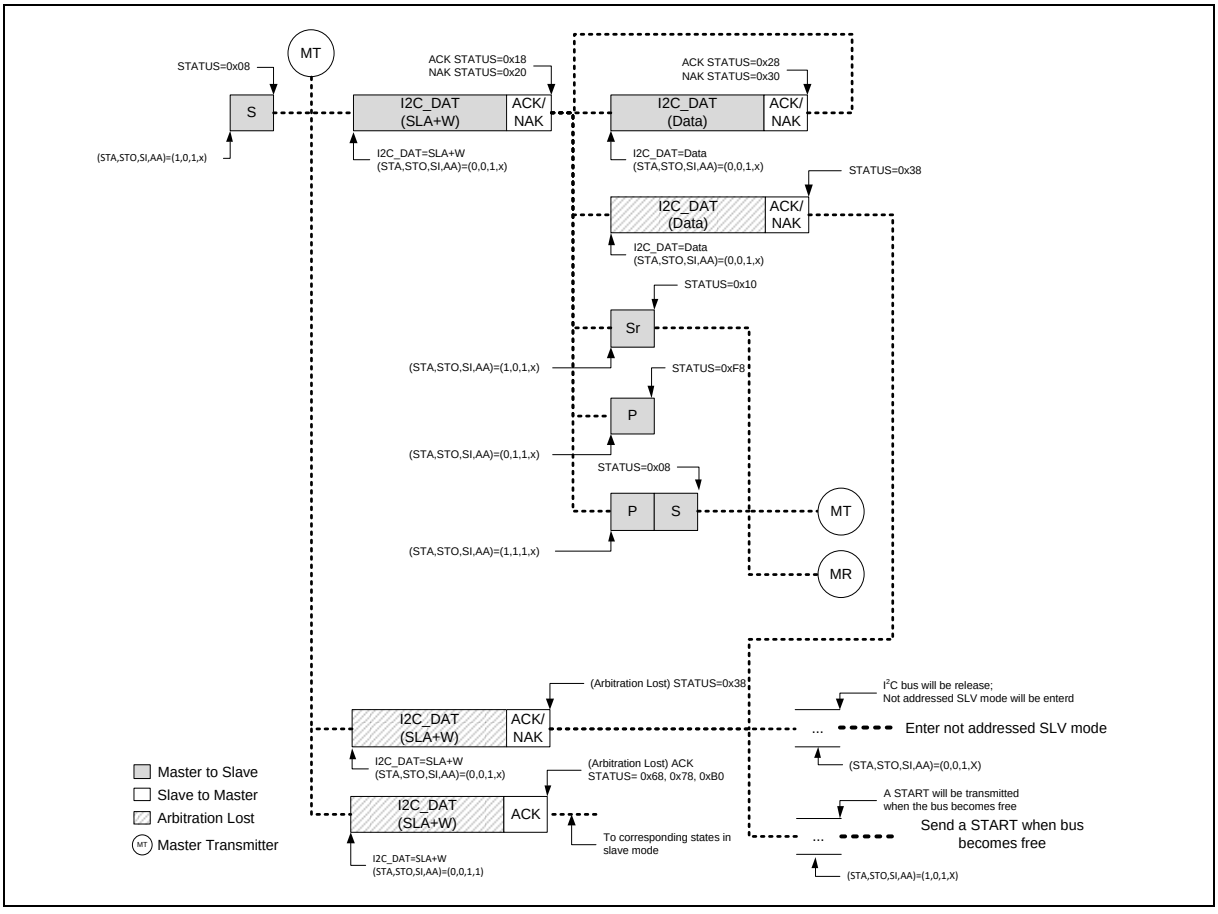


Figure 6.16-12 Master Transmitter Mode Control Flow

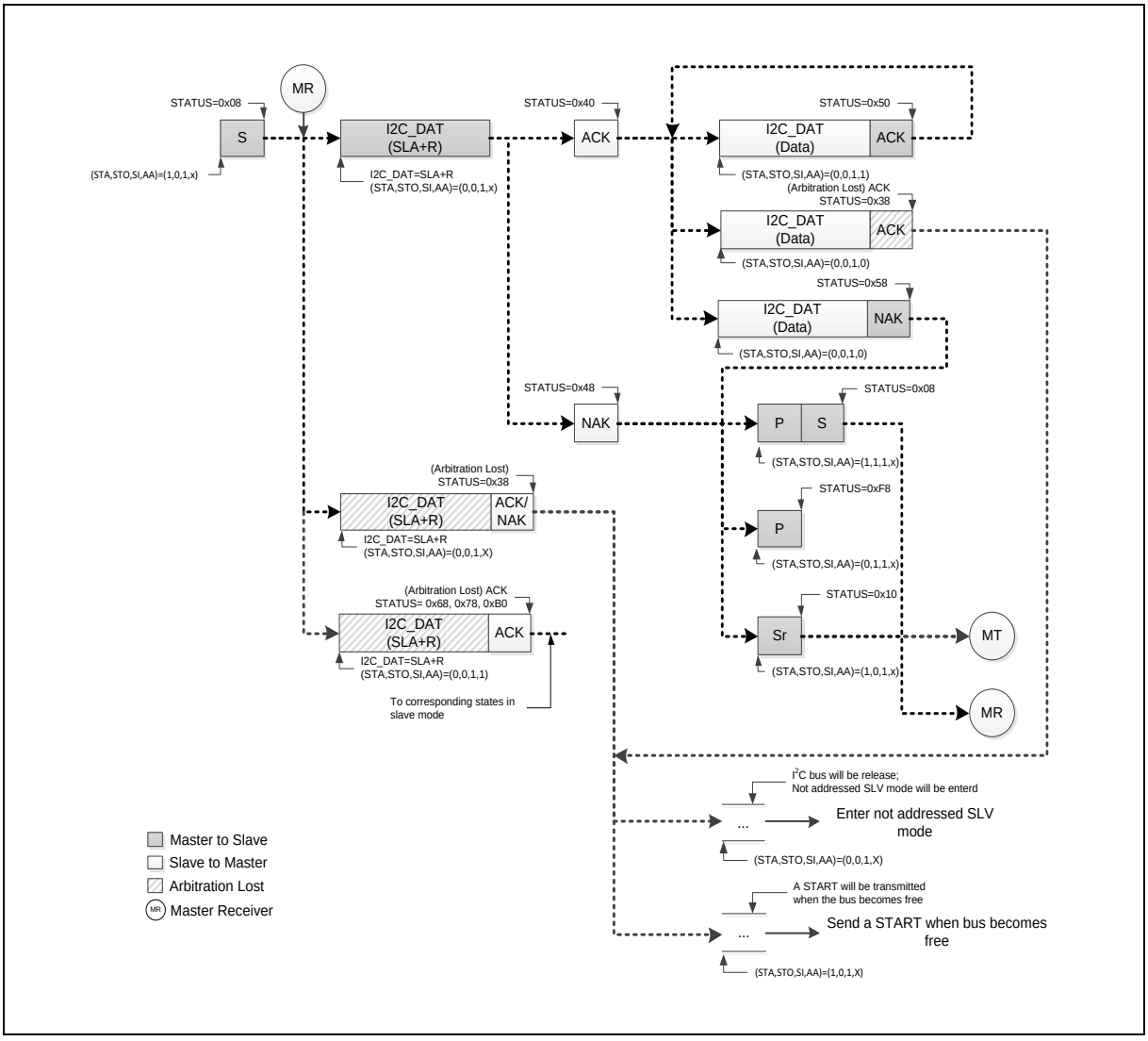


Figure 6.16-13 Master Receiver Mode Control Flow

If the I²C is in Master mode and gets arbitration lost, the status code will be 0x38. In status 0x38, user may set (STA, STO, SI, AA) = (1, 0, 1, X) to send START to re-start Master operation when bus become free. Otherwise, user may set (STA, STO, SI, AA) = (0, 0, 1, X) to release I²C bus and enter not addressed Slave mode.

Slave Mode

When reset default, I²C is not addressed and will not recognize the address on I²C bus. User can set slave address by I2C_ADDRn (n=0~3) and set (STA, STO, SI, AA) = (0, 0, 1, 1) to let I²C recognize the address sent by master. Figure 6.16-14 shows all the possible flow for I²C in Slave mode. Users need to follow a proper flow (as shown in Figure 6.16-14 to implement their own I²C protocol.

If bus arbitration is lost in Master mode, I²C port switches to Slave mode immediately and can detect its own slave address in the same serial transfer. If the detected address is SLA+W (Master want to write data to Slave) after arbitration lost, the status code is 0x68. If the detected address is SLA+R (Master want to read data from Slave) after arbitration lost, the status code is 0xB0.

Note: During I²C communication, the SCL clock will be released when writing '1' to clear SI flag in Slave mode.

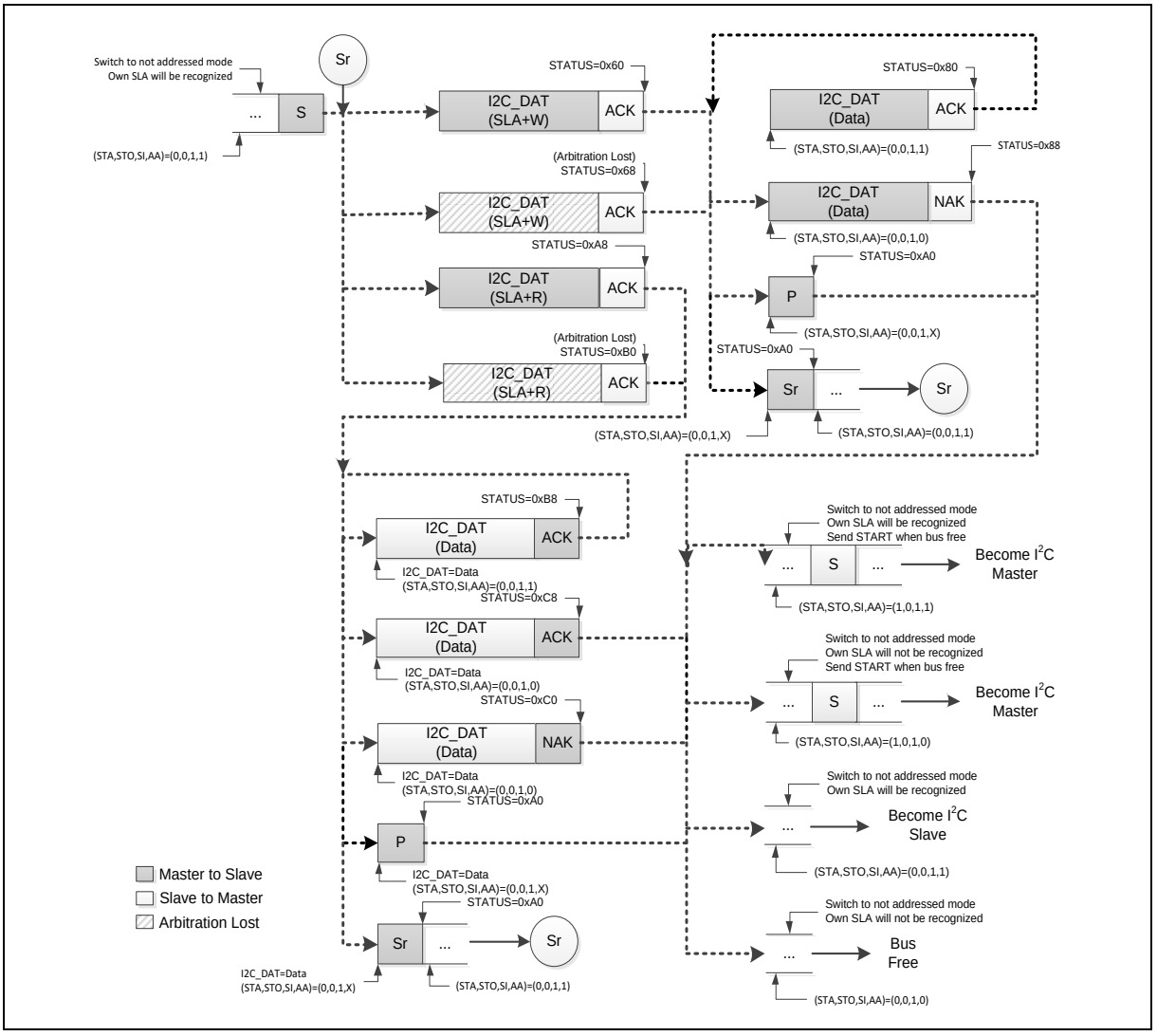


Figure 6.16-14 Slave Mode Control Flow

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 I²C 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 I²C signal or address from master. At this status, I²C should enter idle mode.

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. Note that the default address is 0x00, but user must set an address except for 0x00.

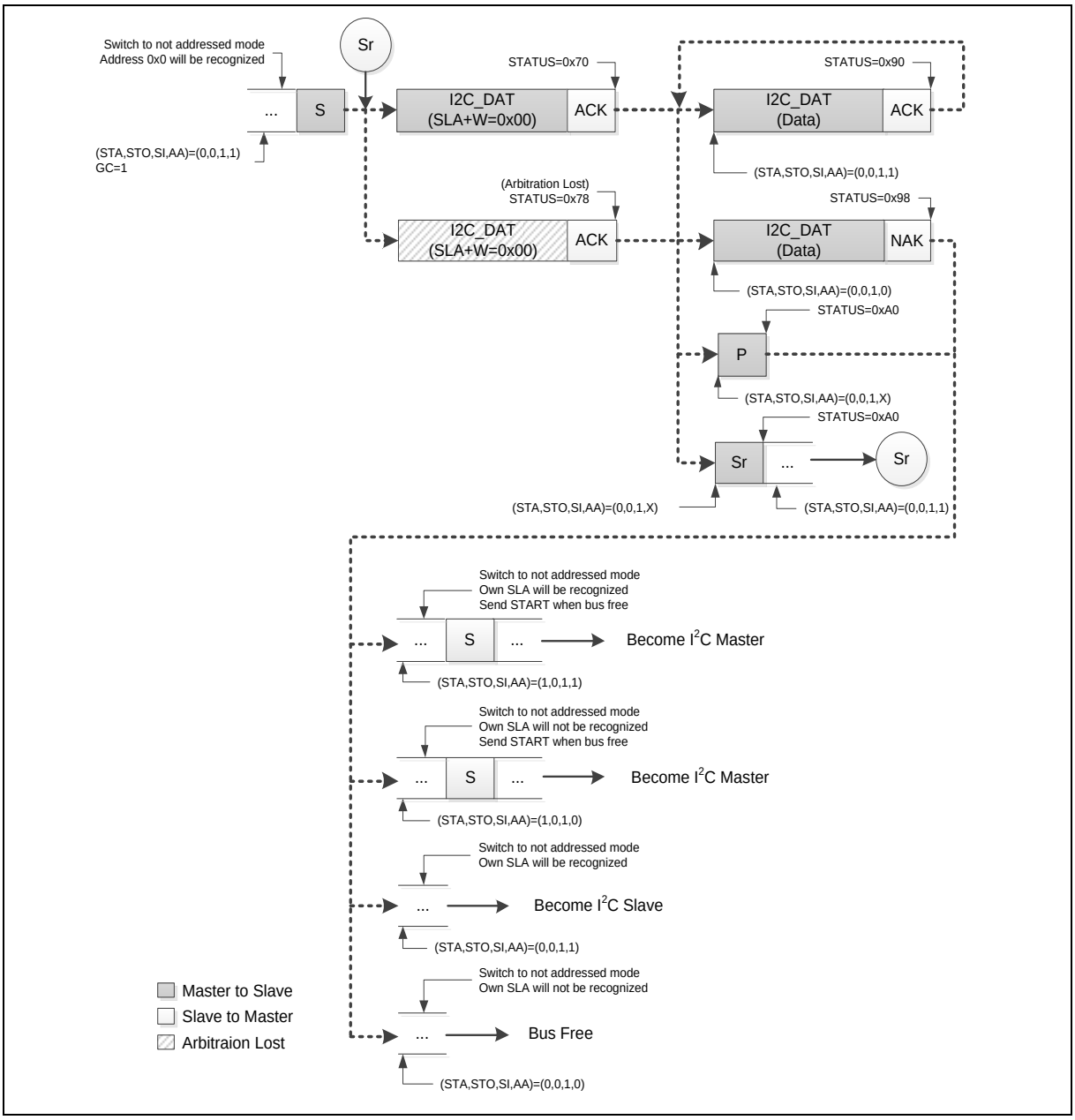


Figure 6.16-15 GC Mode

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 I²C signal or address from master. At this time, the I²C controller should enter idle mode.

Multi-Master

In some applications, there are two or more masters on the same I²C bus to access slaves, and the masters may transmit data simultaneously. The I²C 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.

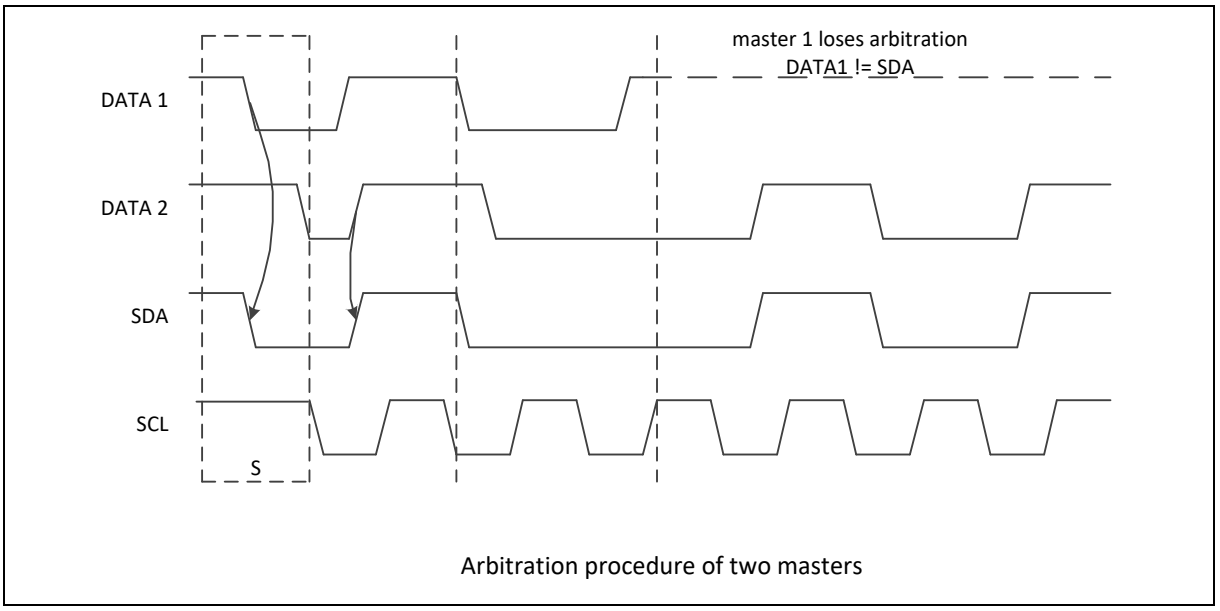


Figure 6.16-16 Arbitration Lost

- When I2C_STATUS0 = 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_STATUS0 = 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

Bus Management (SMBus/PMBus Compatiable)

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 6.16-1 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)

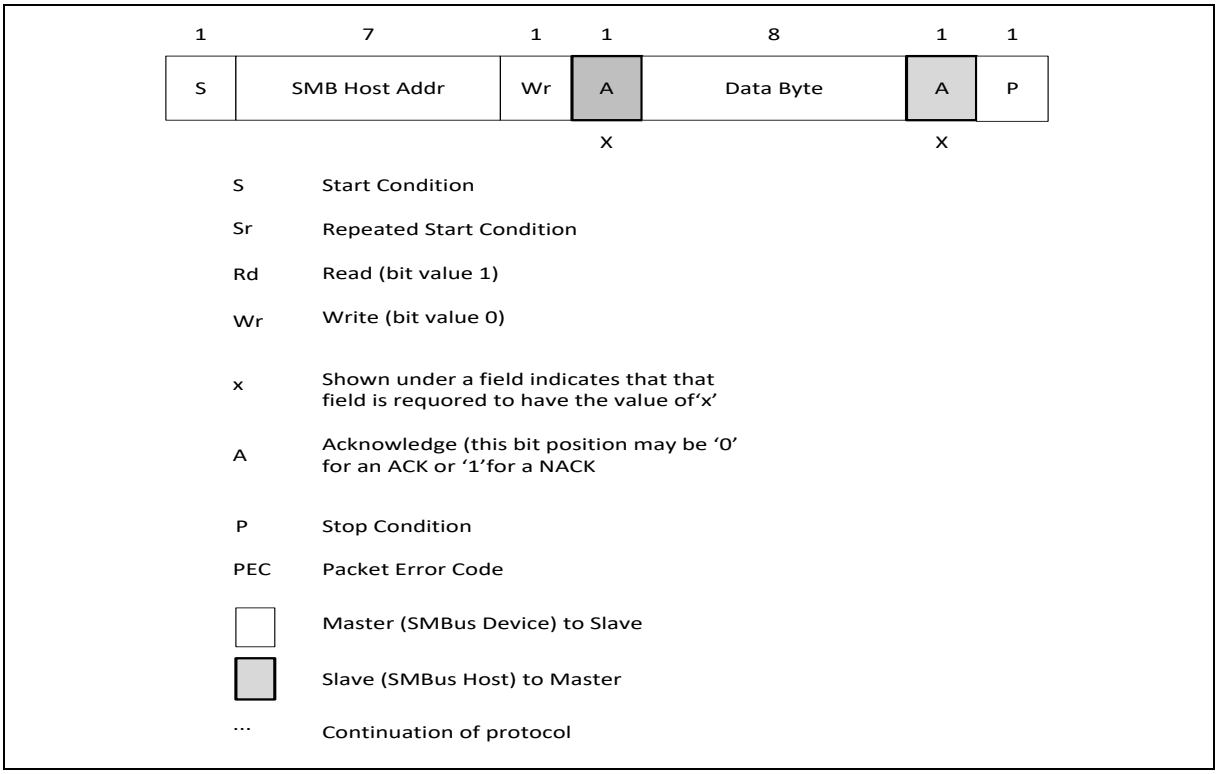


Figure 6.16-17 Bus Management Packet Protocol Diagram Element Key

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 (0b0001000). This protocol is used when the device acts as a master and the host as a slave.

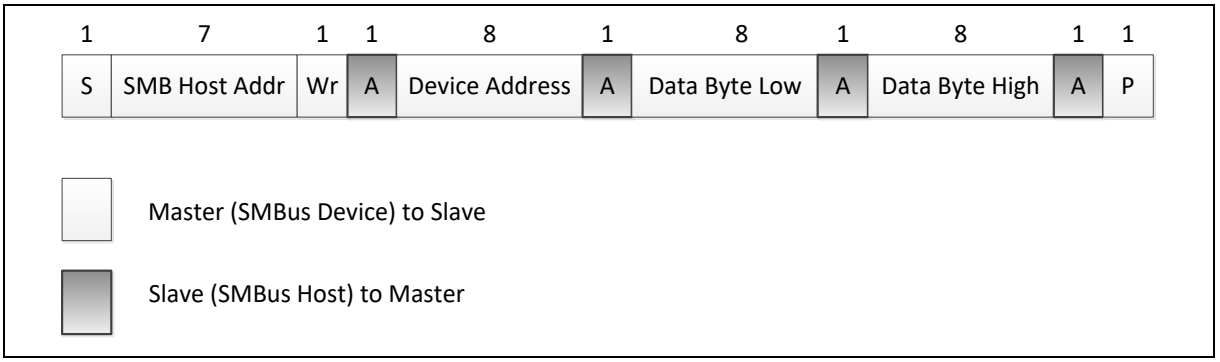


Figure 6.16-187-bit Addressable Device to Host Communication

Bus Management Alert

The Bus Management ALERT optional signal is supported. A slave-only device can signal the host through the Bus Management ALERT 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 (0b0001 100). 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.

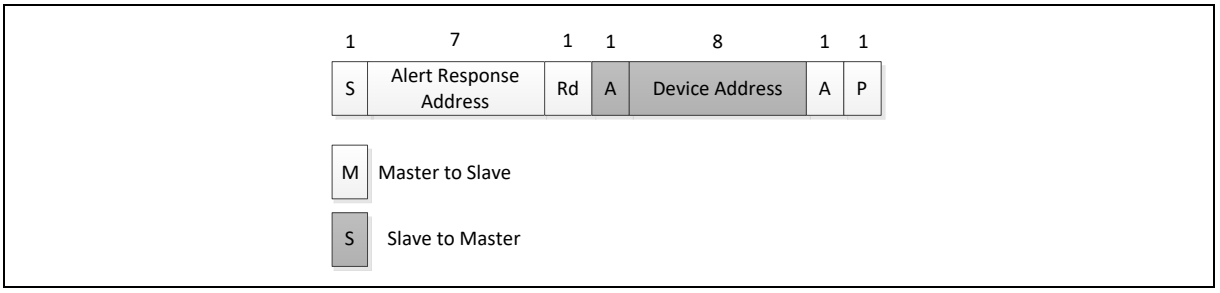


Figure 6.16-197-bit Addressable Device Responds to an ARA

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;

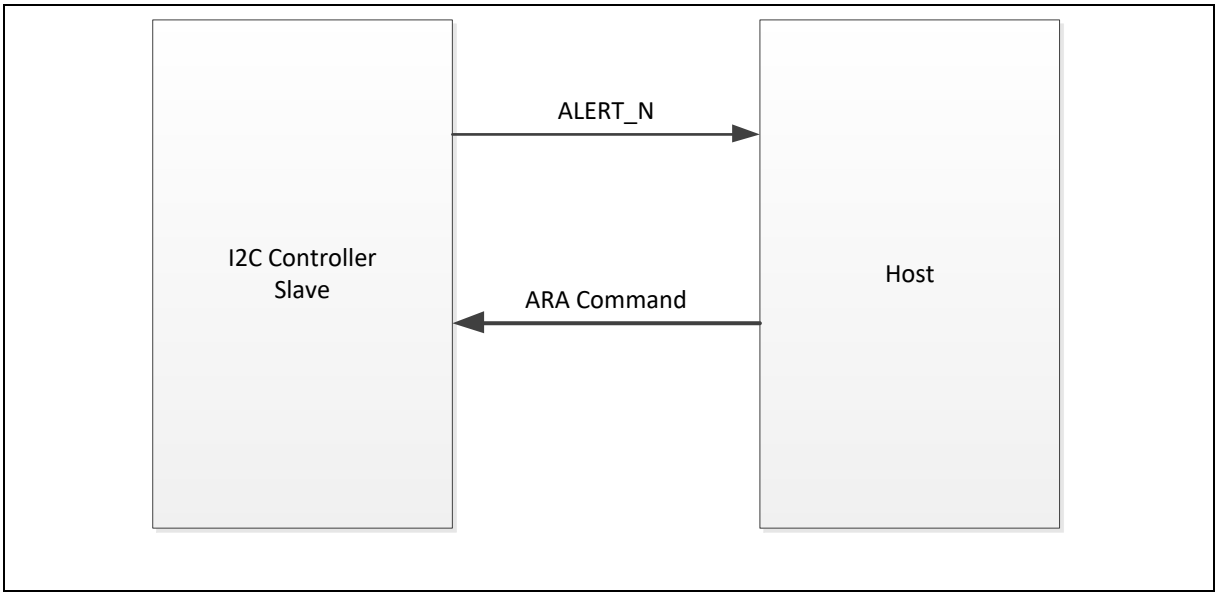


Figure 6.16-20 Bus Management ALERT Function

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) = x8 + x2 + 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:

The SCLK low time-out condition when bus no IDLE

$T_{Time\text{-}out}$ = (BUSTO(I2C_BUSTOUT[7:0]) +1) x 16x1024 (14-bit) x T_{PCLK} (if TOCDIV4 = 0).

= (BUSTO(I2C_BUSTOUT[7:0])+1) x 16x1024 (14-bit) x 4 x T_{PCLK} (if TOCDIV4 = 1)

The bus idle condition (both SCLK and SDA high) when bus IDLE

$T_{Time\text{-}out}$ = (BUSTO(I2C_BUSTOUT[7:0])+ 1) x 4 x T_{PCLK}.

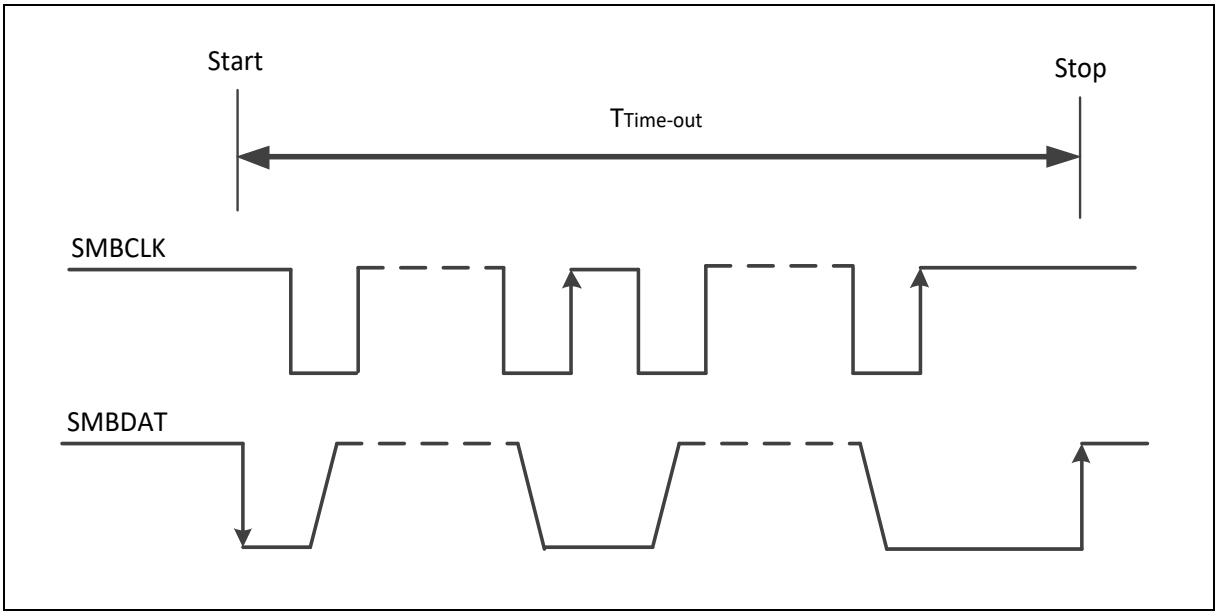


Figure 6.16-21 Bus Management Time Out Timing

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])+1) x 16x1024 (14-bit) x T_{PCLK} (if TOCDIV4= 0).

= (CLKTO (I2C_CLKTOUT[7:0])+1) x 16x1024 (14-bit) x 4 x T_{PCLK} (if TOCDIV4= 1)

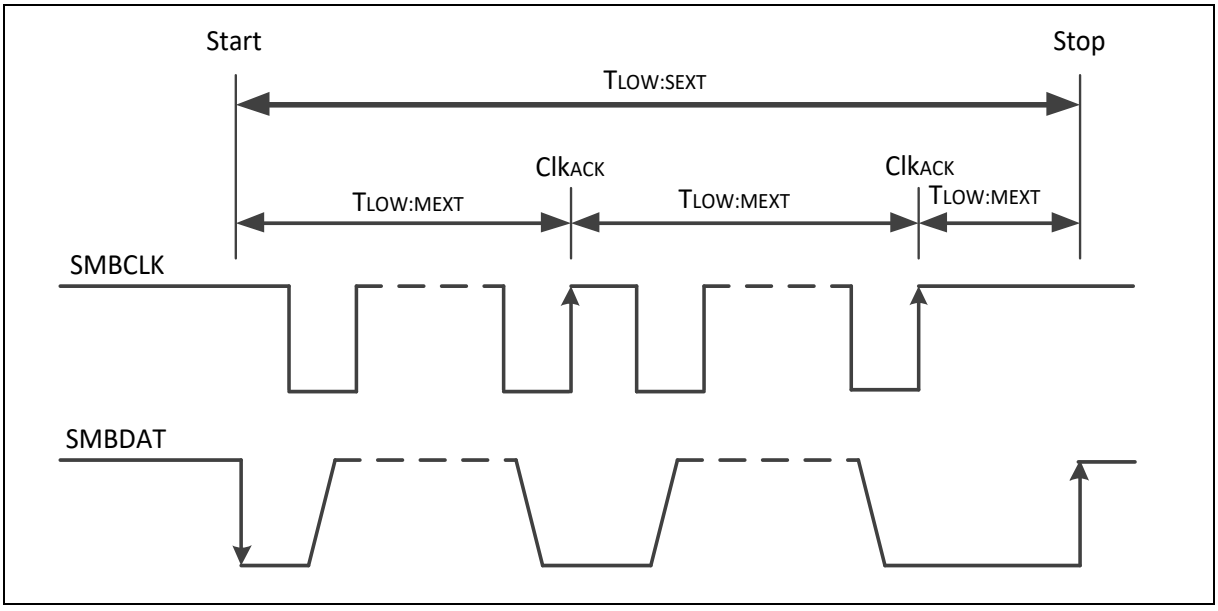


Figure 6.16-22 Bus Clock Low Time Out Timing

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.

6.16.5.3 PDMA Transfer Function

The I²C controller supports PDMA transfer function. When TXPDMAEN (I2C_CTL1 [0]) is set to 1, the I²C controller will issue request to PDMA controller to start the DMA transmission process automatically.

When RXPDMAEN (I2C_CTL1 [1]) is set to 1, the I²C controller will start the receive PDMA process. The I²C controller will issue the request to PDMA controller automatically when there is data written into the received BUFFER.

When I²C enters PDMA mode, the mostly status interrupt will be masked. Let the interrupt not occur besides the bus error or NACK or STOP interrupt (0x20, 0x30, 0x38, 0x48, 0x58, 0x00, 0xA0, 0xC0, 0x88 and 0x98).

Set the PDMASTR (I2C_CTL1 [8]) only the I²C controller in master TX mode. If PDMASTR is cleared to 0, I²C will send STOP automatically after PDMA transfer done and buffer empty. If PDMASTR is set to 1, SI will be set to 1 and I²C bus will be stretched by hardware after PDMA transfer done and buffer empty.

6.16.5.4 Programmable setup and hold times

To guarantee a correct data setup and hold time, the timing must be configured. By programming HTCTL (I2C_TMCTL[24:16]) to configure hold time and STCTL (I2C_TMCTL[8:0]) 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 I²C protocol. Once setup time configuration greater than design limitation, that means if setup time setting make SCL output less than three PCLKs, the I²C controller can't work normally due to SCL must sample three times. And once hold time configuration greater than I²C clock limitation, I²C will occur bus error. It is suggested that user calculate suitable timing with baud rate and protocol before setting timing. Table 6.16-2 shows the relationship between I²C baud rate and PCLK, the number of table represent one clock duty contain how many PCLKs. Setup and hold time configuration even can program some extreme values in the design, but user should follow I²C protocol standard.

I²C 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 6.16-2 Relationship between I²C Baud Rate and PCLK

For setup time wrong adjustment example, assuming one SCL cycle contains 5 PCLKs and set STCTL (I2C_TMCTL[8:0]) to 3 that stretch three PCLKs for setup time setting. The setup time maximum setting value: ST_{limit} = (I2C_CLKDIV[7:0]+1) X 2 - 6.

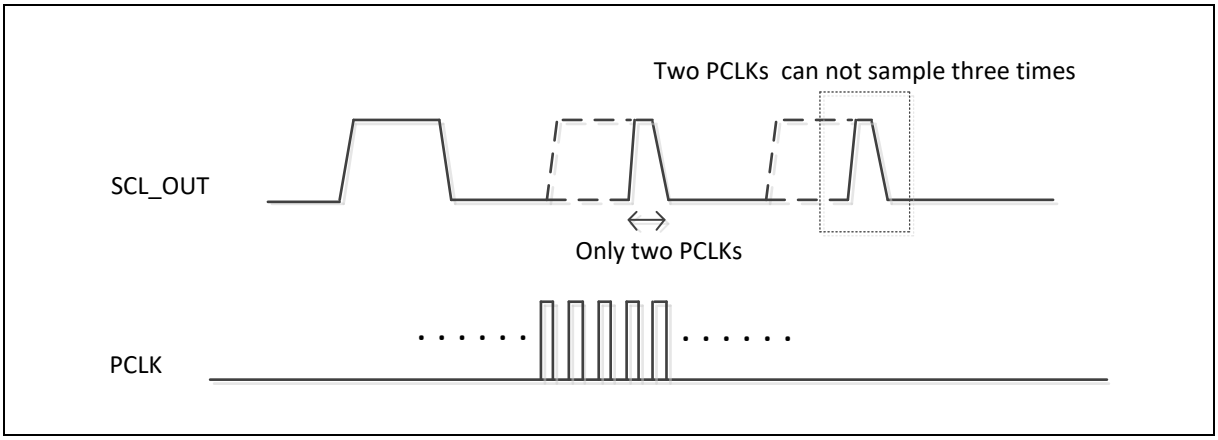


Figure 6.16-23 Setup Time Wrong Adjustment

For hold time wrong adjustment example, use I²C Baud Rate = 1200k and PCLK = 72 MHz, the SCL high/low duty = 60 PCLK. When HTCTL (I2C_TMCTL[24:16]) is set to 61 and STCTL (I2C_TMCTL[8:0]) is set to 0, then SDA output delay will over SCL high duty and cause bus error. The hold time maximum setting value: HT_{limit} = (I2C_CLKDIV[7:0]+1) X 2 - 9.

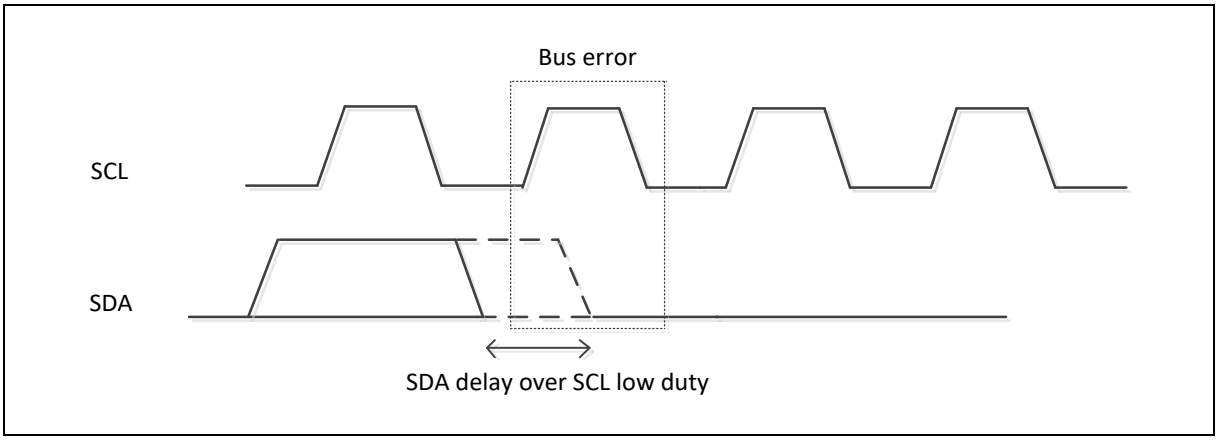


Figure 6.16-24 Hold Time Wrong Adjustment

6.16.5.5 I²C Protocol Registers

To control I²C port through the following fifteen special function registers: I2C_CTL0 (control register), I2C_STATUS0 (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).

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.

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.

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 I²C hardware and cannot be accessed by the CPU. Serial data is shifted into I2C_DAT [7:0] on the rising edges of serial clock pulses on the SCL line. When a byte has been shifted into I2C_DAT [7:0], the serial data is available in I2C_DAT [7:0], and the acknowledge bit (ACK or NACK) is returned by the control logic during the ninth clock pulse. In order to monitor bus status while sending data, the bus data will be shifted to I2C_DAT[7:0] when sending I2C_DAT[7:0] to bus. In the case of sending data, serial data bits are shifted out from I2C_DAT [7:0] on the falling edge of SCL clocks, and is shifted to I2C_DAT [7:0] on the rising edge of SCL clocks. Figure 6.16-25 shows I²C Data Shifting Direction.

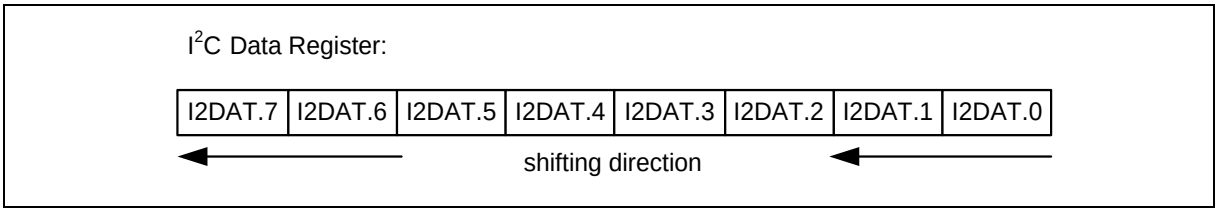


Figure 6.16-25 I²C Data Shifting Direction

Control Register (I2C_CTL0)

The CPU can be read from and written to I2C_CTL0 [7:0] directly. When the I²C port is enabled by setting I2CEN (I2C_CTL0 [6]) to high, the internal states will be controlled by I2C_CTL0 and I²C logic hardware.

There are two bits are affected by hardware: the SI bit is set when the I²C hardware requests a serial interrupt, and the STO bit is cleared when a STOP condition is present on the bus. The STO bit is also cleared when I2CEN = 0.

Once a new status code is generated and stored in I2C_STATUS0, the I²C Interrupt Flag bit SI (I2C_CTL0 [3]) will be set automatically. If the Enable Interrupt bit INTEN (I2C_CTL0 [7]) is set at this time, the I²C interrupt will be generated. The bit field I2C_STATUS0[7:0] stores the internal state code, the content keeps stable until SI is cleared by software.

Status Register (I2C_STATUS0)

I2C_STATUS0 [7:0] is an 8-bit read-only register. The bit field I2C_STATUS0 [7:0] contains the status code and there are 26 possible status codes. All states are listed in Table 6.16-3. When I2C_STATUS0 [7:0] is F8H, no serial interrupt is requested. All other I2C_STATUS0 [7:0] values correspond to the defined I²C states. When each of these states is entered, a status interrupt is requested (SI = 1). A valid status code is present in I2C_STATUS0[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 I²C format frame. A Bus Error may occur during the serial transfer of an address byte, a data byte or an acknowledge bit. To recover I²C from bus error, STO should be set and SI should be cleared to enter Not Addressed Slave mode. Then STO is cleared to release bus and to wait for a new communication. The I²C bus cannot recognize stop condition during

this action when a bus error occurs.

Master Mode		Slave Mode	
STATUS	Description	STATUS	Description
0x08[1]	Start	0xA0	Slave Transmit Repeat Start or Stop
0x10[1]	Master Repeat Start	0xA8[1]	Slave Transmit Address ACK
0x18[1]	Master Transmit Address ACK	0xB8[1]	Slave Transmit Data ACK
0x20	Master Transmit Address NACK	0xC0	Slave Transmit Data NACK
0x28[1]	Master Transmit Data ACK	0xC8[1]	Slave Transmit Last Data ACK
0x30	Master Transmit Data NACK	0x60[1]	Slave Receive Address ACK
0x38	Master Arbitration Lost	0x68[1]	Slave Receive Arbitration Lost
0x40[1]	Master Receive Address ACK	0x80[1]	Slave Receive Data ACK
0x48	Master Receive Address NACK	0x88	Slave Receive Data NACK
0x50[1]	Master Receive Data ACK	0x70[1]	GC mode Address ACK
0x58	Master Receive Data NACK	0x78[1]	GC mode Arbitration Lost
0x00	Bus error	0x90[1]	GC mode Data ACK
		0x98	GC mode Data NACK
		0xB0[1]	Address Transmit Arbitration Lost
0xF0	If the BMDEN =1 and the ACKMEN bit is enabled, the information of I2C_STATUS0 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. Note [1]: No interrupt in PDMA mode		

Table 6.16-3 I²C Status Code Description

Clock Baud Rate Bits (I2C_CLKDIV)

The data baud rate of I²C is determines by DIVIDER(I2C_CLKDIV [7:0]) register when I²C is in Master mode, and it is not necessary in a Slave mode. In the Slave mode, I²C will automatically synchronize it with any clock frequency from master I²C device. In the slave mode, system clock frequency should greater than I²C bus maximum clock 20 times.

The data baud rate of I²C setting is Data Baud Rate of I²C = (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 I²C = 16 MHz/ (4x (40 +1)) = 97.5 Kbits/sec.

Time-out Control Register (I2C_TOCTL)

There is a 14-bit time-out counter which can be used to deal with the I²C bus hang-up. If the time-out counter is enabled, the counter starts up counting until it overflows (TOIF=1) and generates I²C 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 I²C bus hangs up, it causes the I2C_STATUS0 and flag SI are not updated for a period, the 14-bit time-out counter may overflow and acknowledge CPU the I²C interrupt. Refer to Figure 6.16-26 for the 14-bit time-out counter. User may write 1 to clear TOIF to 0.

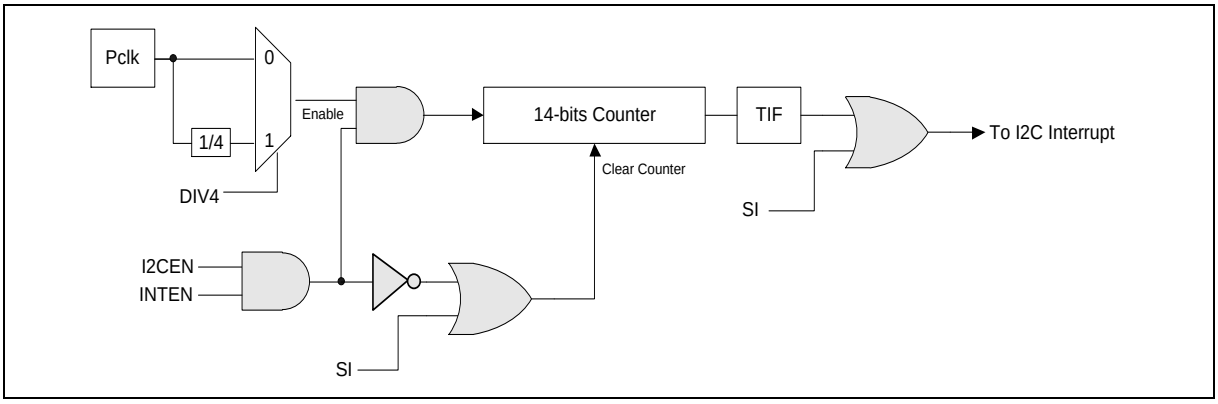


Figure 6.16-26 I²C Time-out Count Block Diagram

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 the chip by addressing the 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 the I²C controller will go ahead. If NHDBUSEN (I2C_WKCTL [7]) is set, the controller will don't stretch the SCL to low. Note 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 the I²C controller and execute the original operation again.

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. Note 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 sends address. The user can read this bit's status to prepare the next transmitted data (WRSTSWK = 1) or to wait the incoming data (WRSTSWK = 0) can be stored in time after the system is wake-up by the address match frame. Note that the WRSTSWK (I2C_WKSTS [2]) bit is cleared when writing 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 to write "1" to clear this bit.

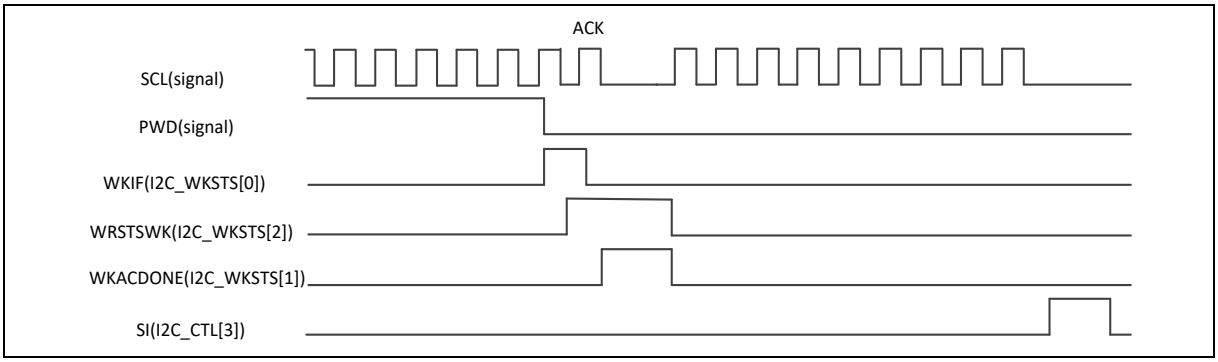


Figure 6.16-27 I²C Wake-Up Related Signals Waveform

I²C Control Register 1 (I2C_CTL1)

If enable 10-bit addressing mode ADDR10EN (I2C_CTL1 [9]) is set, the I²C will run in 10-bit mode.

For PDMA function, set TXPDMAEN (I2C_CTL1 [0]) and RXPDMAEN (I2C_CTL1 [1]) can be set to operate. And set PDMARST (I2C_CTL1 [2]) to reset the PDMA control logic.

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.

I²C Timing Configure Control Register (I2C_TMCTL)

In order to configure setup/hold time, the HTCTL (I2C_TMCTL[24:16]) and STCTL (I2C_TMCTL[8:0]) are set based on actual demand.

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 system bus management enables control by BUSEN(I2C_BUSCTL[7]) bit. The TIDLE(I2C_BUSCTL[9]) is used to calculate the time-out of clock low in bus active and the idle period in bus Idle.

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.

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.

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 SUSCON input status.

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.

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 in the Bus Management (SMBus/PMBus Compatiable).

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 Bus Management (SMBus/PMBus Compatiable).

6.16.5.6 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 as SCL and SDA pins.
2. Enable I²C0 APB clock. Refer to the Basic Configuration section for the clock configuration.
3. Set I2C0RST=1 to reset I²C0 controller then set I²C0 controller to normal operation. Refer to the Basic Configuration section for the reset controller configuration.
4. Set I2CEN=1 to enable I²C0 controller in the “I2C_CTL0” register.
5. Give I²C0 clock a divided register value for I²C clock rate in the “I2C_CLKDIV”.
6. Enable system I²C0 IRQ in system “NVIC” control register.
7. Set INTEN=1 to enable I²C0 Interrupt in the “I2C_CTL0” 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 6.16-28 shows the EEPROM random read operation.

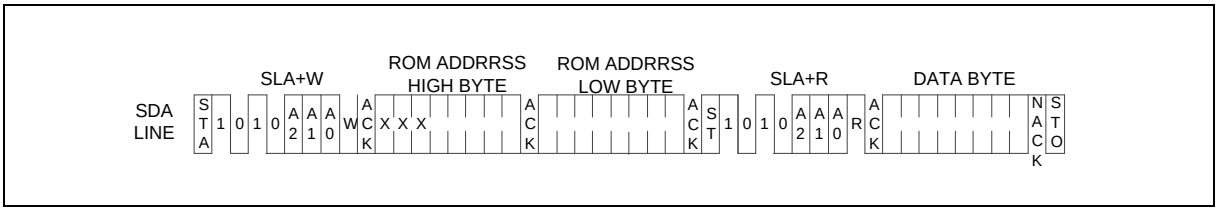


Figure 6.16-28 EEPROM Random Read

Figure 6.16-29 shows how to use the I²C controller to implement the protocol of EEPROM random read.

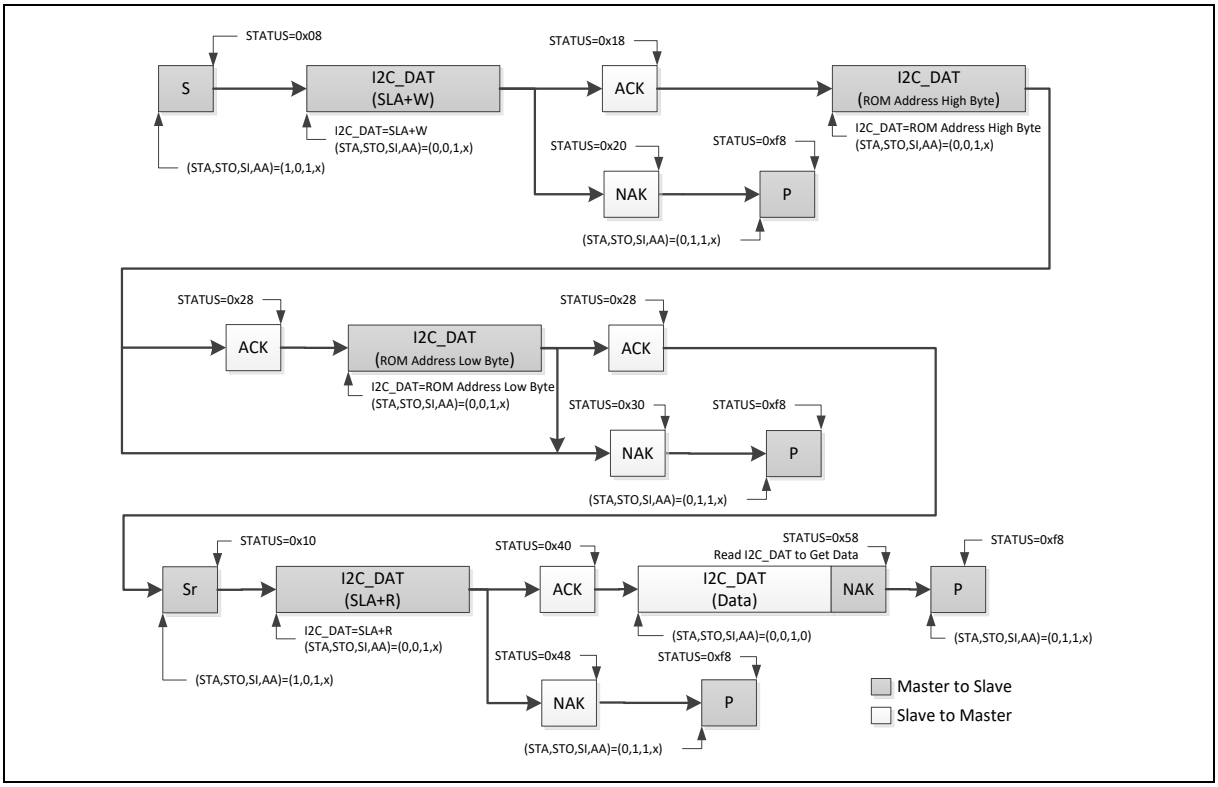


Figure 6.16-29 Protocol of EEPROM Random Read

The I²C 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.

6.16.6 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_CTL0	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_STATUS0	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	0x0002_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

6.16.7 Register Description

I²C Control Register (I2C_CTL0)

Register	Offset	R/W	Description	Reset Value
I2C_CTL0	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.
[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.
[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_STATUS0 register, the SI flag is set by hardware. If bit INTEN (I2C_CTL0 [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.

I²C Data Register (I2C_DAT)

Register	Offset	R/W	Description	Reset Value
I2C_DAT	I2Cn_BA+0x08	R/W	I²C Data Register	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**DAT**	**I²C Data** Bit [7:0] is located with the 8-bit transferred/received data of I²C serial port.

I²C Status Register (I2C_STATUS0)

Register	Offset	R/W	Description	Reset Value
I2C_STATUS0	I2Cn_BA+0x0C	R	I²C Status Register 0	0x0000_00F8

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**STATUS**	**I²C 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_STATUS0 is F8H, no serial interrupt is requested. Others I2C_STATUS0 values correspond to defined I²C states. When each of these states is entered, a status interrupt is requested (SI = 1). A valid status code is present in I2C_STATUS0 one cycle after SI is set by hardware and is still present one cycle after SI has been reset by software. In addition, states 00H stands for a Bus Error. A Bus Error occurs when a START or STOP condition is present at an illegal position in the formation frame. Example of illegal position are during the serial transfer of an address byte, a data byte or an acknowledge bit.

I²C Clock Divided Register (I2C_CLKDIV)

Register	Offset	R/W	Description	Reset Value
I2C_CLKDIV	I2Cn_BA+0x10	R/W	I²C Clock Divided Register	0x0000_0000

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

Bits	Description	
[31:10]	**Reserved**	Reserved.
[15:12]	**NFCNT**	**Noise Filter Counter** The register bits control the input filter width. 0 : filter width 3*PCLK 1 : filter width 4*PCLK *N* : filter width (3+*N*)*PCKL **Note:** Filter width Min :3*PCLK, Max : 18*PCLK
[11:10]	**Reserved**	Reserved.
[9:0]	**DIVIDER**	**I²C Clock Divided** Indicates the I²C clock rate: Data Baud Rate of I²C = (system clock) / (4x (I2C_CLKDIV+1)). **Note:** The minimum value of I2C_CLKDIV is 4.

I²C Time-out Control Register (I2C_TOCTL)

Register	Offset	R/W	Description	Reset Value
I2C_TOCTL	I2Cn_BA+0x14	R/W	I²C Time-out Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					TOCEN	TOCDIV4	TOIF

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**TOCEN**	**Time-out Counter Enable Bit** When enabled, the 14-bit time-out counter will start counting when SI is cleared. 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 extended 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 I²C time-out happened and it can interrupt CPU if I²C 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	I²C Slave Address Register0	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

31	30	29	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.
[10:1]	**ADDR**	**I²C Address** The content of this register is irrelevant when I²C is in Master mode. In the slave mode, the seven most significant bits must be loaded with the chip's own address. The I²C hardware will react if either of the address is matched. **Note:** When software set 10'h000, the address can not 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.
[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 can not use address mask.
[0]	**Reserved**	Reserved.

I²C Wake-up Control Register (I2C_WKCTL)

Register	Offset	R/W	Description	Reset Value
I2C_WKCTL	I2Cn_BA+0x3C	R/W	I²C Wake-up Control Register	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**NHDBUSEN**	**I²C No Hold BUS Enable Bit** 0 = I²C hold bus after wake-up. 1= I²C don't hold bus after wake-up. **Note:** The I²C controller could respond 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 I²C controller and execute the original operation again.
[6:1]	**Reserved**	Reserved.
[0]	**WKEN**	**I²C Wake-up Enable Bit** 0 = I²C wake-up function Disabled. 1= I²C 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	I²C Wake-up Status Register	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**WRSTSWK**	**Read/Write Status Bit in Address Wakeup Frame** 0 = Write command be record on the address match wakeup frame. 1 = Read command be record on the address match wakeup frame. **Note:** This bit will be cleared when software can write 1 to WKAKDONE bit.
[1]	**WKAKDONE**	**Wakeup Address Frame Acknowledge Bit Done** 0 = The ACK bit cycle of address match frame isn't done. 1 = The ACK bit cycle of address match frame is done in power-down. **Note:** This bit can't release WKIF. Software can write 1 to clear this bit.
[0]	**WKIF**	**I²C Wake-up Flag** When chip is woken up from Power-down mode by I²C, 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	I²C 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	PDMASTR
7	6	5	4	3	2	1	0
Reserved					PDMARST	RXPDMAEN	TXPDMAEN

Bits	Description	
[31:8]	**Reserved**	Reserved.
[9]	**ADDR10EN**	**Address 10-bit Function Enable Bit** 0 = Address match 10-bit function Disabled. 1 = Address match 10-bit function Enabled.
[8]	**PDMASTR**	**PDMA Stretch Bit** 0 = I²C send STOP automatically after PDMA transfer done. (only master TX) 1 = I²C SCL bus is stretched by hardware after PDMA transfer done if the SI is not cleared. (only master TX)
[7:3]	**Reserved**	Reserved.
[2]	**PDMARST**	**PDMA Reset** 0 = No effect. 1 = Reset the I²C request to PDMA.
[1]	**RXPDMAEN**	**PDMA Receive Channel Available** 0 = Receive PDMA function Disabled. 1 = Receive PDMA function Enabled.
[0]	**TXPDMAEN**	**PDMA Transmit Channel Available** 0 = Transmit PDMA function Disabled. 1 = Transmit PDMA function Enabled.

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	
[31:8]	**Reserved**	Reserved.
[8]	**ONBUSY**	**On Bus Busy (Read Only)** Indicates that a communication is in progress on the bus. It is set by hardware when a START condition is detected. It is cleared by hardware when a STOP condition is detected. 0 = The bus is IDLE (both SCLK and SDA High). 1 = The bus is busy.
[7:4]	**Reserved**	Reserved.
[3]	**ADMAT3**	**I²C Address 3 Match Status** 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** 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** 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** 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	I²C Timing Configure Control Register	0x0002_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.
[24:16]	**HTCTL**	**Hold Time Configure Control** 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.
[8:0]	**STCTL**	**Setup Time Configure Control** 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	I²C 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.
[13]	**PECDIEN**	**Packet Error Checking Byte Transfer Done Interrupt Enable Bit** 0 = PEC transfer done interrupt Disabled. 1 = PEC transfer done interrupt Enabled. **Note:** This bit is used in PECEN =1.
[12]	**BCDIEN**	**Packet Error Checking Byte Count Done Interrupt Enable Bit** 0 = Byte count done interrupt Disabled. 1 = Byte count done interrupt Enabled. **Note:** This bit is used in PECEN =1.
[11]	**ACKM9SI**	**Acknowledge Manual Enable Extra SI Interrupt** 0 = There is no SI interrupt in the 9th clock cycle when the BUSEN =1 and ACKMEN =1. 1 = There is SI interrupt in the 9th clock cycle when the 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 cleared 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 = PEC calculation is cleared by “Repeat Start” function Disabled. 1 = PEC calculation is cleared by “Repeat Start” function 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 = BUSTOUT is used to calculate the clock low period in bus active. 1 = 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 Disabled. 1 = The system management function Enabled. **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 SUSCON pin in input. 1 = The output enable is active on the SUSCON pin.
[5]	**SCTLOSTS**	**Suspend/Control Data Output Status** 0 = The output of SUSCON pin is low. 1 = The output of SUSCON pin is high.
[4]	**ALERTEN**	**Bus Management Alert Enable Bit** Device Mode (BMHEN =0). 0 = Release the BM_ALERT pin high and Alert Response Header disabled: 0001100x followed by NACK if both of BMDEN and ACKMEN are enabled. 1 = Drive BM_ALERT 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 = BM_ALERT pin not supported. 1 = BM_ALERT 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_STATUS0 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	I²C 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.
[4]	**TORSTEN**	**Time Out Reset Enable Bit** 0 = I²C state machine reset Disabled. 1 = I²C state machine reset Enabled. (The clock and data bus will be released to high)
[3]	**CLKTOIEN**	**Extended Clock Time Out Interrupt Enable Bit** 0 = Clock time out interrupt Disabled. 1 = Clock time out interrupt Enabled.
[2]	**BUSTOIEN**	**Time-out Interrupt Enable Bit** BUSY =1. 0 = SCLK low time-out interrupt Disabled. 1 = SCLK low time-out interrupt Enabled. BUSY =0. 0 = Bus IDLE time-out interrupt Disabled. 1 = Bus IDLE time-out interrupt Enabled.
[1]	**CLKTOEN**	**Cumulative Clock Low Time Out Enable Bit** 0 = Cumulative clock low time-out detection Disabled. 1 = Cumulative clock low time-out detection 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 = Bus clock low time-out detection Disabled. 1 = Bus clock low time-out detection 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	I²C 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	
[31:6]	**Reserved**	Reserved.
[7]	**PECDONE**	**PEC Byte Transmission/Receive Done** 0 = PEC transmission/ receive is not finished when the PECEN is set. 1 = 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 = Cumulative clock low is no any time-out. 1 = Cumulative clock low time-out occurred. **Note:** Software can write 1 to clear this bit.
[5]	**BUSTO**	**Bus Time-out Status** 0 = There is no any time-out or external clock time-out. 1 = 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 SUSCON pin is 0. 1 = The input status of SUSCON pin is 1.
[3]	**ALERT**	**SMBus Alert Status** Device Mode (BMHEN =0). 0 = SMBALERT pin state is low. 1 = SMBALERT pin state is high. Host Mode (BMHEN =1). 0 = No SMBALERT event. 1 = There is SMBALERT event (falling edge) is detected in SMALERT pin when the BMHEN = 1 (SMBus host configuration) and the ALERTEN = 1. **Note:** 1. The SMBALERT 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 = PEC value equal the received PEC data packet. 1 = 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 = Byte count transmission/ receive is not finished when the PECEN is set. 1 = 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 = Bus is IDLE (both SCLK and SDA High). 1 = Bus is busy.

I²C Byte Number Register (I2C_PKTSIZE)

Register	Offset	R/W	Description	Reset Value
I2C_PKTSIZE	I2Cn_BA+0x5C	R/W	I²C 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.
[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	I²C 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.
[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) = X8 + X2 + 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	I²C 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.
[7:0]	BUSTO	**Bus Management Time-out Value** Indicates the bus time-out value in bus is IDLE or SCLK low. **Note:** If the user wants to revise the value of BUSTOUT, the TORSTEN (I2C_BUSTCTL[4]) bit shall be set to 1 and cleared to 0 first when the BUSEN(I2C_BUSCTL[7]) is set.

I²C Clock Low Timer Register (I2C_CLKTOUT)

Register	Offset	R/W	Description	Reset Value
I2C_CLKTOUT	I2Cn_BA+0x68	R/W	I²C 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.
[7:0]	**CLKTO**	**Bus Clock Low Timer** The field is used to configure the cumulative clock extension time-out. **Note:** If the user wants to revise the value of CLKLTOUT, the TORSTEN bit shall be set to 1 and cleared to 0 first when the BUSEN is set.

6.17 Customize IR Receiver (CIR)

6.17.1 Overview

The CIR controller supports to receive data from the output pin of IR receiver. The received data will be stored in registers and wait for further analysis.

6.17.2 Features

- Supports one channel CIR control signal receiver.
- Supports 5 clock sources with 3-bit prescale for vary bit rate sampling.
- Supports a digital filter to be used to filter out noise.
- Supports observeing the noise filter output signal.
- Supports input signal inversion.
- Supports two 32-bit data registers and receive buffer full flag.
- Supports Header, Data0, Data1, End and Special patterns.
- Supports to receive data in Power-down mode and be woken up by specified first 8 bits data pattern. (User needs to switch clock source to LIRC before entering Power-down mode.)
- Offers 11 different kinds of interrupt options and flags.
- Supports receive error detection.
- Supports three receive formats for the CIR control signal waveform.
- Clock PCLK frequency must be faster than CIR internal clock CIR0_CLK frequency.

6.17.3 Block Diagram

The CIR controller block diagram and clock control are shown as follows.

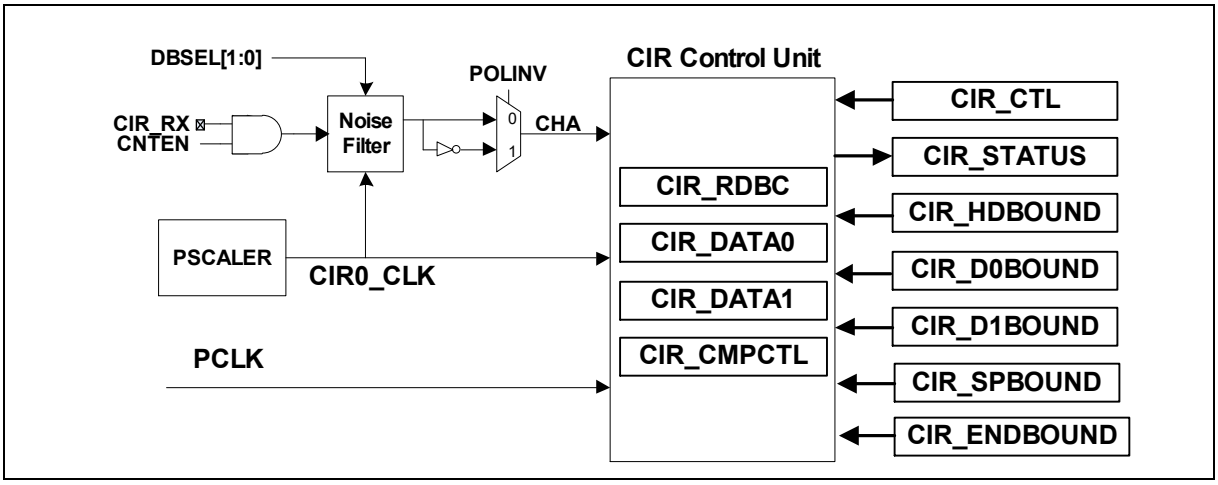


Figure 6.17-1 CIR Controller Block Diagram

The CIR controller input pin, CIR_RX, is connected to the output pin of an IR sensor. Before the external input signal enters CIR control unit, it is processed by a noise filter and a polarity control circuit. Finally, the received data will be stored in CIR_DATA0 and CIR_DATA1 based on setting of the boundary value for each data pattern.

Setting CNTEN (CIR_CTL[0]) will enable the CIR controller. The clock source of CIR0 can be enabled by setting CIR0CKEN (CLK_APBCLK1[15]) and selecting the source of CIR0 peripheral clock on CIR0SEL (CLK_CLKSEL2[26:24]). The block diagram is shown in Figure 6.17-2.

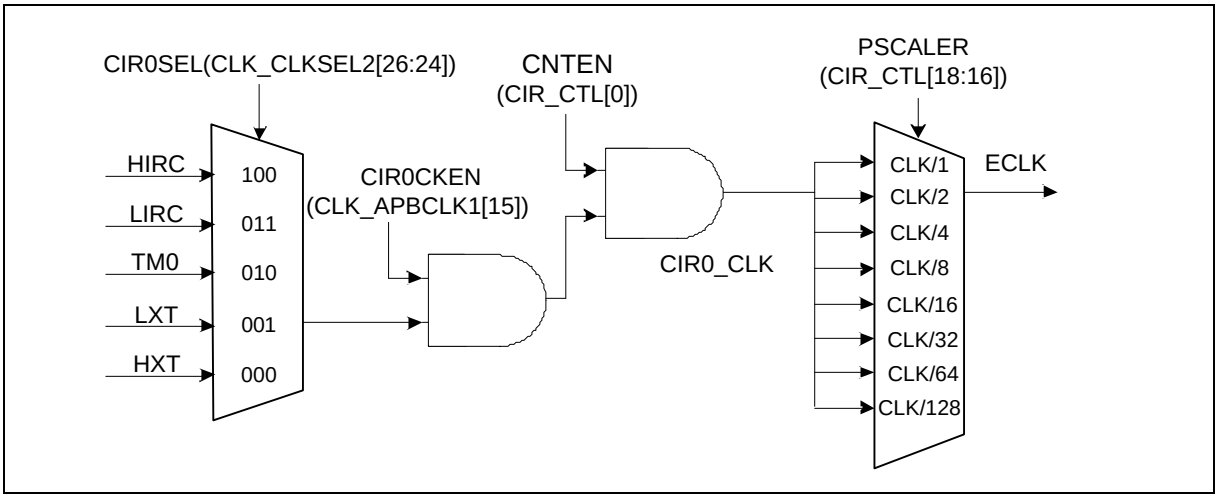


Figure 6.17-2 Clock Source of CIR Controller

6.17.4 Basic Configuration

Setting CNTEN (CIR_CTL[0]) will enable the CIR controller. The clock source of CIR0 in the CIR controller can be enabled by setting CIR0CKEN (CLK_APBCLK1[15]) and switch depending on the setting of the register CIR0SEL (CLK_CLKSEL2[26:24]).

6.17.4.1 CIR0 Basic Configurations

- Clock Source Configuration

- Select the source of CIR0 peripheral clock on CIR0CKEN (CLK_APBCLK1[15]).
- Enable CIR0 peripheral clock in CIR0CKEN (CLK_APBCLK1[15]).

- Reset Configuration
  - Reset CIR0 controller in CIR0RST (SYS_IPRST2[15]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
CIR0	CIR0	PA.2, PC.14, PF.12, PI.4	MFP15, MFP10, MFP15, MFP14

Table 6.17-2 CIR0 Pin Configuration

6.17.5 Functional Description

6.17.5.1 CIR Interrupt Flag

The CIR controller supports 11 types of interrupts including SPMF (CIR_STATUS[0]), D1PMF(CIR_STATUS[1]), D0PMF(CIR_STATUS[2]), HPMF(CIR_STATUS[3]) and EPMF(CIR_STATUS[8]). The interrupt sources can be set while the value of internal timer matches any boundary value from registers CIR_SPBOUND, CIR_D1BOUND, CIR_D0BOUND, CIR_HDBOUND and CIR_ENDBOUND. RBUFF(CIR_STATUS[4]) is set when the data registers DATA0(CIR_DATA0[31 :0]) and DATA1(CIR_DATA0[31 :0]) are full. DRECF(CIR_STATUS[5]) is set when the CIR controller detects CIR_RX positive edge and is cleared when it detects End pattern. If the CIR controller detects error condition, RERRF(CIR_STATUS[6]) is set. The CIR controller provides compared-match function. When the converted data CIR_DATA0[7 :0] matches CMPDAT(CIR_CMPCTL[7 :0]), the flag COMPMF(CIR_STATUS[7]) is set. RBMF(CIR_STATUS[9]) is set when RBITCNT(CIR_RDBC[5 :0]) is equal to RBITCMP(CIR_RDBC[21:16]). All interrupt flags have their corresponding enable bits in CIR_INTCTL[9 :0].

6.17.5.2 CIR Type Mode

The CIR controller provides three type modes: standardized positive edge mode, standardized negative edge mode and flexible positive edge mode.

Standardized Positive Edge Mode

If the CIR controller is configured to standardized positive edge mode (CIR_CTL[6:5] is 00) and CNTEN (CIR_CTL[0]) is set, the internal timer starts to count. Once the input signal CIR_RX detects a positive edge happened, the counter value will be latched and compared with Header(CIR_HDBOUND), DATA0(CIR_D0BOUND), DATA1(CIR_D1BOUND), Special(CIR_SPBOUND) and End(CIR_ENDBOUND). In standardized positive edge mode, the CIR controller needs to receive Header at first and reinitialize RBITCNT(CIR_RDBC[5 :0]).

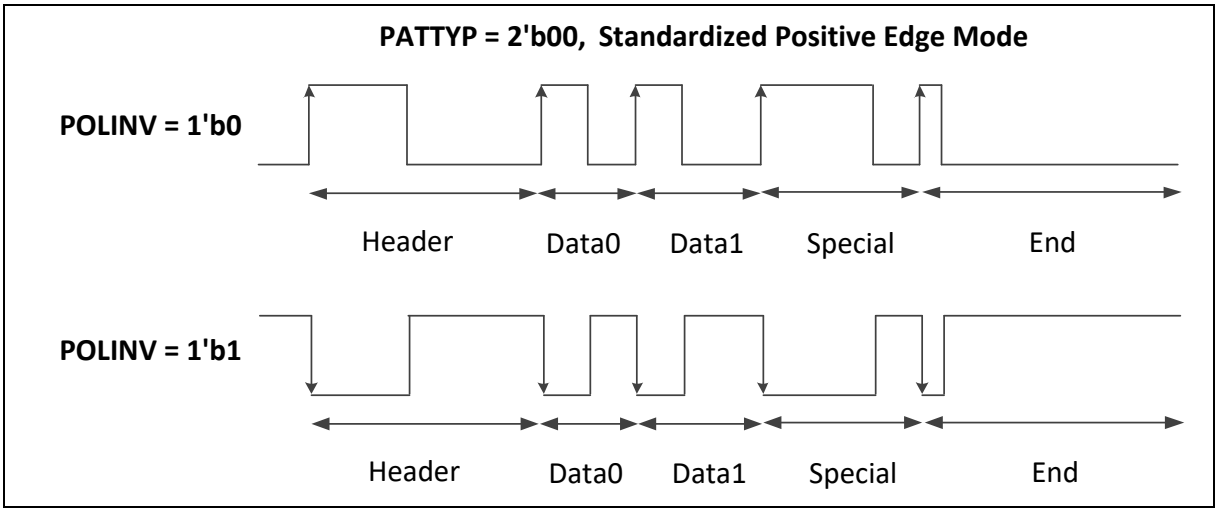


Figure 6.17-3 Standardized Positive Edge Mode in CIR Controller

Standardized Negative Edge Mode

If the CIR controller is configured to standardized negative edge mode (CIR_CTL[6:5] is 01) and CNTEN (CIR_CTL[0]) is set, the internal timer starts to count. Once the input signal CIR_RX detects a negative edge happened, the counter value will be latched and compared with Header(CIR_HDBOUND), DATA0(CIR_D0BOUND), DATA1(CIR_D1BOUND), Special(CIR_SPBOUND) and End(CIR_ENDBOUND). In standardized negative edge mode, the CIR controller needs to receive Header at first and reinitialize RBITCNT(CIR_RDBC[5 :0]).

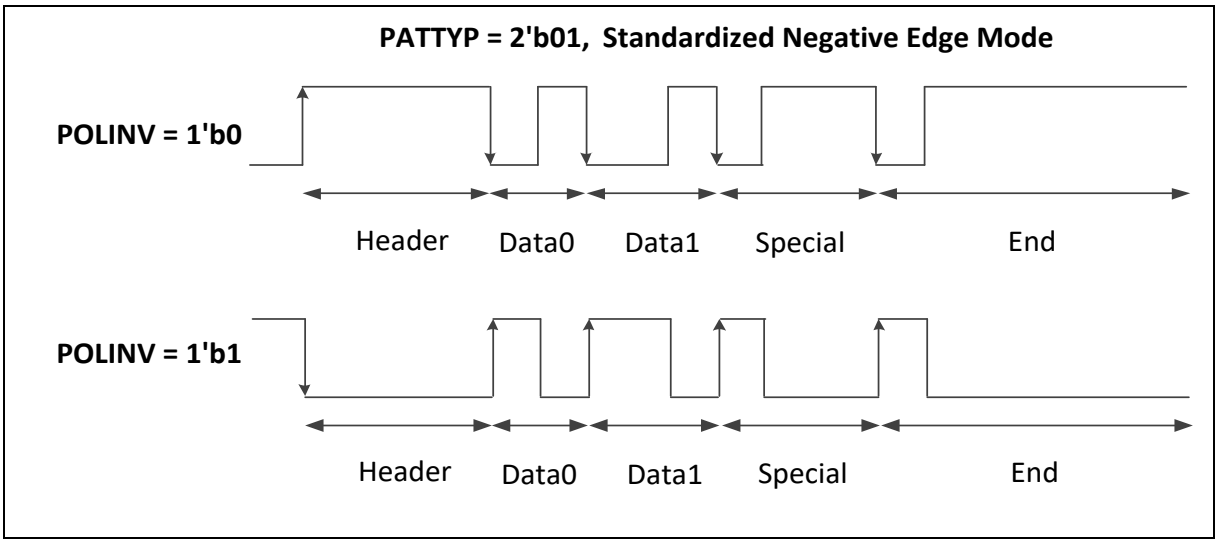


Figure 6.17-4 Standardized Negative Edge Mode in CIR Controller

Flexible Positive Edge Mode

If the CIR controller is configured to flexible positive edge mode (CIR_CTL[6:5] is 10) and CNTEN (CIR_CTL[0]) is set, the internal timer starts to count. Once the input signal CIR_RX detects a positive edge happened, the counter value will be latched and compared with Header(CIR_HDBOUND), DATA0(CIR_D0BOUND), DATA1(CIR_D1BOUND), Special(CIR_SPBOUND) and End(CIR_ENDBOUND). In flexible positive edge mode, the CIR controller is not necessary to receive Header at the beginning, and RBITCNT(CIR_RDBC[5 :0]) will not be reinitialized until Header signal is received in the CIR controller. User can utilize End pattern to clear RBITCNT(CIR_RDBC[5 :0]) in

Power-down mode.

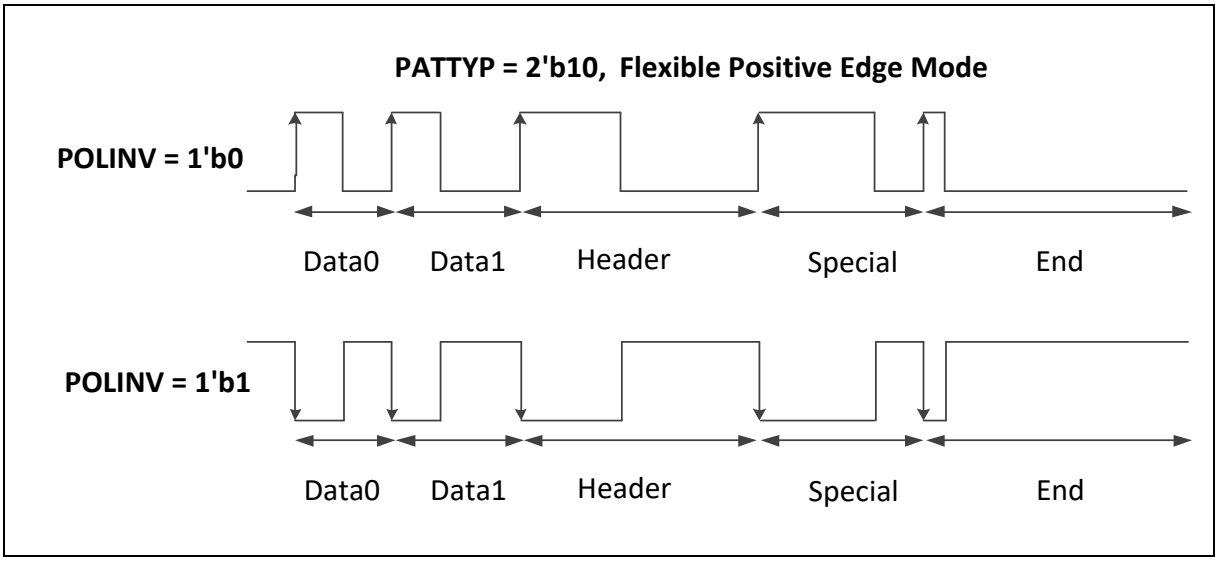


Figure 6.17-5 Flexible Positive Edge Mode in CIR Controller

6.17.5.3 Data Compared-Match Function and Power Down Wake Up Source

The CIR controller provides data compared-match function which is used to identify the data start point from a serial of external input signals. COMPMF (CIR_STATUS[7]) and PDWKF (CIR_STATUS[10]) are used to indicate compare match flag and power down wake up flag, respectively. For power consumption purpose, user can enable both interrupts, CMPMIEN (INTCTL[7]) and PDWKIEN (INTCTL[10]), and specify the first 8-bit data CMPDAT (CMPCTL[7:0]), CMPVALID (CMPCTL[26:24]) and DCMPEN (CMPCTL[8]) for the conditions of wake up system. User needs to clear the interrupt flags COMPMF (CIR_STATUS[7]) and PDWKF (CIR_STATUS[10]) after system wakes up. Once the converted data is matched with the specified data, the compare match flag will be raised and then it will wake up the system. Please make sure to switch CIR source clock to LXT or LIRC since other clocks will be stopped under Power-down mode. Also, remember to enable the NVIC IRQ number of CIR and enable ERRBYP (CIR_CTL[4]) before system enters Power-down mode.

To use the data compared-match function, user needs to write the registers DCMPEN(CIR_CMPCTL[8]), CMPDAT (CIR_CMPCTL[7:0]), CMPVALID (CIR_CMPCTL[26:24]) and clears the register RBITCNT (CIR_RDBC[5:0]) before setting the register CNTEN (CIR_CTL[0]) to 1. After the CIR controller starts to convert the data, COMPMF will be set when the input signals are equal to CMPDAT (CIR_CMPCTL[7:0]) and then the CIR controller will stop monitoring DATA0[7:0] (CIR_DATA0[7 :0). User needs to set the register CMPMSK(CIR_CMPCTL[16]) to re-start the data compared-match function.

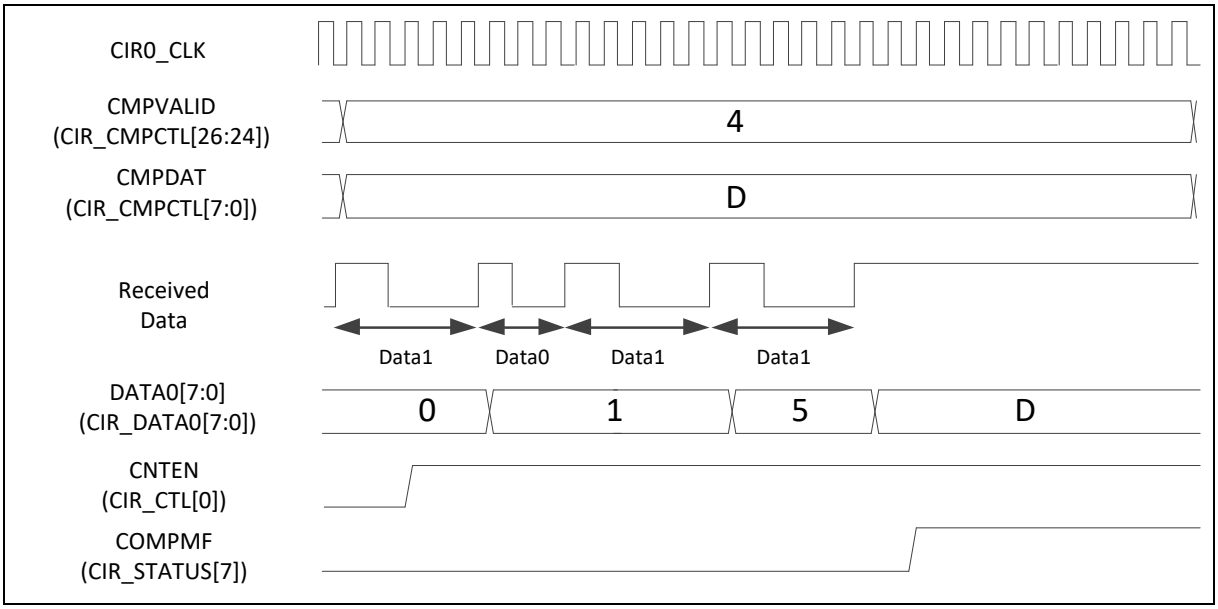


Figure 6.17-6 Compared-Match Function in CIR Controller

6.17.5.4 Bit Count Compared-Match Function

The Bit count compared-match function is suitable for the situation when the user already knows the length of the data. The register RBITCNT (CIR_RDBC[5:0]) is used to record the number of bits in the converted data. User needs to write the register RBITCMP(CIR_RDBC[21:16]) before setting the register CNTEN (CIR_CTL[0]) to 1. After the CIR controller starts to convert the data, RBMF will be set when RBITCNT value is equal to RBITCMP, and if the user has set RBMIEN (CIR_INTCTL[9]) to 1 at the same time, an interrupt will be asserted.

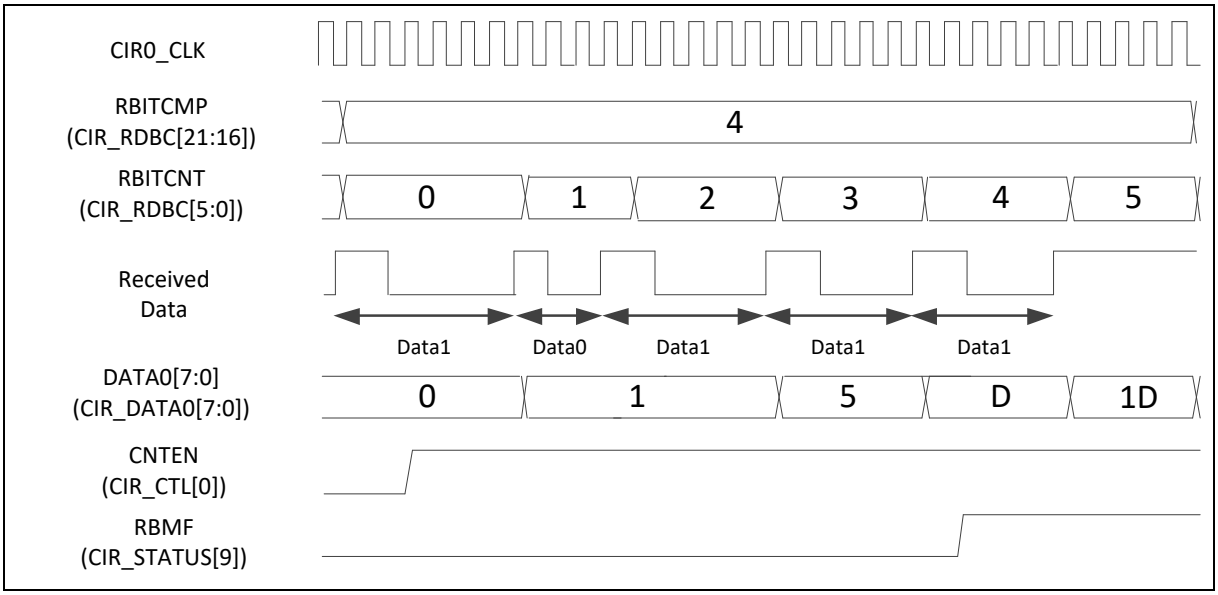


Figure 6.17-7 Bit Count Compared-Match Function in CIR Controller

6.17.5.5 Error Flag

The CIR controller monitors some operating conditions and uses error flag RERRF (CIR_STATUS[6]) to notify users that the operating conditions have been violated.

1. The converted data did not meet the high and low boundary value of Header, Data0, Data1 or

Special pattern and the flag RERRF (CIR_STATUS[6]) will be set.

2. In standardized positive edge mode (CIR_CTL[6:5] is 00) and standardized negative edge mode (CIR_CTL[6:5] is 01), the CIR should receive Header pattern prior to other patterns. If the CIR controller receives Data0, Data1, Special without first receiving Header, the flag RERRF (CIR_STATUS[6]) will be set.
3. When the input signal is not changed for a while, it will trigger the CIR controller timeout mechanism and set EPMF (CIR_STATUS[8]). Then the received data flag DRECF (CIR_STATUS[5]) will be cleaned by EPMF (CIR_STATUS[8]). However, when an input signal comes in at the same time, the received data flag DRECF (CIR_STATUS[5]) will not be decided. Therefore, the error flag RERRF (CIR_STATUS[6]) will be set.
4. The converted data will not be stored in CIR_DATAx registers by default if an error flag is asserted. The CIR controller enables user to set ERRBYP (CIR_CTL[4]) to decide whether to continue converting data after RERRF (CIR_STATUS[6]) is set. An interrupt will be asserted if the PERRIEN (CIR_INTCTL[6]) is set and RERRF (CIR_STATUS[6]) is raisied.

6.17.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
CIR0 Base Address: **CIR0_BA = 0x4005_F000**				
CIR_CTL	CIR0_BA+0x00	R/W	CIR Control Register	0x0000_0000
CIR_CMPCTL	CIR0_BA+0x04	R/W	CIR Data Compare Control Register	0x0000_0000
CIR_STATUS	CIR0_BA+0x08	R/W	CIR Status Register	0x0000_0000
CIR_INTCTL	CIR0_BA+0x10	R/W	CIR Interrupt Control Register	0x0000_0000
CIR_HDBOUND	CIR0_BA+0x18	R/W	CIR Header Pattern Boundry Register	0x0000_0000
CIR_D0BOUND	CIR0_BA+0x1C	R/W	CIR Data 0 Pattern Boundry Register	0x0000_0000
CIR_D1BOUND	CIR0_BA+0x20	R/W	CIR Data 1 Pattern Boundry Register	0x0000_0000
CIR_SPBOUND	CIR0_BA+0x24	R/W	CIR Special Pattern Boundry Register	0x0000_0000
CIR_ENDBOUND	CIR0_BA+0x28	R/W	CIR End Pattern Boundry Register	0x0000_0000
CIR_LTVR	CIR0_BA+0x38	R	CIR Latch Timer Value Register	0x0000_0000
CIR_RDBC	CIR0_BA+0x3C	R/W	CIR Receive Data Bit Count Register	0x0000_0000
CIR_DATA0	CIR0_BA+0x40	R/W	CIR Receive Data0 Register	0x0000_0000
CIR_DATA1	CIR0_BA+0x44	R/W	CIR Receive Data1 Register	0x0000_0000

6.17.7 Register Description

CIR Control Register (CIR_CTL)

Register	Offset	R/W	Description	Reset Value
CIR_CTL	CIR0_BA+0x00	R/W	CIR Control Register	0x0000_0000

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

Bits	Description	
[31:19]	**Reserved**	Reserved.
[18:16]	**PSCALER**	**Sampling Clock Prescaler** 000 = No prescaler. 001 = Prescaler is 2 clocks 010 = Prescaler is 4 clocks. 011 = Prescaler is 8 clocks. 100 = Prescaler is 16 clocks. 101 = Prescaler is 32 clocks. 110 = Prescaler is 64 clocks. 111 = Prescaler is 128 clocks. **Note:** The sampling clock should be less than PCLK1.
[15:12]	**Reserved**	Reserved.
[11]	**FOSTRS**	**Filter Output Signal Stored in Register Selection** 0 = Filter output signal stored in CIR_STATUS[16] Disabled. 1 = Filter output signal stored in CIR_STATUS[16] Enabled.
[10:9]	**DBSEL**	**Debounce Sampling Selection** 00 = CIR noise filter Disabled. 01 = CIR input debounce count Enabled with two sample matched. 10 = CIR input debounce count Enabled with three sample matched. 11 = CIR input debounce count Enabled with four sample matched.
[8:7]	**Reserved**	Reserved.
[6:5]	**PATTYP**	**CIR Pattern Format Selection** 00 = Standardized positive edge mode 01 = Standardized negative edge mode.

		10 = Flexible positive edge mode. 11 = Reserved.
[4]	**ERRBYP**	**Error Pattern Bypass** 0 = Data will be dropped if RERRF(CIR_STATUS[6]) is 1. 1 = Data will keep to save in DATAx if RERRF(CIR_STATUS[6]) flag is 1. **Note:** 1.If user clears RERRF(CIR_STATUS[6]), then CIR will keep to convert data and store in CIR_DATAx. 2.User must set ERRBYP (CIR_CTL[4]) to 1 before entering Power-down mode.
[3:2]	**Reserved**	Reserved.
[1]	**POLINV**	**CIR Input Polarity Inverse** 0 = CIR input polarity is normal. 1 = CIR input polarity is inversed.
[0]	**CNTEN**	**CIR Counter Enable** 0 = CIR counter Disabled. 1 = CIR counter Enabled. **Note:** When user changes CNTEN (CIR_CTL[0]) from 0 to 1, system will generate a signal to initialize all interrupt flags, RBITCNT (CIR_RDBC[5:0])and ITVR (CIR_ITVR[31:0]).

CIR Data Compare Control Register (CIR_CMPCTL)

Register	Offset	R/W	Description	Reset Value
CIR_CMPCTL	CIR0_BA+0x04	R/W	CIR Data Compare Control Register	0x0000_0000

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

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:24]	**CMPVALID**	**Data Compared Valid Bit Selection** 000 = Compare bit 0. 001 = Compare bit 0 to bit 1. 010 = Compare bit 0 to bit 2. 011 = Compare bit 0 to bit 3. 100 = Compare bit 0 to bit 4. 101 = Compare bit 0 to bit 5. 110 = Compare bit 0 to bit 6. 111 = Compare bit 0 to bit 7. **Note:** The sampling clock should be less than PCLK1.
[23:17]	**Reserved**	Reserved.
[16]	**CMPMSK**	**Data Compared Mask Initialization** 0 = No effect. 1 = Re-initialize the data compared match function to monitor DATA0[N:0] (CIR_DATA0[N :0). **Note:** This bit is auto cleared by hardware.
[15:9]	**Reserved**	Reserved.
[8]	**DCMPEN**	**Data Compared Match Function Selection** 0 = Data compared match function Disabled. 1 = Data compared match function Enabled.
[7:0]	**CMPDAT**	**Compared Match Data** This bit field should be filled with the expected data. It will be compared with CIR_DATA0[N:0]. **Note:** N is determined by CMPVALID(CIR_CMPCTL[26 :24]).

CIR Status Register (CIR_STATUS)

Register	Offset	R/W	Description	Reset Value
CIR_STATUS	CIR0_BA+0x08	R/W	CIR Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						RBITCBS	NFOS
15	14	13	12	11	10	9	8
Reserved					PDWKF	RBMF	EPMF
7	6	5	4	3	2	1	0
COMPMF	RERRF	DRECF	RBUFF	HPMF	D0PMF	D1PMF	SPMF

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**RBITCBS**	**RBITCNT Busy Clearing Status** 0 = RBITCNT has completed the clearing process when user writes 1 to RBITCNT(CIR_RDBC[5:0]) or user changes CNTEN (CIR_CTL[0]) from 0 to 1, and system will generate a signal to initialize RBITCNT (CIR_RDBC[5:0]). 1 = RBITCNT undergoes clearing process when user writes 1 to RBITCNT(CIR_RDBC[5:0]) or user changes CNTEN (CIR_CTL[0]) from 0 to 1, and system will generate a signal to initialize RBITCNT (CIR_RDBC[5:0]).
[16]	**NFOS**	**Noise Filter Output Signal Status** 0 = Noise filter output value is 0. 1 = Noise filter output value is 1.
[15:11]	**Reserved**	Reserved.
[10]	**PDWKF**	**Power Down Wake Up Flag** 1 = Power down wake up happened. 0 = Power down wake up never happened. **Note:** This bit is only cleared by writing 1 to it.
[9]	**RBMF**	**Receive Bit Match Flag** 1 = Receive bit match happened. 0 = Receive bit match never happened. **Note:** This bit is only cleared by writing 1 to it.
[8]	**EPMF**	**End Pattern Match Flag** 1 = End pattern match happened. 0 = End pattern match never happened. **Note:** This bit is only cleared by writing 1 to it.
[7]	**COMPMF**	**Compare Match Flag** 1 = Compare match happened.

		0 = Compare match never happened. **Note:** This bit is only cleared by writing 1 to it.
[6]	**RERRF**	**Receive Error Flag** 1 = Receive error happened. 0 = Receive error never happened. **Note:** This bit is only cleared by writing 1 to it.
[5]	**DRECF**	**Data Receive Flag** 1 = CIR has started to convert data. 0 = CIR has not started to convert data. **Note:** This bit is only cleared by writing 1 to it.
[4]	**RBUFF**	**Receiving Buffer Full Flag** 1 = Receiving buffer full happened. 0 = Receiving buffer full never happened. **Note:** This bit is only cleared by writing 1 to it.
[3]	**HPMF**	**Header Pattern Match Flag** 1 = Header pattern happened. 0 = Header pattern never happened. **Note:** This bit is only cleared by writing 1 to it.
[2]	**D0PMF**	**Data0 Pattern Match Flag** 1 = Data0 pattern happened. 0 = Data0 pattern never happened. **Note:** This bit is only cleared by writing 1 to it.
[1]	**D1PMF**	**Data1 Pattern Match Flag** 1 = Data1 pattern happened. 0 = Data1 pattern never happened. **Note:** This bit is only cleared by writing 1 to it.
[0]	**SPMF**	**Special Pattern Match Flag** 1 = Special pattern happened. 0 = Special pattern never happened. **Note:** This bit is only cleared by writing 1 to it.

CIR Interrupt Control Register (CIR_INTCTL)

Register	Offset	R/W	Description	Reset Value
CIR_INTCTL	CIR0_BA+0x10	R/W	CIR Interrupt 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					PDWKIEN	RBMIEN	EPMIEN
7	6	5	4	3	2	1	0
CMPMIEN	PERRIEN	DRECIEN	RBUFIEN	HPMIEN	D0PMIEN	D1PMIEN	SPMIEN

Bits	Description	
[31:10]	**Reserved**	Reserved.
[10]	**PDWKIEN**	**Power Down Wake-up interrupt Enable Bit** 0 = Power down wake-up interrupt Disabled. 1 = Power down wake-up interrupt Enabled.
[9]	**RBMIEN**	**Receive Bit Match Interrupt Enable Bit** 0 = Receive bit match interrupt Disabled. 1 = Receive bit match interrupt Enabled.
[8]	**EPMIEN**	**End Pattern Match Interrupt Enable Bit** 0 = End pattern match interrupt Disabled. 1 = End pattern match interrupt Enabled.
[7]	**CMPMIEN**	**Compare Match Interrupt Enable Bit** 0 = Compare match interrupt Disabled. 1 = Compare match interrupt Enabled.
[6]	**PERRIEN**	**Pattern Error Interrupt Enable Bit** 0 = Pattern error interrupt Disabled. 1 = Pattern error interrupt Enabled.
[5]	**DRECIEN**	**Data Receive Interrupt Enable Bit** 0 = Data receive interrupt Disabled. 1 = Data receive interrupt Enabled.
[4]	**RBUFIEN**	**Receive Buffer Full Interrupt Enable Bit** 0 = Receive buffer full interrupt Disabled. 1 = Receive buffer full interrupt Enabled.
[3]	**HPMIEN**	**Header Pattern Match Interrupt Enable Bit** 0 = Header pattern match interrupt Disabled.

		1 = Header pattern match interrupt Enabled.
[2]	**D0PMIEN**	**Data0 Pattern Match Interrupt Enable Bit** 0 = Data0 pattern match interrupt Disabled. 1 = Data0 pattern match interrupt Enabled.
[1]	**D1PMIEN**	**Data1 Pattern Match Interrupt Enable Bit** 0 = Data1 pattern match interrupt Disabled. 1 = Data1 pattern match interrupt Enabled.
[0]	**SPMIEN**	**Special Pattern Match Interrupt Enable Bit** 0 = Special pattern match interrupt Disabled. 1 = Special pattern match interrupt Enabled.

CIR Header Bound Register (CIR_HDBOUND)

Register	Offset	R/W	Description	Reset Value
CIR_HDBOUND	CIR0_BA+0x18	R/W	CIR Header Pattern Boundry Register	0x0000_0000

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

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:16]	**HBOUND**	**High Boundary Header Pattern** Upper limit of Header pattern input range. **Note:** If HBOUND and LBOUND are equal to 0, the CIR controller will not monitor the Header pattern boundary.
[15:11]	**Reserved**	Reserved.
[10:0]	**LBOUND**	**Low Boundary Header Pattern** Lower limit of Header pattern input range. **Note:** If HBOUND and LBOUND are equal to 0, the CIR controller will not monitor the Header pattern boundary.

CIR Data 0 Bound Register (CIR_D0BOUND)

Register	Offset	R/W	Description	Reset Value
CIR_D0BOUND	CIR0_BA+0x1C	R/W	CIR Data 0 Pattern Boundry Register	0x0000_0000

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

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:16]	**HBOUND**	**High Boundary Data0 Pattern** Upper limit of Data 0 pattern input range. **Note:** If HBOUND and LBOUND are equal to 0, the CIR controller will not monitor the Data0 pattern boundary.
[15:11]	**Reserved**	Reserved.
[10:0]	**LBOUND**	**Low Boundary Data0 Pattern** Lower limit of Data 0 pattern input range. **Note:** If HBOUND and LBOUND are equal to 0, the CIR controller will not monitor the Data0 pattern boundary.

CIR Data 1 Bound Register (CIR_D1BOUND)

Register	Offset	R/W	Description	Reset Value
CIR_D1BOUND	CIR0_BA+0x20	R/W	CIR Data 1 Pattern Boundry Register	0x0000_0000

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

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:16]	**HBOUND**	**High Boundary Data 1 Pattern** Upper limit of Data 1 pattern input range. **Note:** If HBOUND and LBOUND are equal to 0, the CIR controller will not monitor the Data1 pattern boundary.
[15:11]	**Reserved**	Reserved.
[10:0]	**LBOUND**	**Low Boundary Data 1 Pattern** Upper limit of Data 1 pattern input range. **Note:** If HBOUND and LBOUND are equal to 0, the CIR controller will not monitor the Data1 pattern boundary.

CIR Special Pattern Bound Register (CIR_SPBOUND)

Register	Offset	R/W	Description	Reset Value
CIR_SPBOUND	CIR0_BA+0x24	R/W	CIR Special Pattern Boundry Register	0x0000_0000

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

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:16]	**HBOUND**	**High Boundary Special Pattern** Upper limit of Special pattern input range. **Note:** If HBOUND and LBOUND are equal to 0, the CIR controller will not monitor the Special pattern boundary.
[15:11]	**Reserved**	Reserved.
[10: 0]	**LBOUND**	**Low Boundary Special Pattern** Lower limit of Special pattern input range. **Note:** If HBOUND and LBOUND are equal to 0, the CIR controller will not monitor the Special pattern boundary.

CIR End Point Bound Register (CIR_ENDBOUND)

Register	Offset	R/W	Description	Reset Value
CIR_ENDBOUND	CIR0_BA+0x28	R/W	CIR End Pattern Boundry Register	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**Reserved**	Reserved.
[15:11]	**Reserved**	Reserved.
[10:0]	**LBOUND**	**Low Boundary End Pattern** Lower limit of End pattern input range.

CIR Latch Timer Value Register (CIR_LTLVR)

Register	Offset	R/W	Description	Reset Value
CIR_LTVR	CIR0_BA+0x38	R	CIR Latch Timer 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					LTV		
7	6	5	4	3	2	1	0
LTV							

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10:0]	**LTV**	**Latch Timer Value** The register is used to record CIR latch timer value. **Note:** User can only read this register when HPMF (CIR_STATUS[3]), D0PMF (CIR_STATUS[2]), D1PMF (CIR_STATUS[1]), SPMF (CIR_STATUS[0]) or RERRF (CIR_STATUS[6]) occurred.

CIR Receive Data Bit Count Register (CIR_RDBC)

Register	Offset	R/W	Description	Reset Value
CIR_RDBC	CIR0_BA+0x3C	R/W	CIR Receive Data Bit Count Register	0x0000_0000

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

Bits	Description	
[31:23]	**Reserved**	Reserved.
[22:16]	**RBITCMP**	**Receive Data Bit Compare Data** User can limit the converted data length by RBITCMP register. When CIR starts to convert data and RBITCNT(CIR_RDBC[5:0]) is equal to RBITCMP (CIR_RDBC[21:16]), the flag RBMF(CIR_STATUS[9]) will be asserted **Note:** The maximum value of RBITCMP (CIR_RDBC[22:16]) is 7'h40.
[15:9]	**Reserved**	Reserved.
[8]	**BCCMEN**	**Bit Count Compared Match Selection** 0 = Bit count compared match function Disabled. 1 = Bit count compared match function Enabled.
[7]	**Reserved**	Reserved.
[6:0]	**RBITCNT**	**Receive Data Bit Counts** RBITCNT (CIR_RDBC[5 :0]) correspond to CIR_DATA0 and CIR_DATA1 when CIR starts to convert data. **Note:** 1. User can write 1 to CIR_RDBC[5:0] to clean RBITCNT (CIR_RDBC[5 :0]) value. 2. RBITCNT (CIR_RDBC[5 :0]) value indicates the amount of data DATA0 (CIR_DATA0) that has already been confirmed. 3. The maximum value of RBITCNT (CIR_RDBC[5 :0]) is 7'h40.

CIR Receive Data0 Register (CIR_DATA0)

Register	Offset	R/W	Description	Reset Value
CIR_DATA0	CIR0_BA+0x40	R/W	CIR Receive Data0 Register	0x0000_0000

31	30	29	28	27	26	25	24
DATA0							
23	22	21	20	19	18	17	16
DATA0							
15	14	13	12	11	10	9	8
DATA0							
7	6	5	4	3	2	1	0
DATA0							

Bits	Description	
[31:0]	DATA0	**CIR DATA0 Register** CIR converts data and stores the data in Data0 when RBITCNT(CIR_RDBC[5 :0]) value is between 0 to 31. **Note:** User can write 1 to CIR_DATA0[31:0] to clean DATA0 value only when the register CNTEN(CIR_CTL[0]) is set to 0.

CIR Receive Data1 Register (CIR_DATA1)

Register	Offset	R/W	Description	Reset Value
CIR_DATA1	CIR0_BA+0x44	R/W	CIR Receive Data1 Register	0x0000_0000

31	30	29	28	27	26	25	24
DATA1							
23	22	21	20	19	18	17	16
DATA1							
15	14	13	12	11	10	9	8
DATA1							
7	6	5	4	3	2	1	0
DATA1							

Bits	Description	
[31:0]	DATA1	**CIR DATA1 Register** CIR converts data and stores the data in Data1 when RBITCNT(CIR_RDBC[5 :0]) value is between 32 to 63. **Note:** User can write 1 to CIR_DATA1[31:0] to clean DATA1 value only when the register CNTEN(CIR_CTL[0]) is set to 0.

6.18 CRC Controller (CRC)

6.18.1 Overview

The Cyclic Redundancy Check (CRC) generator can perform CRC calculation with four common polynomials CRC-CCITT, CRC-8, CRC-16, and CRC-32 settings.

6.18.2 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

6.18.3 Block Diagram

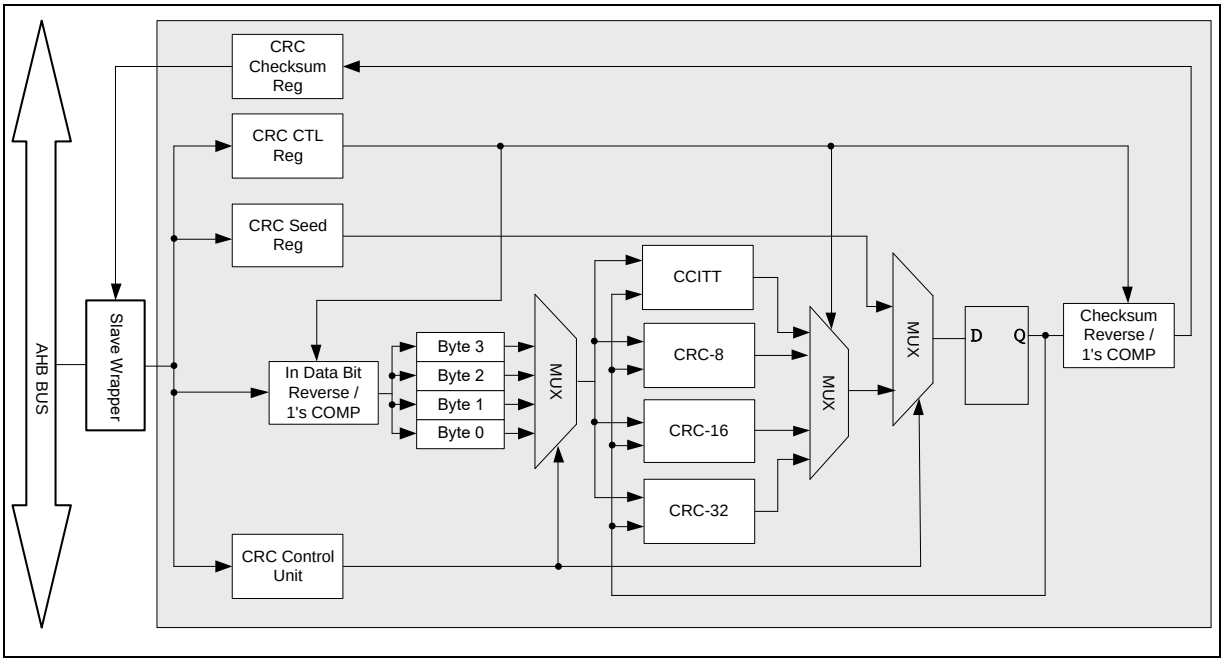


Figure 6.18-1 CRC Generator Block Diagram

6.18.4 Basic Configuration

- Clock Source Configuration
  - Enable CRC peripheral clock in CRCCKEN (CLK_AHBCLK[7]).
- Reset Configuration
  - Reset CRC controller in CRCRST (SYS_IPRST0[7]).

6.18.5 Functional Description

CRC generator can perform CRC calculation with four common 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).
   1) Initial setting for CRC calculation.
2. Configure 1's complement for CRC checksum by setting CHKSFMT (CRC_CTL[27] Checksum 1's Complement).
   1) 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 6.18-2.
   2) Configure 1's complement for CRC write data by setting DATFMT (CRC_CTL[26] Write Data 1's Complement).
   3) 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 6.18-3.
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.

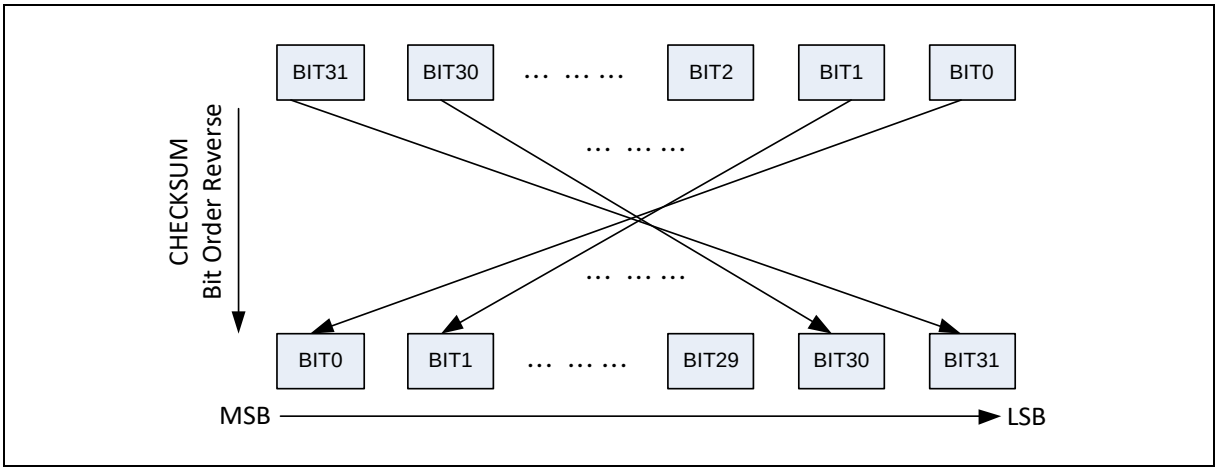


Figure 6.18-2 CHECKSUM Bit Order Reverse Functional Block

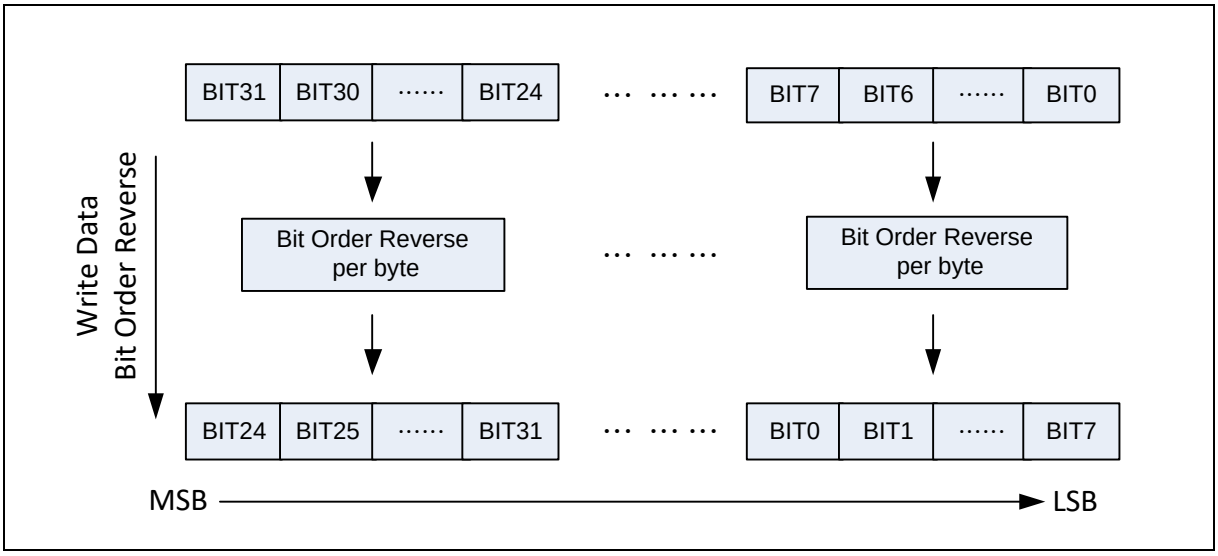


Figure 6.18-3 Write Data Bit Order Reverse Functional Block

6.18.6 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

6.18.7 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 checksum result 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 per byte 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 in is 0x55DD33BB.
[23:2]	**Reserved**	Reserved.
[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.

6.19 Pseudo Random Number Generator (PRNG)

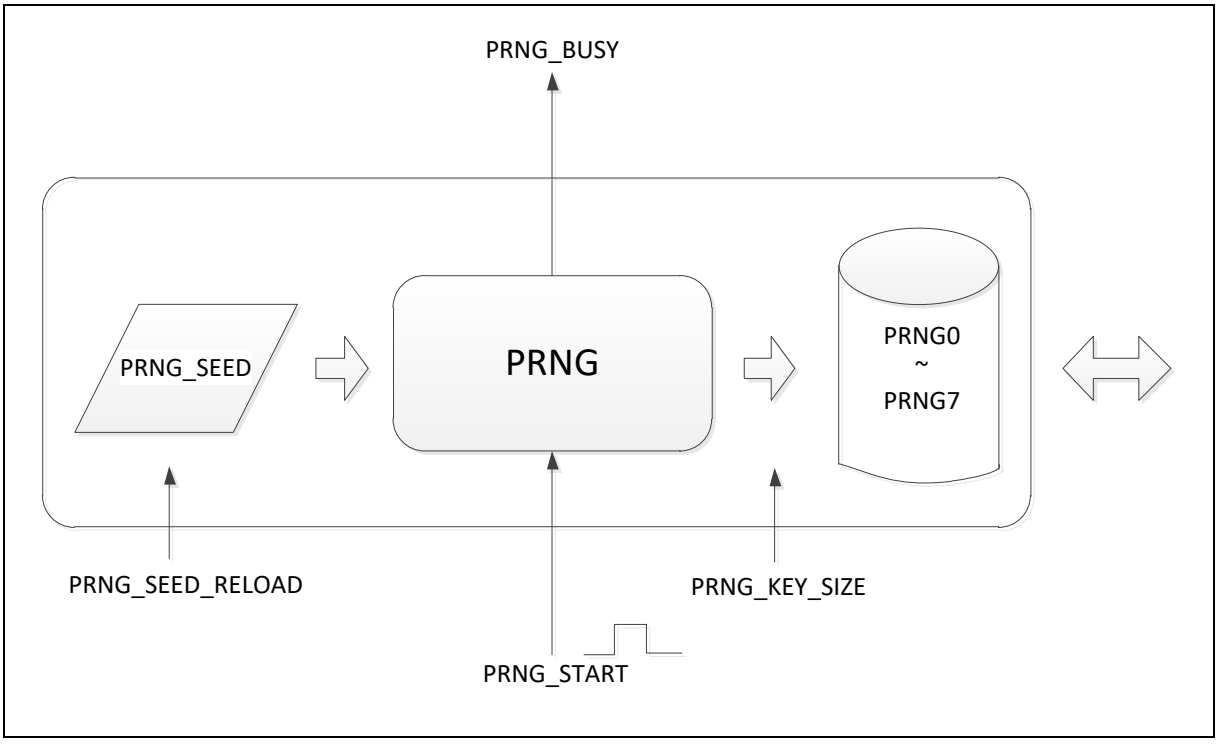
6.19.1 Overview

The PRNG core supports 64, 128, 192 and 256 bits random number generation.

6.19.2 Features

- PRNG
  - Supports 64, 128, 192 and 256 bits random number generation

6.19.3 Block Diagram



6.19.4 Basic Configuration

- Clock Source Configuration
  - Enable PRNG peripheral clock in (CLK_APBCLK1[24])
- Reset Configuration
  - Reset PRNG controller in (SYS_IPRST2[24])

6.19.5 Functional Description

Software can control the data flow by enabling the PRNG_INTEN, and monitor the accelerator status by checking the PRNG_INTSTS status register.

Programming steps to get the pseudo random number are depicted below.

1. Check the BUSY(PRNG_CTL[8]) until it comes to 0.
2. Write a random seed to PRNG_SEED. Note that PRNG_SEED should be initialized since it is not initialized as the chip powers up.

3. Configure PRNG control register PRNG_CTL for key size(PRNG_CTL[3:2]), seed reload(PRNG_CTL[1]), and PRNG start(PRNG_CTL[0]).
4. Software checks BUSY(PRNG_CTL[8]) until it comes to 0, or waits for the PRNGIF(INTSTS[16]) (must enable PRNGIEN(INTEN[16])). Then software can read the output random numbers from PRNG_KEY0 ~ PRNG_KEY7.

6.19.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
PRNG Base Address: **PRNG_BA = 0x400B_A000**				
PRNG_INTEN	PRNG_BA+0x000	R/W	PRNG Interrupt Enable Control Register	0x0000_0000
PRNG_INTSTS	PRNG_BA+0x004	R/W	PRNG Interrupt Flag	0x0000_0000
PRNG_CTL	PRNG_BA+0x008	R/W	PRNG Control Register	0x0000_0000
PRNG_SEED	PRNG_BA+0x00C	W	Seed for PRNG	0xFFFF_FFFF
PRNG_KEY0	PRNG_BA+0x010	R	PRNG Generated Key0	0xFFFF_FFFF
PRNG_KEY1	PRNG_BA+0x014	R	PRNG Generated Key1	0xFFFF_FFFF
PRNG_KEY2	PRNG_BA+0x018	R	PRNG Generated Key2	0xFFFF_FFFF
PRNG_KEY3	PRNG_BA+0x01C	R	PRNG Generated Key3	0xFFFF_FFFF
PRNG_KEY4	PRNG_BA+0x020	R	PRNG Generated Key4	0xFFFF_FFFF
PRNG_KEY5	PRNG_BA+0x024	R	PRNG Generated Key5	0xFFFF_FFFF
PRNG_KEY6	PRNG_BA+0x028	R	PRNG Generated Key6	0xFFFF_FFFF
PRNG_KEY7	PRNG_BA+0x02C	R	PRNG Generated Key7	0xFFFF_FFFF

6.19.7 Register Description

6.19.7.1 PRNG Register

PRNG Interrupt Enable Control Register (PRNG_INTEN)

Register	Offset	R/W	Description	Reset Value
PRNG_INTEN	PRNG_BA+0x000	R/W	PRNG Interrupt Enable Control Register	0x0000_0000

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

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**PRNGIEN**	**PRNG Interrupt Enable Bit** 0 = PRNG interrupt Disabled. 1 = PRNG interrupt Enabled.
[15:0]	**Reserved**	Reserved.

PRNG Interrupt Flag Register (PRNG_INTSTS)

Register	Offset	R/W	Description	Reset Value
PRNG_INTSTS	PRNG_BA+0x004	R/W	PRNG Interrupt Flag	0x0000_0000

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

Bits	Description	
[31:18]	**Reserved**	Reserved.
[16]	**PRNGIF**	**PRNG Finish Interrupt Flag** 0 = No PRNG interrupt. 1 = PRNG key generation done interrupt. **Note:** This bit is cleared by writing 1, and it has no effect by writing 0.
[15:0]	**Reserved**	Reserved.

PRNG Control Register (PRNG_CTL)

Register	Offset	R/W	Description	Reset Value
PRNG_CTL	PRNG_BA+0x008	R/W	PRNG 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							BUSY
7	6	5	4	3	2	1	0
Reserved				KEYSZ		SEEDRLD	START

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**BUSY**	**PRNG Busy (Read Only)** 0 = PRNG engine is idle. 1 = PRNG engine is generating PRNG_KEYx.
[7:4]	**Reserved**	Reserved.
[3:2]	**KEYSZ**	**PRNG Generate Key Size** 00 = 64 bits. 01 = 128 bits. 10 = 192 bits. 11 = 256 bits.
[1]	**SEEDRLD**	**Reload New Seed for PRNG Engine** 0 = Generating key based on the current seed. 1 = Reload new seed.
[0]	**START**	**Start PRNG Engine** 0 = Stop PRNG engine. 1 = Generate a new key and store the new key to the register PRNG_KEYx, which will be cleared when the new key is generated.

PRNG Seed Register (PRNG_SEED)

Register	Offset	R/W	Description	Reset Value
PRNG_SEED	PRNG_BA+0x00C	W	Seed for PRNG	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	**Seed for PRNG (Write Only)** The bits store the seed for PRNG engine.

PRNG Key x Register (PRNG_KEYx)

Register	Offset	R/W	Description	Reset Value
PRNG_KEY0	PRNG_BA+0x010	R	PRNG Generated Key0	0xFFFF_FFFF
PRNG_KEY1	PRNG_BA+0x014	R	PRNG Generated Key1	0xFFFF_FFFF
PRNG_KEY2	PRNG_BA+0x018	R	PRNG Generated Key2	0xFFFF_FFFF
PRNG_KEY3	PRNG_BA+0x01C	R	PRNG Generated Key3	0xFFFF_FFFF
PRNG_KEY4	PRNG_BA+0x020	R	PRNG Generated Key4	0xFFFF_FFFF
PRNG_KEY5	PRNG_BA+0x024	R	PRNG Generated Key5	0xFFFF_FFFF
PRNG_KEY6	PRNG_BA+0x028	R	PRNG Generated Key6	0xFFFF_FFFF
PRNG_KEY7	PRNG_BA+0x02C	R	PRNG Generated Key7	0xFFFF_FFFF

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

Bits	Description	
[31:0]	**KEY**	**Store PRNG Generated Key (Read Only)** The bits store the key that is generated by PRNG.

6.20 Enhanced 12-bit Analog-to-Digital Converter (EADC)

6.20.1 Overview

The chip contains one 12-bit successive approximation analog-to-digital converter (SAR EADC converter) with 24 external input channels and 3 internal channels. The EADC converter can be started by software trigger, EPWM0/1 triggers, BPWM0/1 triggers, timer0~3 overflow pulse triggers, ADINT0 interrupt EOC (End of conversion) and ADINT1 interrupt EOC pulse trigger and external pin (EADC0_ST) input signal.

6.20.2 Features

- Analog input voltage range: 0~ V_{REF} (Max to 5.5V)
- Reference voltage from V_{REF} pin or AV_{DD}
- 12-bit resolution
- Up to 24 single-end analog external input channels or 1 pair differential analog input channels
- Up to 3 internal channels, they are band-gap voltage (V_{BG}), temperature sensor (V_{TEMP}) , and DAC0 output
- Four EADC interrupts (ADINT0~3) with individual interrupt vector addresses for EADC
- Maximum EADC clock frequency is 36 MHz for EADC
- Up to 1.895 MSPS conversion rate for EADC
- Supports calibration function and calibration interrupt
- Supports internal reference voltage V_{REF}: 2.048V, 2.56V, 3.072V, and 4.096V.
- Supports power-down mode and ultra low frequency mode
- Up to 27 sample modules
  - Sample modules0~23 is configurable for EADC converter channel (EADC_CH0~23) and trigger source for each EADC
  - Sample module 24~26 is fixed for channel 24, 25, 26 input sources as band-gap voltage, temperature sensor, and DAC output
  - Double buffer for sample control logic module 0~3
  - Configurable sampling time for each sample module
  - Conversion results are held in 27 data registers with valid and overrun indicators.
- Averaging (2^n times, n=0~8) to support up to 12-bit result and over-sampling, or called Accumulation, (2^n times, n=0~8) to support up to 16-bit result
- Any EADC conversion of each EADC can be started by:
  - Write 1 to SWTRGn (EADC_SWTRG[n], n = 0~26)
  - External pin EADC0_ST
  - Timer0~3 overflow pulse triggers
  - ADINT0 and ADINT1 interrupt EOC (End of conversion) pulse triggers
  - EPWM/BPWM triggers
- Supports PDMA transfer
- Conversion Result Monitor by Compare Mode

6.20.3 Block Diagram

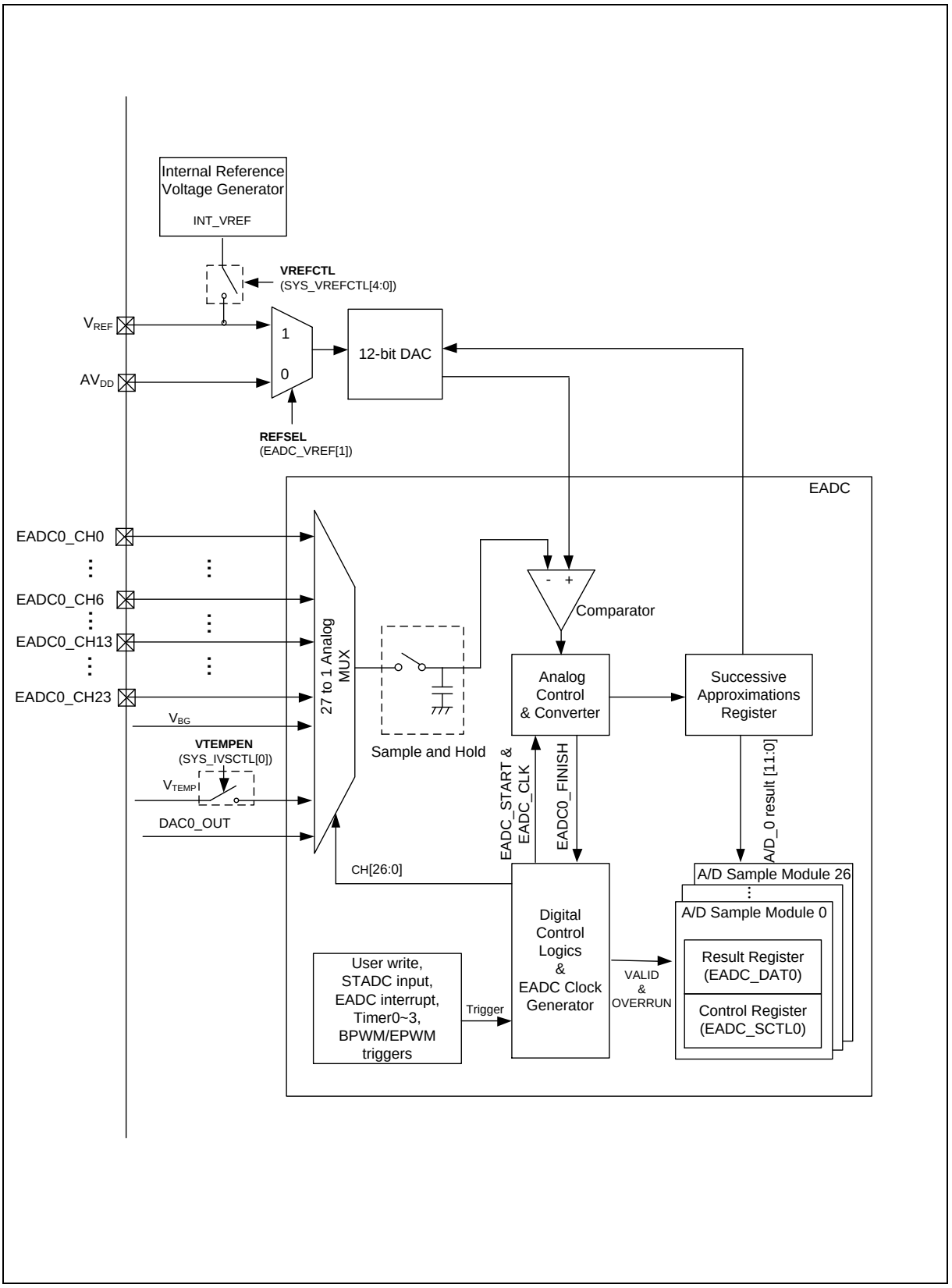


Figure 6.20-1 EADC Converter Block Diagram

6.20.4 Basic Configuration

- Clock source Configuration
  - Select the clock divider number on EADCDIV (CLK_CLKDIV0[23:16])
  - Enable EADC peripheral clock in EADCCKEN (CLK_APBCLK0[28]).
- Reset Configuration
  - Reset EADC controller in EADCRST (SYS_IPRST1[28]).
- Pin configuration

Group	Pin Name	GPIO	MFP
EADC0	EADC0_CH0	PB.0	MFP1
	EADC0_CH1	PB.1	MFP1
	EADC0_CH2	PB.2	MFP1
	EADC0_CH3	PB.3	MFP1
	EADC0_CH4	PB.4	MFP1
	EADC0_CH5	PB.5	MFP1
	EADC0_CH6	PB.6	MFP1
	EADC0_CH7	PB.7	MFP1
	EADC0_CH8	PB.8	MFP1
	EADC0_CH9	PB.9	MFP1
	EADC0_CH10	PB.10	MFP1
	EADC0_CH11	PB.11	MFP1
	EADC0_CH12	PB.12	MFP1
	EADC0_CH13	PB.13	MFP1
	EADC0_CH14	PB.14	MFP1
	EADC0_CH15	PB.15	MFP1
	EADC0_CH16	PD.10	MFP1
	EADC0_CH17	PD.11	MFP1
	EADC0_CH18	PD.12	MFP1
	EADC0_CH19	PC.13	MFP1
	EADC0_CH20	PA.8	MFP1
	EADC0_CH21	PA.9	MFP1
	EADC0_CH22	PA.10	MFP1
	EADC0_CH23	PA.11	MFP1
	EADC0_ST	PF.5	MFP11
		PC.13, PD.12	MFP14
		PC.1, PG.15, PI.2	MFP15

6.20.5 Functional Description

The EADC controller consists of a 27 channel analog switch, 27 sample modules and a 12-bit successive approximation analog-to-digital converter. The EADC operation is based on sample module 0~26, and sample module 0~23 has its configuration to decide which trigger source to start the conversion, which channel to convert. Sample module 0~23 can be configured to EADC0_CH0~23 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 4~23 are shows as follows. Note that sample 24~26 is fixed for internal channel of band-gap voltage, (V_{BG}), temperature sensor, and DAC0 output.

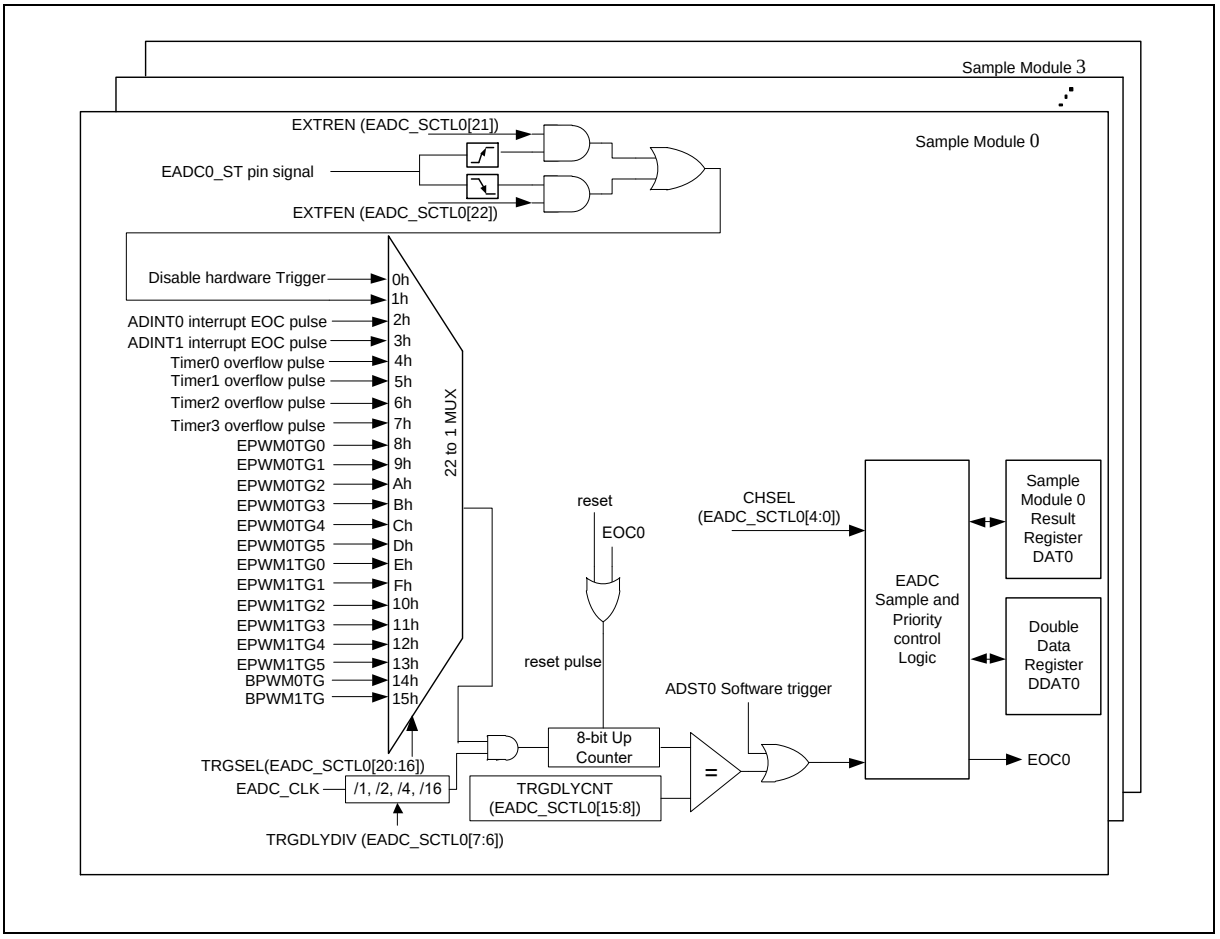


Figure 6.20-2 Sample Module 0~3 Block Diagram

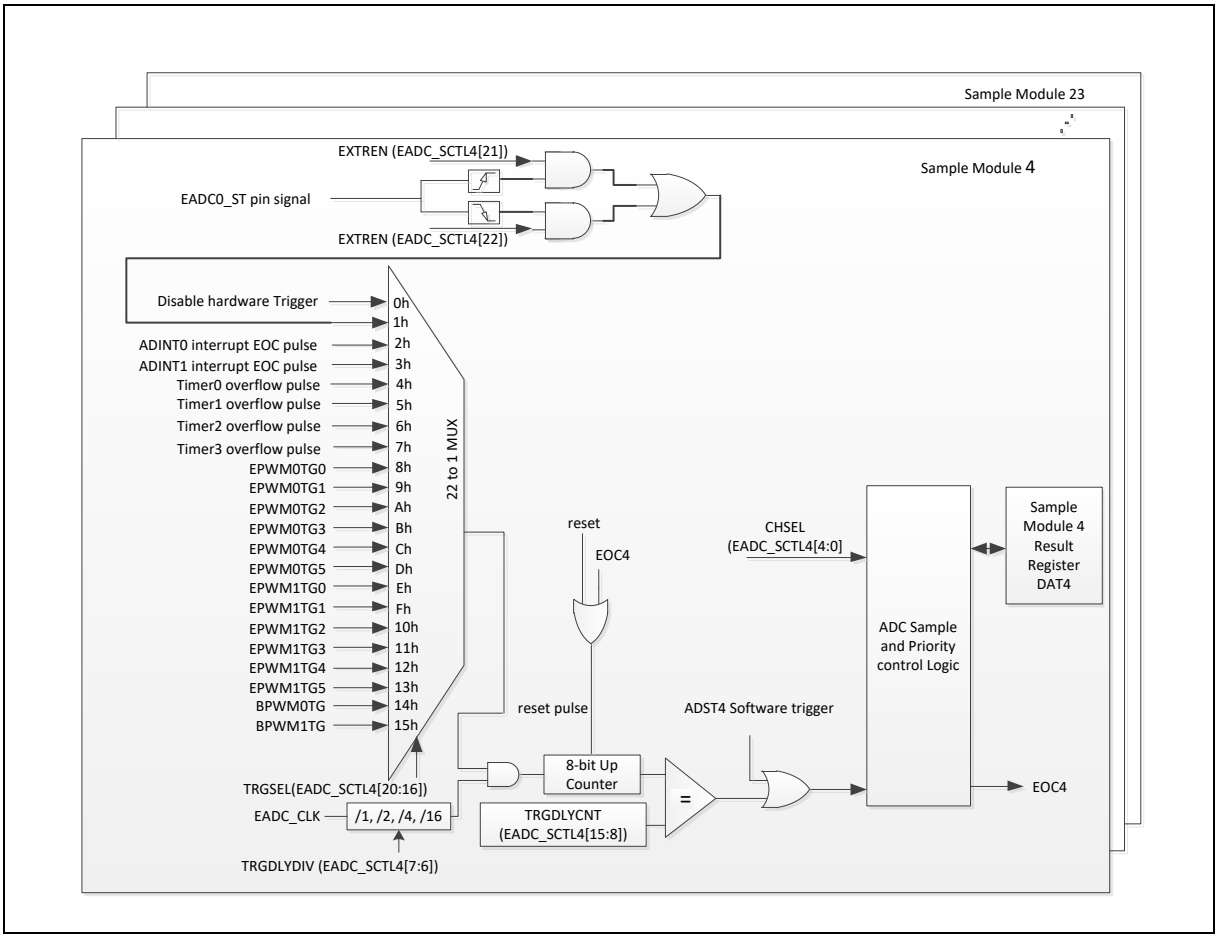


Figure 6.20-3 Sample Module 4~23 Block Diagram

Sample module 24~26 can convert internal channel (V_{BG}, V_{TEMP}, DAC0_OUT) and can be triggered by user write SWTRGn (EADC_SWTRG[n], n = 16~18). Figure 6.20-4 shows the sample module 24~26.

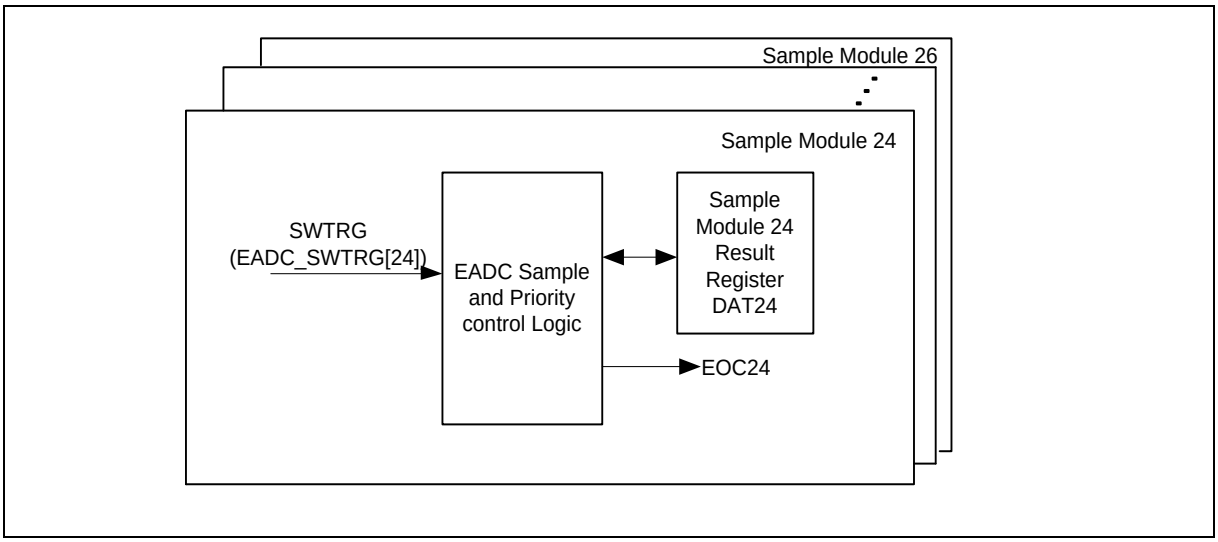


Figure 6.20-4 Sample Module 24~26 Block Diagram

The EADC conversion trigger sources in sample module 0~23 are listed below:

- Write 1 to SWTRGn (EADC_SWTRG[n], n = 0~23)
- External pin EADC0_ST
- Timer0~3 overflow pulse triggers
- ADINT0, ADINT1 EADC interrupt EOC (End of conversion) pulse triggers
- EPWM/BPWM triggers

The ADINT0 or ADINT1 interrupt pulses are generated whenever the specific sample module EADC EOC (End of conversion) pulse is generated. ADINT0 or ADINT1 interrupt pulse triggers can be fed back to trigger another EADC conversion, and is useful if a continuous scan conversion is needed.

6.20.5.1 EADC Clock Generator

The maximum EADC clock frequency is up to 36 MHz and the maximum sampling rate is up to 1.895 MSPS.

The clock control of EADC is shown as Figure 6.20-5. The EADC peripheral clock source is from PCLK1 clock, the EADC clock frequency is divided by an 8-bit pre-scalar with the following formula:

EADC clock frequency = (PCLK1) / (EADCDIV/1 (CKL_CLKDIV0[23:16])+1)

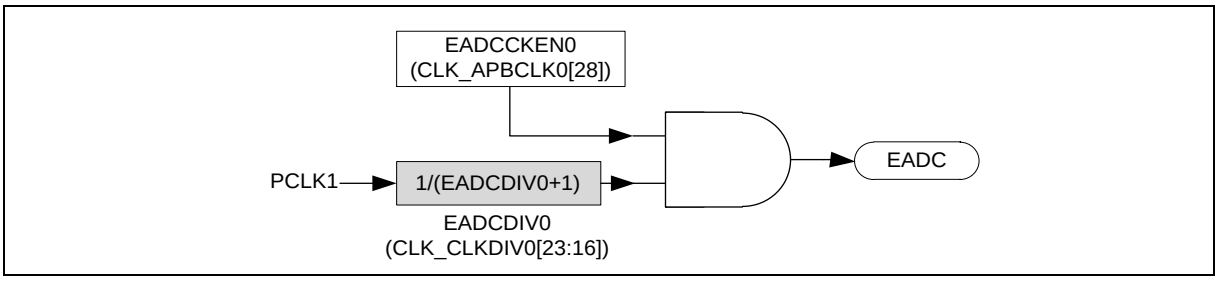


Figure 6.20-5 EADC Clock Control

6.20.5.2 EADC Software Trigger Mode

When a EADC conversion is performed on the sample module specified single channel, the operations are as follows:

1. EADC conversion is started when the SWTRGn (EADC_SWTRG[n], n=0~26) is set to 1 by user or other trigger inputs.
2. When EADC conversion is finished, the 12-bit result is stored in the EADC data register EADC_DATn (n=0~26) corresponding to the sample module.
3. On completion of conversion, the ADIFn (EADC_STATUS2[3:0], n=0~3) is set to 1 and EADC interrupt (ADINTn, n=0~3) is requested if the ADCIENn (EADC_CTL[5:2], n=0~3) bit is set to 1.
4. When EADC conversion ends, the SWTRGn (n=0~26) bit is automatically cleared to 0 and the EADC converter will do another pending conversion.

The timing diagram of a conversion cycle is shown in Figure 6.20-6.

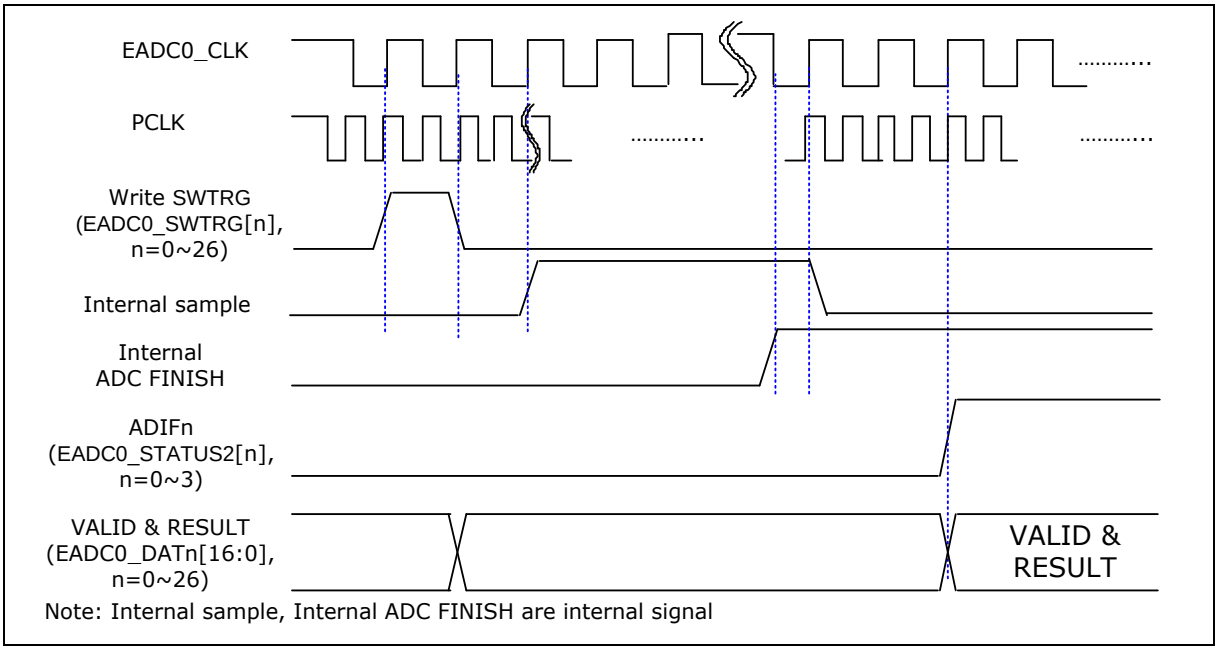


Figure 6.20-6 Example EADC Conversion Timing Diagram, n=0~26

If more than one sample module is enabled to convert analog signal, 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 26 is lowest priority.

6.20.5.3 EADC 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 6.20-7. 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=0~26) are set to 1 by HW. After the Sample Module finish the conversion, STPF(EADC_PENDSTS[n], n=0~26) 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=0~26) 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.

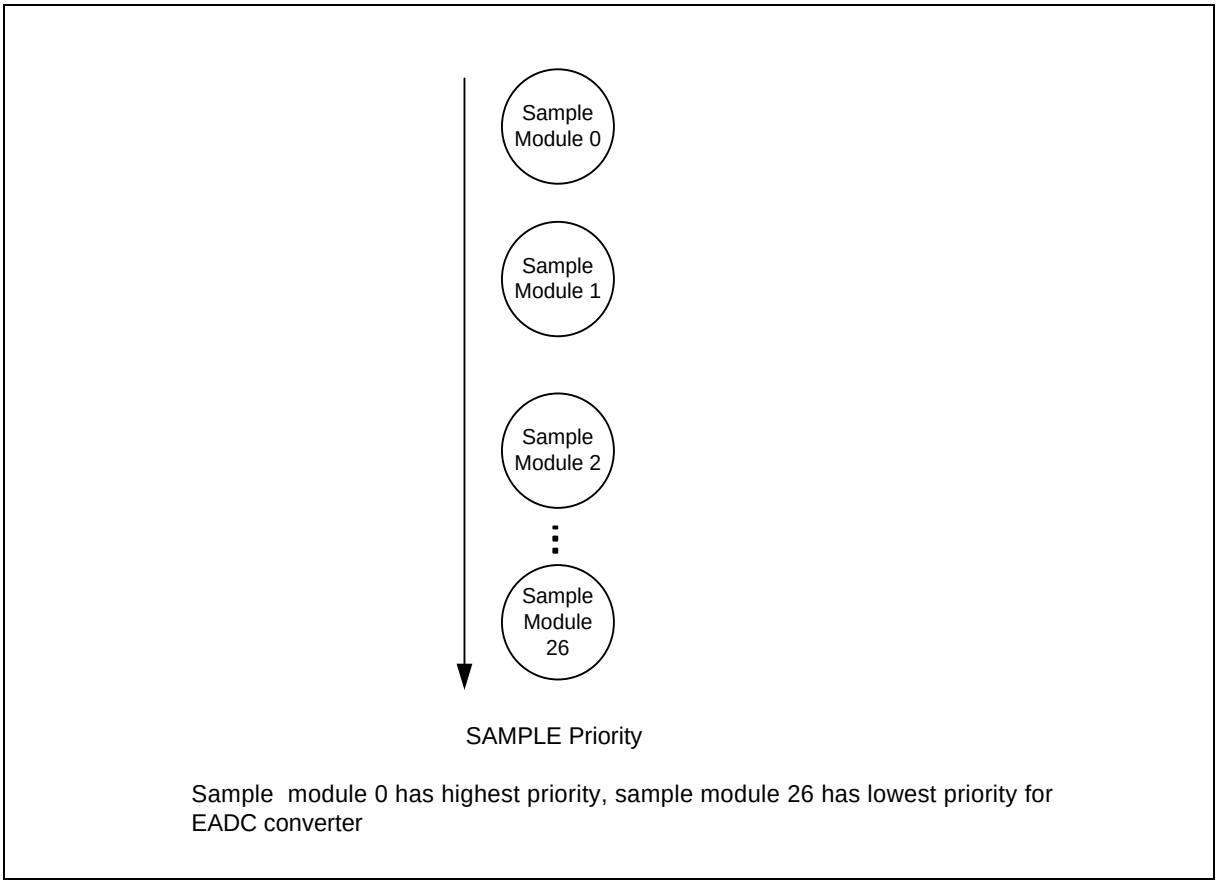


Figure 6.20-7 Sample Module Conversion Priority Arbitrator Diagram

6.20.5.4 Maximum Sampling Frequency Conversion by Software Trigger

If user needs to geet maximum sampling frequency, the conversion needs 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 0~15 to carry out successive convertion. Set CHSEL (EADC_SCTL0~15[4:0]) as one of fast channel (EADC0_CH10~ EADC0_CH15). Set EXTSMPT (EADC_SCTL0~15[31:24]) and TRGDLYCNT (EADC_SCTL0~15[15:8]) as 0x00 to minimize the sampling time.
2. Set SWTRG (EADC_SWTRG[26:0]) as 0xffff to trigger Module 0~15.
3. Wait CURSPL (EADC_STATUS3[4:0]) changes to 0xf which means Module 0~14 have been executed and Module 15 is in the process. Set SWTRG (EADC_SWTRG[26:0]) as 0x7fff to trigger Module 0~14 again for next round.
4. Wait CURSPL (EADC_STATUS3[4:0]) changes to 0x1, set SWTRG (EADC_SWTRG[26:0]) as 0x8000 to trigger Module 15.
5. Repeat Step 3~4 to continue the conversion.

6.20.5.5 EADC Sample Module End of Conversion Interrupt Operation

There are 4 EADC 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 0~26 End of conversion pulses) as its interrupt trigger source. Figure 6.20-8 shows the control logic of interrupts. Take ADINT0 as an example, when ADCIEN0 (EADC_CTL[2]) = 1 and SPLIEn (EADC_INTSRC0[n]) = 1 (n=0~26), the specific module EOC (End of conversion) pulses will set flag ADIF0

(EADC_STATUS2[0]) as 1 and interrupt (ADINT0) will be asserted either.

The interrupt pulses (ADINT0/1) are generated whenever the specific sample module EADC EOC pulse is generated. It also can be the sample module conversion trigger sources, and user can use it to do the EADC continuous scan conversion.

The example of continuous scan triggered by interrupt is as follows:

1. If EADC 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 EADC conversion, after the conversion completes, it generates an EOC2 pulse signal and ADINT0 interrupt pulse at end of sample module 2 EADC conversion, ADINT0 interrupt pulse will trigger the sample module 0, 1, 2 to start the EADC conversions.
3. ADINT0 interrupt pulse repeats to trigger sample module 0, 1, 2 EADC 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: The average conversion cycles of continuous scan triggered by interrupt is 19 EADC_CLK.

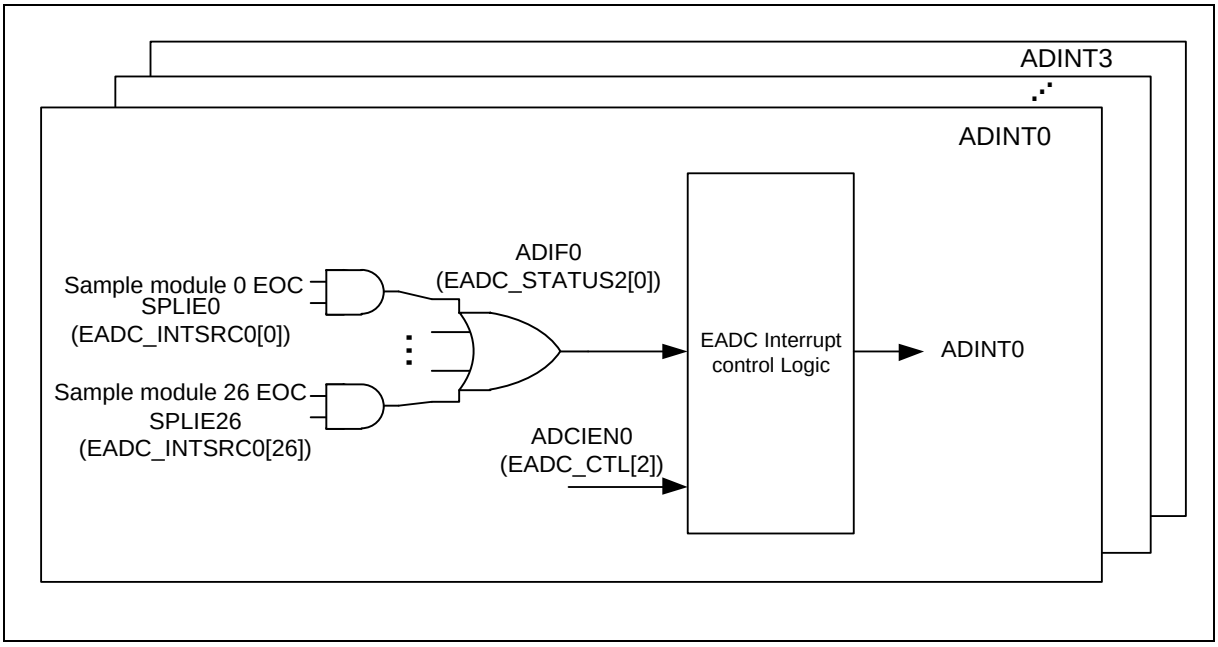


Figure 6.20-8 Specific Sample Module EADC EOC Signal for ADINT0~3 Interrupt

6.20.5.6 EADC 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 0~23 to trigger EADC start when Timer overflow occurs.

6.20.5.7 EADC Start Synchronous with EPWM/BPWM Trigger

Besides user start, ADINT0/1 interrupt pulse, external pin EADC0_ST and Timer0~3 overflow pulse to start EADC conversion, this device has new feature to allow EPWM/BPWM channels to trigger the EADC start. User may configure EPWM/BPWM trigger types: rising, falling EPWM/BPWM edge or center point of EPWM/BPWM (center-aligned mode only) to trigger EADC start. The device also allows user to configure the amount of delay period to EADC start after hardware detected the external trigger. User can configure the trigger delay time by setting TRGDLYCNT

(EADC_SCTLn[15:8], n=0~23) and TRGDLYDIV (EADC_SCTLn[7:6], n=0~23). Figure 6.20-9 shows the programmable delay time for EPWM/BPWM-triggered EADC start conversion.

Figure 6.20-10 shows the programmable delay time for other trigger source.

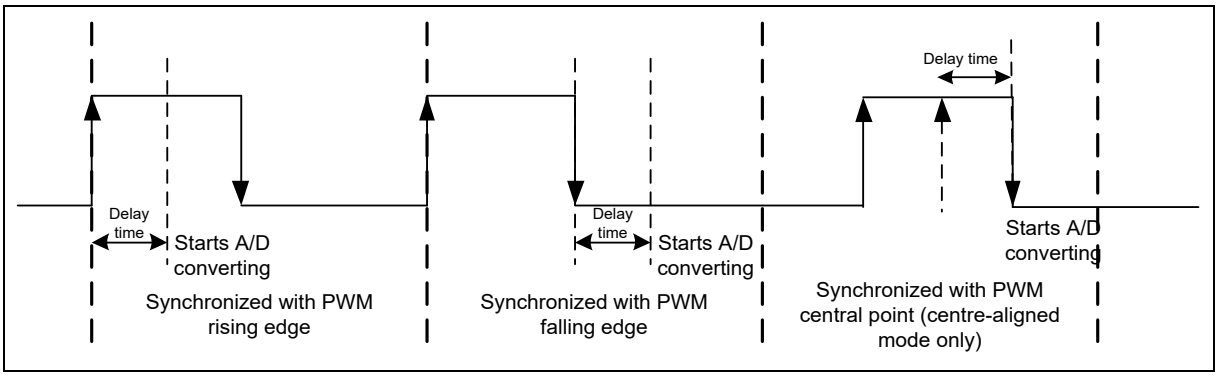


Figure 6.20-9 EPWM-triggered EADC Start Conversion

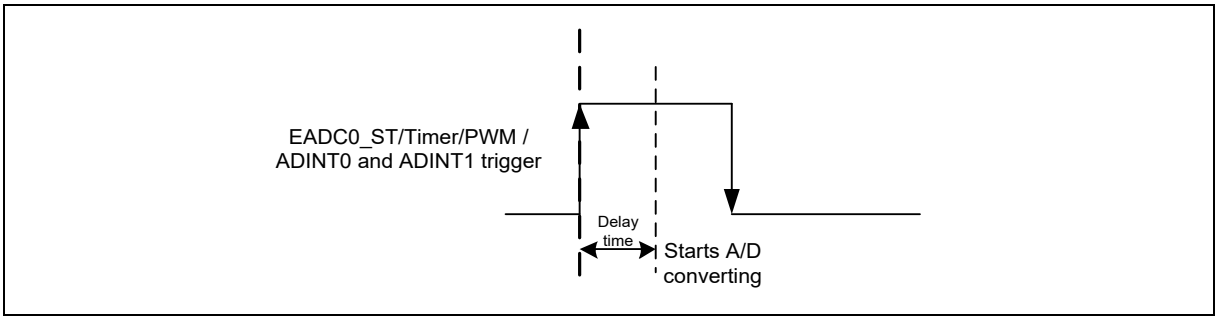


Figure 6.20-10 External triggered EADC Start Conversion

6.20.5.8 EADC Conversion Time and External Trigger

The EADC converter sample the analog input when EADC conversion start delay time (Td) has passed after SWTRGn (EADC_SWTRG[n], n=0~26) is set to 1, then start conversion. Due to EADC clock is generated by PCLK divided by (EADCDIV (CLKDIV0[23:16])+1), the maximum delay time from user write SWTRGn to EADC start sampling analog input time is four EADC clock cycles that operate for synchronization and control pre-processingThe start delay time is shown in Figure 6.20-11.

EADC conversion can be triggered by external pin EADC0_ST request. Setting the TRGSEL (EADC_SCTLn[20:16], n=0~23) to 0x01 is to select external trigger input from the EADC0_ST pin. User can set EXTFEN (EADC_SCTLn[22], n=0~23) and EXTREN (EADC_SCTLn[21], n=0~23) 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 6.20-12.

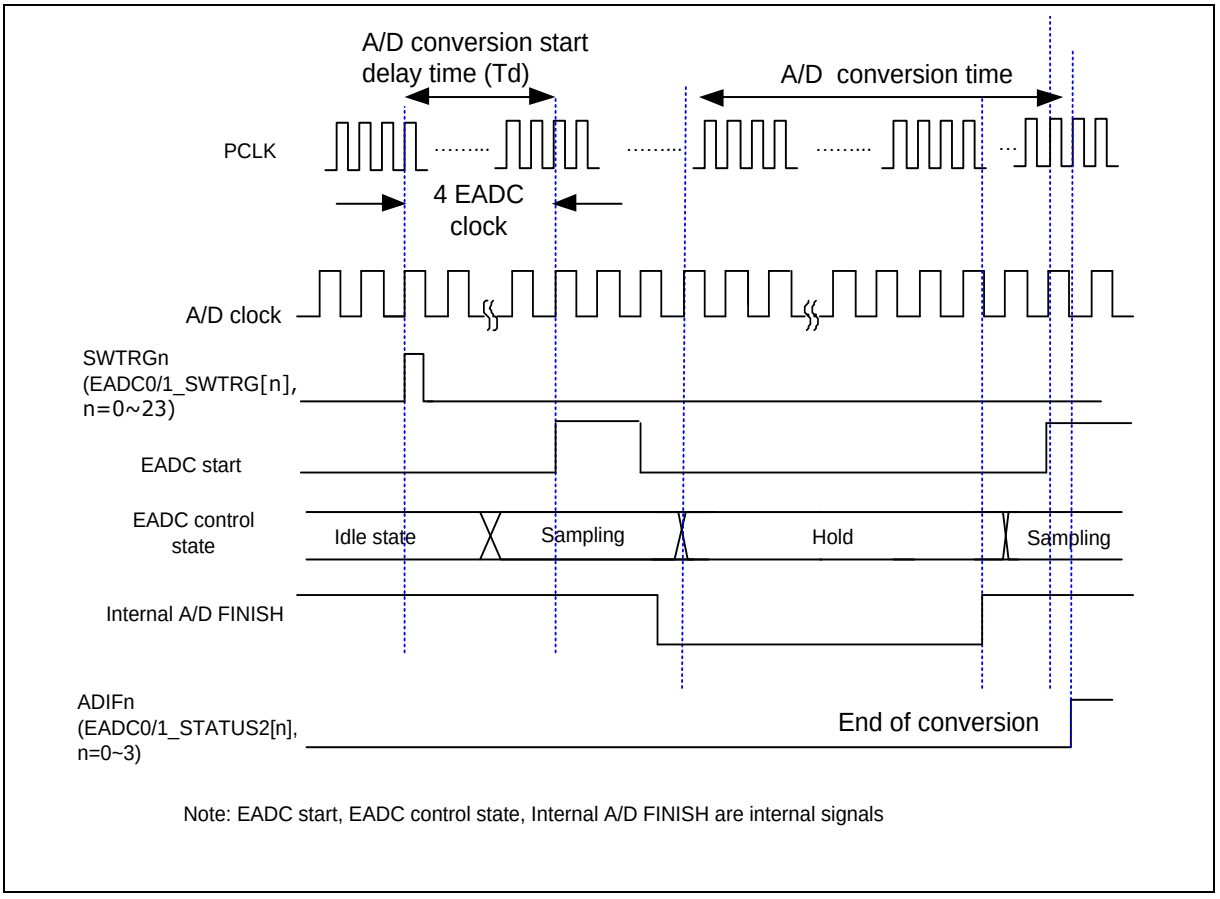


Figure 6.20-11 Conversion Start Delay Timing Diagram

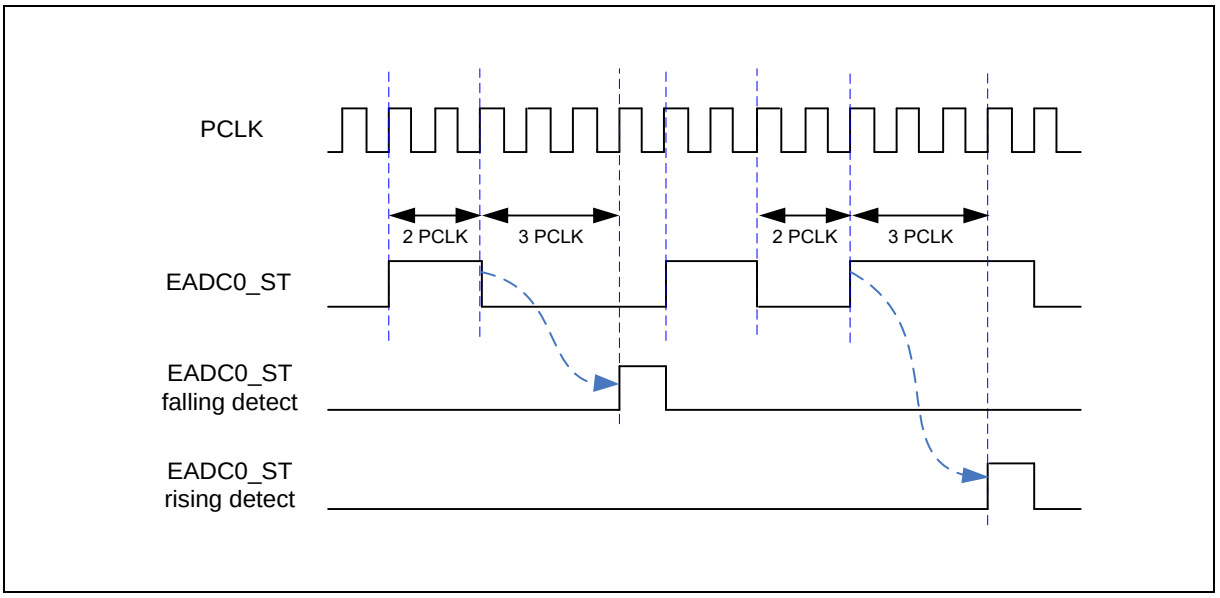


Figure 6.20-12 EADC0_ST De-bounce Timing Diagram

6.20.5.9 EADC Extend Sampling Time

When EADC operates at high EADC clock rate, the sampling time of analog input voltage may not be 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=0~26) for each sample module. The EADC extend sampling time is present between EADC controller judge which channel to be converting and EADC start to conversion. The range of setting extend sampling time is from 0~255 EADC clock. For example, EADC sample module2 is configured with extend sampling time, as shown in Figure 6.20-13.

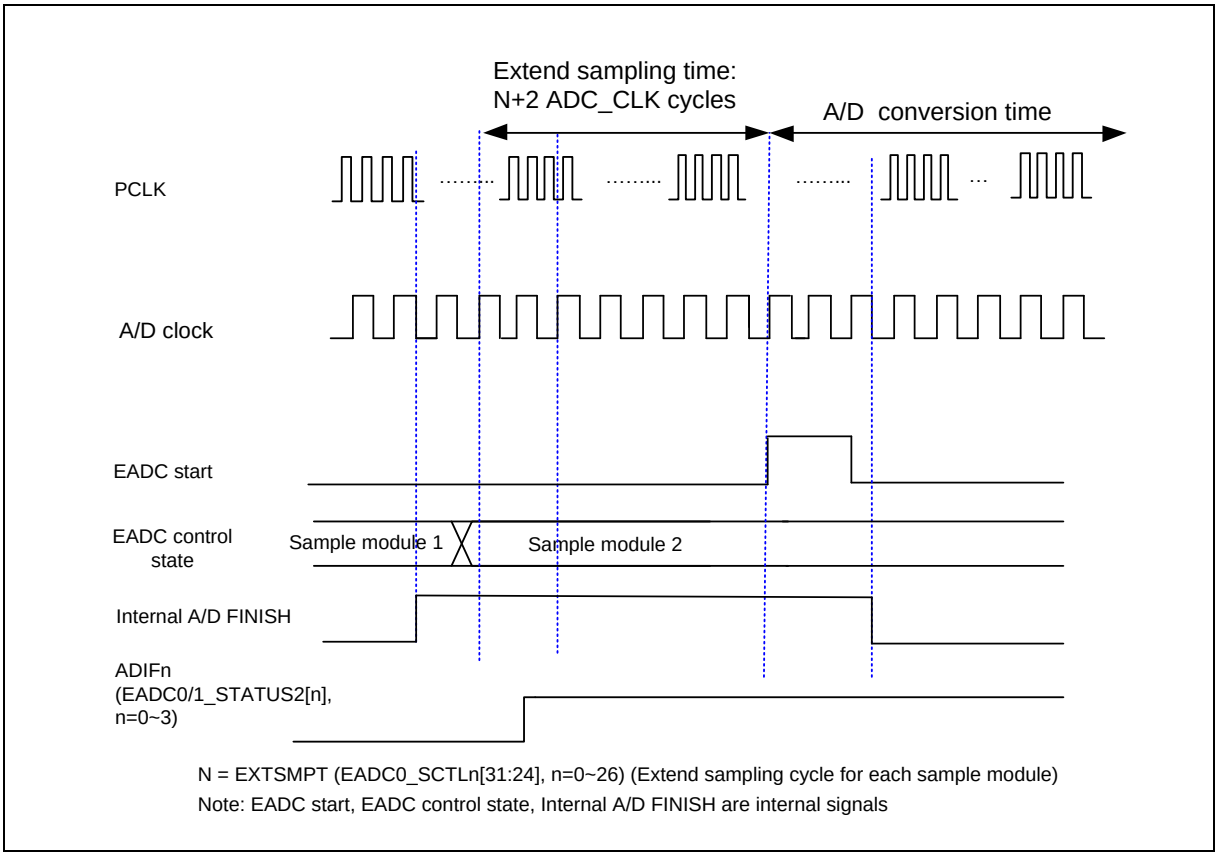


Figure 6.20-13 EADC Extend Sampling Timing Diagram

6.20.5.10 Oversampling Mode

The function provides multiple consecutive conversion results to be accumulated to a final conversion result. Setting ACU (EADC_MnCTL1[7:4], n=0~23) determine whcich sample module and the number of samples to be accumulated. It is noted that when more than 16 samples to be accumulated, the final result after accumulating will exceed the 16-bit RESULT(EADC_DATn [0:15], n=0~23) register size. In this case, the result is right shifted automatically to fit within the appropriate register size. It is noted that accumulation function is only effective without data left alignment.

ACU (EADC_MnCTL1 [7:4]), N=0~23	Number Of Samples To Be Accumulated	Right Shift Division Factor	Final Result Precision
0x0	1	1	12 bits
0x1	2	1	13 bits
0x2	4	1	14 bits
0x3	8	1	15 bits
0x4	16	1	16 bits

0x5	32	2	16 bits
0x6	64	4	16 bits
0x7	128	8	16 bits
0x8	256	16	16 bits

Table 6.20-1 Setting of Accumulation and Conversion Result Precision

6.20.5.11 Averaging Mode

The function increasing the sample accuracy with reduced sampling rate and the feature is suitable in noisy operating environment for getting more stable average conversion results. According to the accumulated final result in 6.20.5.10, setting AVG (EADC_MnCTL1[1]) can enable averaging function as shown in Table 6.20-2. The automatic division factor is decided by the number of accumulated samples and the final result will get 12 bits precision. It is noted that averaging is only effective when accumulating two more samples without data left alignment.

AVG EADC_MnCTL1 [1]	ACU EADC_MnCTL1 [7:4]	Number Of Samples To Be Accumulated	Automatic Division Factor	Final Result Precision
0x1	0x0	1	1	12 bits
0x1	0x1	2	2	12 bits
0x1	0x2	4	4	12 bits
0x1	0x3	8	8	12 bits
0x1	0x4	16	16	12 bits
0x1	0x5	32	32	12 bits
0x1	0x6	64	64	12 bits
0x1	0x7	128	128	12 bits
0x1	0x8	256	256	12 bits

Table 6.20-2 Setting of Averaging and Conversion Result Precision

6.20.5.12 Conversion Result Monitor by Compare Mode

The EADC controller provides four sets of compare registers EADC_CMP0 ~ EADC_CMP3 to monitor a maximum of four specified sample module 0~26 conversion results from EADC conversion module, as shown in Figure 6.20-14. 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], where 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], where n =0~3). When the conversion of the sample module specified by CMPSPL 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, where n =0~3) then EADCMPFn (EADC_STATUS2[7:4], where 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 6.20-14.

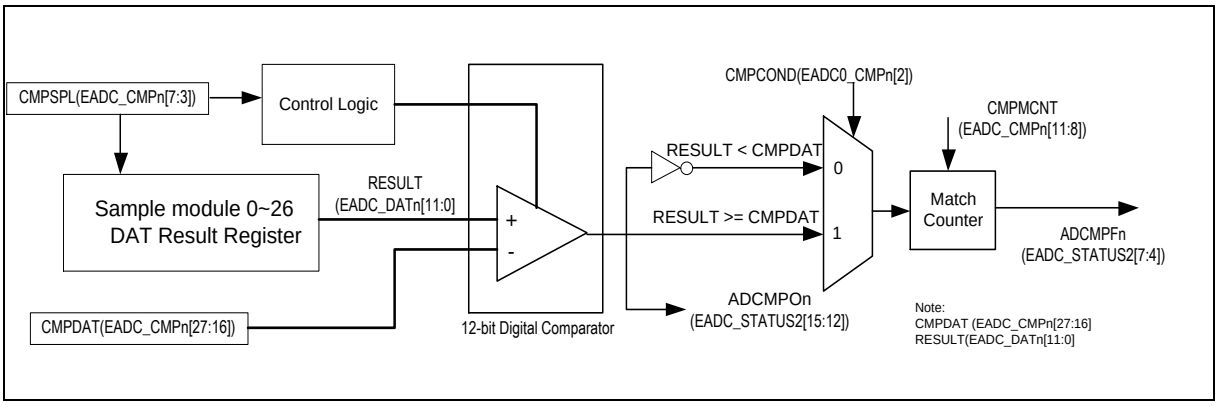


Figure 6.20-14 EADC Conversion Result Monitor Logics Diagram

The EADC controller supports a window compare mode. User can set CMPWEN (EADC_CMP0[15]/ EADC_CMP2[15]) to enable this function. If user enables this function, EADCMPF0 (EADC_STATUS2[4]) will be set when both EADC_CMP0 and EADC_CMP1 compared condition matched. EADCMPF2 (EADC_STATUS2[6]) will be set when both EADC_CMP2 and EADC_CMP3 compared condition matched.

6.20.5.13 Differential Mode

The EADC controller supports analog fully-differential mode. A fully-differential SAR converts the differential voltage across its inputs; however, in this case both inputs change dynamically and are complementary or inverted from each other. In this case, the VINP and VINM pins must be driven 180° out of phase with respect to each other, centered on a fixed common mode voltage, for example VREF/2. Fully-differential EADCs have an input voltage common-mode (VCM) range specification. VCM is defined as the average voltage between the inputs : VCM = (VINP + VINM) / 2. Most fully-differential input SAR EADCs prohibit the input common-mode voltage from varying more than approximately 10 percent beyond the mid-scale input (V_{REF} /2).

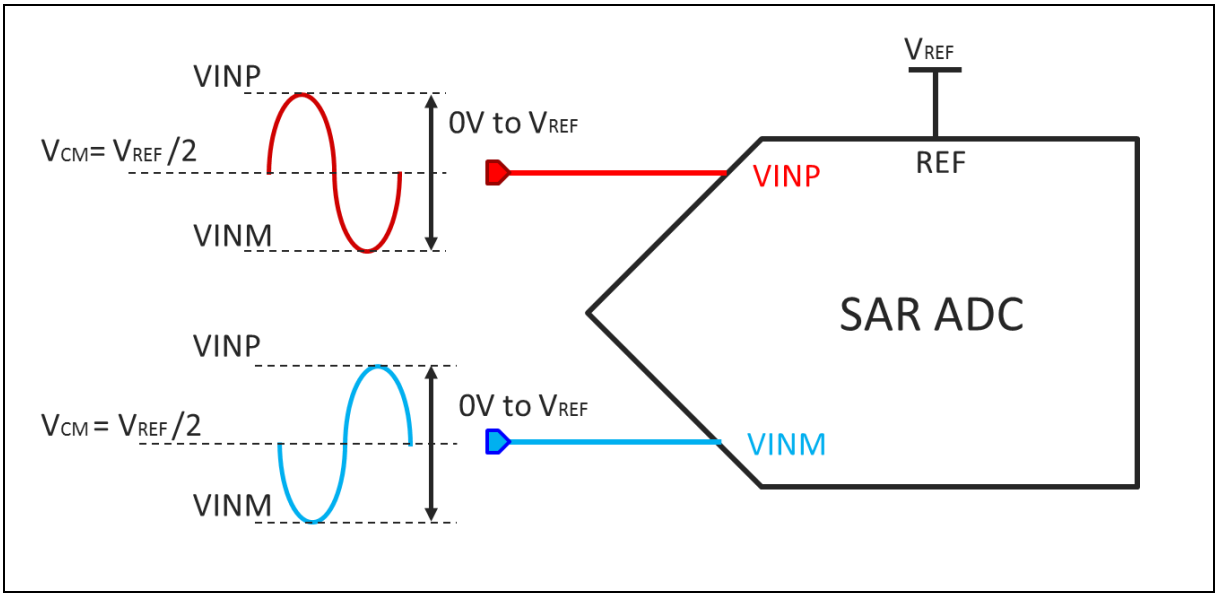


Figure 6.20-15 Input of Fully-Differential EADC

If user enables DIFFEN (EADC_CTL[8]), the differential mode will enable. The pair of analog input

channel is as Table 6.20-3.

Differential Analog Input Paired Channel	ADC Analog Input	
	V_{plus}	V_{minus}
0	EADC_CH15	EADC_CH14

Table 6.20-3 EADC Differential Model Channel Selection

In differential analog input mode, only the even number of the two corresponding channels needs to be enabled in CHSEL (EADC_SCTLn[4:0]). The conversion result will be placed to the corresponding data register of the enabled channel.

6.20.5.14 Double Buffer Mode

The EADC 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 EADC 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 EADC converts finish, VALID (EADC_DDATn[17], n=0~3) will set to high either. Then, user can get the EADC results from EADC_DATn and EADC_DDATn register.

6.20.5.15 PDMA Request

The EADC controller supports PDMA. PDMA could service each channel when corresponding channel PDMA transfer enanble bit, PDMATEN (EADC_PDMACTL[26:0]), is activated. For example, user can enable PDMATEN for specific channels and configure PDMA channel's source address as CURDAT(EADC_CURDAT[26:0]). After enabling PDMATEN and PDMA channel, if any VALID (EADC_DATn[17],n=0~26) is high, EADC 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 EADC_DAT register will be automatically cleared.

6.20.5.16 Interrupt Sources

The EADC 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[5], n=0~23). If EADCIENn (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 6.20-16.

It is noted that calibration interrupt CALINT is also shown in Figure 6.20-16.

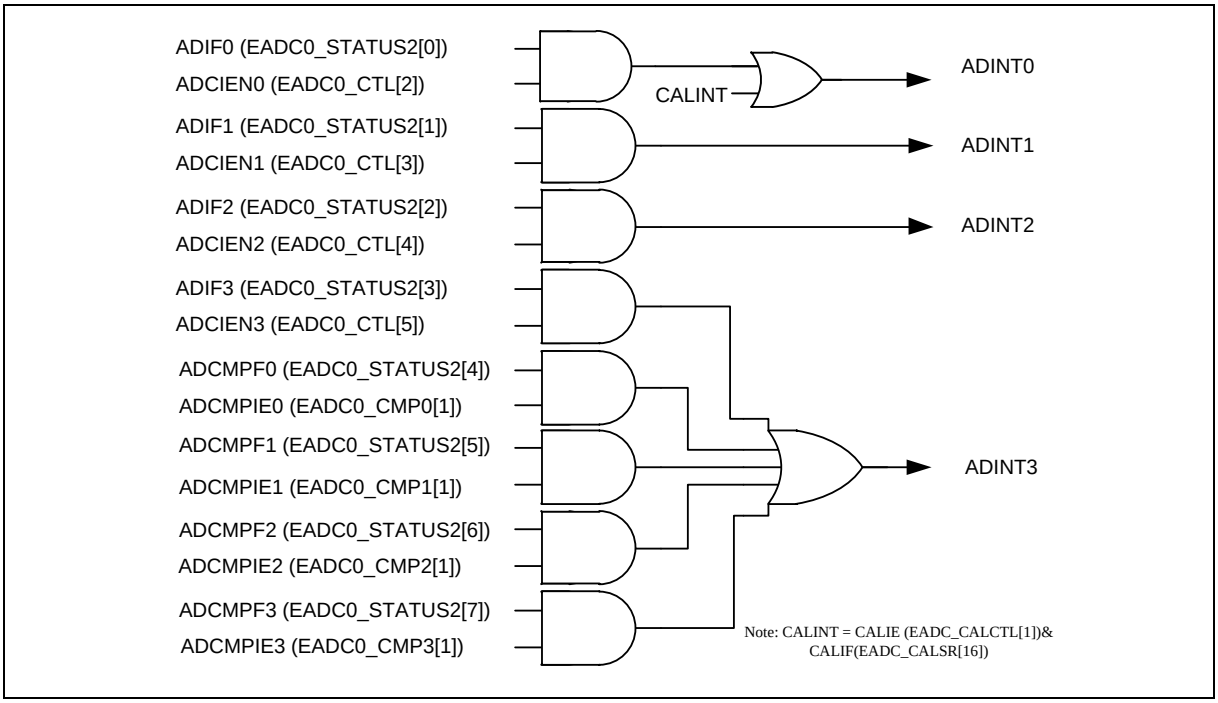


Figure 6.20-16 EADC Controller Interrupts

6.20.5.17 Calibration Mode

To decrease the effect of electrical random noise, the calibration mode performs an offset and mismatch measurement cycles. Afterwards, in normal operation mode, the calibration engine applies to the capacitor array, so that the offset and mismatch are removed. If chip power off, calibration function should be executed again.

In Calibration Mode:

1. Set CAL(EADC_CALCTL[0]) bit to 1.
2. After 3 ADC_CLK when CAL is set to 1, calibration is starting.
3. After calibration is finished, the flag CALIF(EADC_CALSR[16]) will be set to 1. If CALIE (EADC_CALCTL[1]) is set, the calibration interrupt will be generated.
4. After calibration is finished, the mode will become Normal Mode and CAL is cleared by hardware automatically.

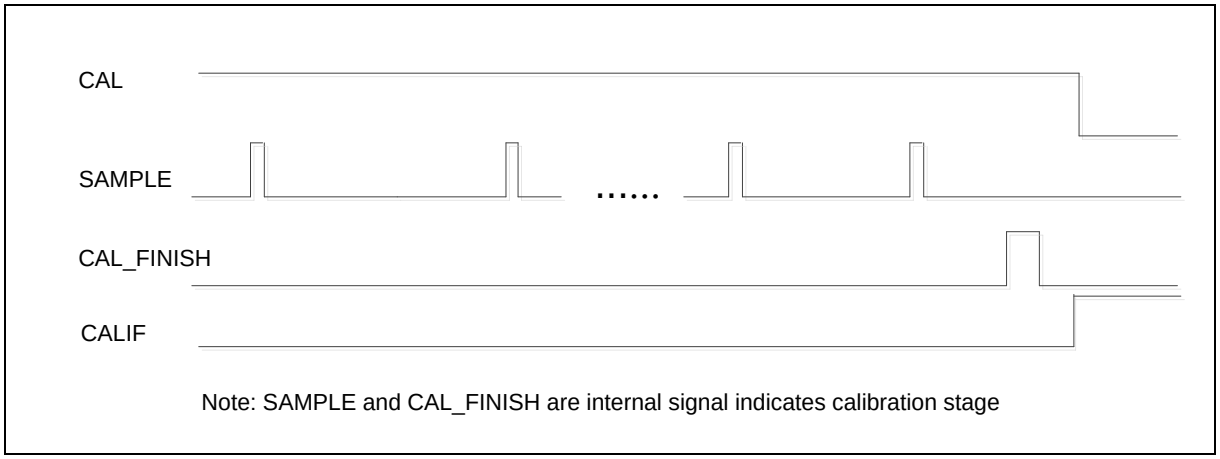


Figure 6.20-17 Calibration Mode with 16 Times Average

6.20.5.18 Reference Voltage Source Select

ADC bases on reference voltage to convert an analog signal to digital value. The reference voltage source can be selected from AV_{DD} or V_{REF} by REFSEL (EADC_VREF[1]). And the V_{REF} can select as external pin, internal reference voltage 2.048V, 2.56V, 3.072V and 4.096V by VREFCTL (SYS_VREFCTL[4:0]).

6.20.6 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	EADC Data Register 0 for Sample Module 0	0x0000_0000
EADC_DAT1	EADC_BA+0x04	R	EADC Data Register 1 for Sample Module 1	0x0000_0000
EADC_DAT2	EADC_BA+0x08	R	EADC Data Register 2 for Sample Module 2	0x0000_0000
EADC_DAT3	EADC_BA+0x0C	R	EADC Data Register 3 for Sample Module 3	0x0000_0000
EADC_DAT4	EADC_BA+0x10	R	EADC Data Register 4 for Sample Module 4	0x0000_0000
EADC_DAT5	EADC_BA+0x14	R	EADC Data Register 5 for Sample Module 5	0x0000_0000
EADC_DAT6	EADC_BA+0x18	R	EADC Data Register 6 for Sample Module 6	0x0000_0000
EADC_DAT7	EADC_BA+0x1C	R	EADC Data Register 7 for Sample Module 7	0x0000_0000
EADC_DAT8	EADC_BA+0x20	R	EADC Data Register 8 for Sample Module 8	0x0000_0000
EADC_DAT9	EADC_BA+0x24	R	EADC Data Register 9 for Sample Module 9	0x0000_0000
EADC_DAT10	EADC_BA+0x28	R	EADC Data Register 10 for Sample Module 10	0x0000_0000
EADC_DAT11	EADC_BA+0x2C	R	EADC Data Register 11 for Sample Module 11	0x0000_0000
EADC_DAT12	EADC_BA+0x30	R	EADC Data Register 12 for Sample Module 12	0x0000_0000
EADC_DAT13	EADC_BA+0x34	R	EADC Data Register 13 for Sample Module 13	0x0000_0000
EADC_DAT14	EADC_BA+0x38	R	EADC Data Register 14 for Sample Module 14	0x0000_0000
EADC_DAT15	EADC_BA+0x3C	R	EADC Data Register 15 for Sample Module 15	0x0000_0000
EADC_DAT16	EADC_BA+0x40	R	EADC Data Register 16 for Sample Module 16	0x0000_0000
EADC_DAT17	EADC_BA+0x44	R	EADC Data Register 17 for Sample Module 17	0x0000_0000
EADC_DAT18	EADC_BA+0x48	R	EADC Data Register 18 for Sample Module 18	0x0000_0000
EADC_CURDAT	EADC_BA+0x4C	R	EADC PDMA Current Transfer Data Register	0x0000_0000
EADC_CTL	EADC_BA+0x50	R/W	EADC Control Register	0x0000_0000

Register	Offset	R/W	Description	Reset Value
EADC_SWTRG	EADC_BA+0x54	W	EADC Sample Module Software Start Register	0x0000_0000
EADC_PENDSTS	EADC_BA+0x58	R/W	EADC Start of Conversion Pending Flag Register	0x0000_0000
EADC_OVSTS	EADC_BA+0x5C	R/W	EADC Sample Module Start of Conversion Overrun Flag Register	0x0000_0000
EADC_CTL1	EADC_BA+0x60	R/W	EADC Control1 Register	0x0000_0000
EADC_SCTL0	EADC_BA+0x80	R/W	EADC Sample Module 0 Control Register	0x0000_0000
EADC_SCTL1	EADC_BA+0x84	R/W	EADC Sample Module 1 Control Register	0x0000_0000
EADC_SCTL2	EADC_BA+0x88	R/W	EADC Sample Module 2 Control Register	0x0000_0000
EADC_SCTL3	EADC_BA+0x8C	R/W	EADC Sample Module 3 Control Register	0x0000_0000
EADC_SCTL4	EADC_BA+0x90	R/W	EADC Sample Module 4 Control Register	0x0000_0000
EADC_SCTL5	EADC_BA+0x94	R/W	EADC Sample Module 5 Control Register	0x0000_0000
EADC_SCTL6	EADC_BA+0x98	R/W	EADC Sample Module 6 Control Register	0x0000_0000
EADC_SCTL7	EADC_BA+0x9C	R/W	EADC Sample Module 7 Control Register	0x0000_0000
EADC_SCTL8	EADC_BA+0xA0	R/W	EADC Sample Module 8 Control Register	0x0000_0000
EADC_SCTL9	EADC_BA+0xA4	R/W	EADC Sample Module 9 Control Register	0x0000_0000
EADC_SCTL10	EADC_BA+0xA8	R/W	EADC Sample Module 10 Control Register	0x0000_0000
EADC_SCTL11	EADC_BA+0xAC	R/W	EADC Sample Module 11 Control Register	0x0000_0000
EADC_SCTL12	EADC_BA+0xB0	R/W	EADC Sample Module 12 Control Register	0x0000_0000
EADC_SCTL13	EADC_BA+0xB4	R/W	EADC Sample Module 13 Control Register	0x0000_0000
EADC_SCTL14	EADC_BA+0xB8	R/W	EADC Sample Module 14 Control Register	0x0000_0000
EADC_SCTL15	EADC_BA+0xBC	R/W	EADC Sample Module 15 Control Register	0x0000_0000
EADC_SCTL16	EADC_BA+0xC0	R/W	EADC Sample Module 16 Control Register	0x0000_0000
EADC_SCTL17	EADC_BA+0xC4	R/W	EADC Sample Module 17 Control Register	0x0000_0000
EADC_SCTL18	EADC_BA+0xC8	R/W	EADC Sample Module 18 Control Register	0x0000_0000
EADC_INTSRC0	EADC_BA+0xD0	R/W	EADC Interrupt 0 Source Enable Control Register.	0x0000_0000
EADC_INTSRC1	EADC_BA+0xD4	R/W	EADC Interrupt 1 Source Enable Control Register.	0x0000_0000
EADC_INTSRC2	EADC_BA+0xD8	R/W	EADC Interrupt 2 Source Enable Control Register.	0x0000_0000
EADC_INTSRC3	EADC_BA+0xDC	R/W	EADC Interrupt 3 Source Enable Control Register.	0x0000_0000
EADC_CMP0	EADC_BA+0xE0	R/W	EADC Result Compare Register 0	0x0000_0000
EADC_CMP1	EADC_BA+0xE4	R/W	EADC Result Compare Register 1	0x0000_0000
EADC_CMP2	EADC_BA+0xE8	R/W	EADC Result Compare Register 2	0x0000_0000

Register	Offset	R/W	Description	Reset Value
EADC_CMP3	EADC_BA+0xEC	R/W	EADC Result Compare Register 3	0x0000_0000
EADC_STATUS0	EADC_BA+0xF0	R	EADC Status Register 0	0x0000_0000
EADC_STATUS1	EADC_BA+0xF4	R	EADC Status Register 1	0x0000_0000
EADC_STATUS2	EADC_BA+0xF8	R/W	EADC Status Register 2	0x0019_0000
EADC_STATUS3	EADC_BA+0xFC	R	EADC Status Register 3	0x0000_001F
EADC_DDAT0	EADC_BA+0x100	R	EADC Double Data Register 0 for Sample Module 0	0x0000_0000
EADC_DDAT1	EADC_BA+0x104	R	EADC Double Data Register 1 for Sample Module 1	0x0000_0000
EADC_DDAT2	EADC_BA+0x108	R	EADC Double Data Register 2 for Sample Module 2	0x0000_0000
EADC_DDAT3	EADC_BA+0x10C	R	EADC Double Data Register 3 for Sample Module 3	0x0000_0000
EADC_CALCTL	EADC_BA+0x114	R/W	EADC Calibration Control Register	0x0000_0020
EADC_CALSR	EADC_BA+0x118	R/W	EADC Calibration Status Register	0x0000_0000
EADC_PDMACTL	EADC_BA+0x130	R/W	EADC PDMA Control Register	0x0000_0000
EADC_M0CTL1	EADC_BA+0x140	R/W	EADC Sample Module0 Control Register 1	0x0000_0000
EADC_M1CTL1	EADC_BA+0x144	R/W	EADC Sample Module1 Control Register 1	0x0000_0000
EADC_M2CTL1	EADC_BA+0x148	R/W	EADC Sample Module2 Control Register 1	0x0000_0000
EADC_M3CTL1	EADC_BA+0x14C	R/W	EADC Sample Module3 Control Register 1	0x0000_0000
EADC_M4CTL1	EADC_BA+0x150	R/W	EADC Sample Module4 Control Register 1	0x0000_0000
EADC_M5CTL1	EADC_BA+0x154	R/W	EADC Sample Module5 Control Register 1	0x0000_0000
EADC_M6CTL1	EADC_BA+0x158	R/W	EADC Sample Module6 Control Register 1	0x0000_0000
EADC_M7CTL1	EADC_BA+0x15C	R/W	EADC Sample Module7 Control Register 1	0x0000_0000
EADC_M8CTL1	EADC_BA+0x160	R/W	EADC Sample Module8 Control Register 1	0x0000_0000
EADC_M9CTL1	EADC_BA+0x164	R/W	EADC Sample Module9 Control Register 1	0x0000_0000
EADC_M10CTL1	EADC_BA+0x168	R/W	EADC Sample Module10 Control Register 1	0x0000_0000
EADC_M11CTL1	EADC_BA+0x16C	R/W	EADC Sample Module11 Control Register 1	0x0000_0000
EADC_M12CTL1	EADC_BA+0x170	R/W	EADC Sample Module12 Control Register 1	0x0000_0000
EADC_M13CTL1	EADC_BA+0x174	R/W	EADC Sample Module13 Control Register 1	0x0000_0000
EADC_M14CTL1	EADC_BA+0x178	R/W	EADC Sample Module14 Control Register 1	0x0000_0000
EADC_M15CTL1	EADC_BA+0x17C	R/W	EADC Sample Module15 Control Register 1	0x0000_0000
EADC_M16CTL1	EADC_BA+0x180	R/W	EADC Sample Module16 Control Register 1	0x0000_0000
EADC_M17CTL1	EADC_BA+0x184	R/W	EADC Sample Module17 Control Register 1	0x0000_0000

Register	Offset	R/W	Description	Reset Value
EADC_M18CTL1	EADC_BA+0x188	R/W	EADC Sample Module18 Control Register 1	0x0000_0000
EADC_M19CTL1	EADC_BA+0x18C	R/W	EADC Sample Module19 Control Register 1	0x0000_0000
EADC_M20CTL1	EADC_BA+0x190	R/W	EADC Sample Module20 Control Register 1	0x0000_0000
EADC_M21CTL1	EADC_BA+0x194	R/W	EADC Sample Module21 Control Register 1	0x0000_0000
EADC_M22CTL1	EADC_BA+0x198	R/W	EADC Sample Module22 Control Register 1	0x0000_0000
EADC_M23CTL1	EADC_BA+0x19C	R/W	EADC Sample Module23 Control Register 1	0x0000_0000
EADC_DAT19	EADC_BA+0x200	R	EADC Data Register 19 for Sample Module 19	0x0000_0000
EADC_DAT20	EADC_BA+0x204	R	EADC Data Register 20 for Sample Module 20	0x0000_0000
EADC_DAT21	EADC_BA+0x208	R	EADC Data Register 21 for Sample Module 21	0x0000_0000
EADC_DAT22	EADC_BA+0x20C	R	EADC Data Register 22 for Sample Module 22	0x0000_0000
EADC_DAT23	EADC_BA+0x210	R	EADC Data Register 23 for Sample Module 23	0x0000_0000
EADC_DAT24	EADC_BA+0x214	R	EADC Data Register 24 for Sample Module 24	0x0000_0000
EADC_DAT25	EADC_BA+0x218	R	EADC Data Register 25 for Sample Module 25	0x0000_0000
EADC_DAT26	EADC_BA+0x21C	R	EADC Data Register 26 for Sample Module 26	0x0000_0000
EADC_SCTL19	EADC_BA+0x220	R/W	EADC Sample Module 19 Control Register	0x0000_0000
EADC_SCTL20	EADC_BA+0x224	R/W	EADC Sample Module 20 Control Register	0x0000_0000
EADC_SCTL21	EADC_BA+0x228	R/W	EADC Sample Module 21 Control Register	0x0000_0000
EADC_SCTL22	EADC_BA+0x22C	R/W	EADC Sample Module 22 Control Register	0x0000_0000
EADC_SCTL23	EADC_BA+0x230	R/W	EADC Sample Module 23 Control Register	0x0000_0000
EADC_SCTL24	EADC_BA+0x234	R/W	EADC Sample Module 24 Control Register	0x0000_0000
EADC_SCTL25	EADC_BA+0x238	R/W	EADC Sample Module 25 Control Register	0x0000_0000
EADC_SCTL26	EADC_BA+0x23C	R/W	EADC Sample Module 26 Control Register	0x0000_0000
EADC_VREF	EADC_BA+0xFF8	R/W	EADC Reference Voltage Control Register	0x0000_0000

6.20.7 Register Description

EADC Data Registers (EADC_DAT0~ EADC_DAT18)

Register	Offset	R/W	Description	Reset Value
EADC_DAT0	EADC_BA+0x00	R	EADC Data Register 0 for Sample Module 0	0x0000_0000
EADC_DAT1	EADC_BA+0x04	R	EADC Data Register 1 for Sample Module 1	0x0000_0000
EADC_DAT2	EADC_BA+0x08	R	EADC Data Register 2 for Sample Module 2	0x0000_0000
EADC_DAT3	EADC_BA+0x0C	R	EADC Data Register 3 for Sample Module 3	0x0000_0000
EADC_DAT4	EADC_BA+0x10	R	EADC Data Register 4 for Sample Module 4	0x0000_0000
EADC_DAT5	EADC_BA+0x14	R	EADC Data Register 5 for Sample Module 5	0x0000_0000
EADC_DAT6	EADC_BA+0x18	R	EADC Data Register 6 for Sample Module 6	0x0000_0000
EADC_DAT7	EADC_BA+0x1C	R	EADC Data Register 7 for Sample Module 7	0x0000_0000
EADC_DAT8	EADC_BA+0x20	R	EADC Data Register 8 for Sample Module 8	0x0000_0000
EADC_DAT9	EADC_BA+0x24	R	EADC Data Register 9 for Sample Module 9	0x0000_0000
EADC_DAT10	EADC_BA+0x28	R	EADC Data Register 10 for Sample Module 10	0x0000_0000
EADC_DAT11	EADC_BA+0x2C	R	EADC Data Register 11 for Sample Module 11	0x0000_0000
EADC_DAT12	EADC_BA+0x30	R	EADC Data Register 12 for Sample Module 12	0x0000_0000
EADC_DAT13	EADC_BA+0x34	R	EADC Data Register 13 for Sample Module 13	0x0000_0000
EADC_DAT14	EADC_BA+0x38	R	EADC Data Register 14 for Sample Module 14	0x0000_0000
EADC_DAT15	EADC_BA+0x3C	R	EADC Data Register 15 for Sample Module 15	0x0000_0000
EADC_DAT16	EADC_BA+0x40	R	EADC Data Register 16 for Sample Module 16	0x0000_0000
EADC_DAT17	EADC_BA+0x44	R	EADC Data Register 17 for Sample Module 17	0x0000_0000
EADC_DAT18	EADC_BA+0x48	R	EADC Data Register 18 for Sample Module 18	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.
[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**	**EADC Conversion Result** This field contains 12 bits conversion result. The 12-bit EADC conversion result with unsigned format will be filled in RESULT[11:0] and zero will be filled in RESULT[15:12]. **Note:** When operating in oversampling mode, RESULT[15:0] can represent oversampling results.

EADC PDMA Current Transfer Data Register (EADC_CURDAT)

Register	Offset	R/W	Description	Reset Value
EADC_CURDAT	EADC_BA+0x4C	R	EADC PDMA Current Transfer Data Register	0x0000_0000

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

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:0]	**CURDAT**	**EADC PDMA Current Transfer Data (Read Only)** This register is a shadow register of EADC_DATn (n=0~26) for PDMA support.

EADC Control Register (EADC_CTL)

Register	Offset	R/W	Description	Reset Value
EADC_CTL	EADC_BA+0x50	R/W	EADC 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							DIFFEN
7	6	5	4	3	2	1	0
Reserved		ADCIEN3	ADCIEN2	ADCIEN1	ADCIEN0	ADCRST	ADCEN

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**DIFFEN**	**Differential Analog Input Mode Enable Bit** 0 = Single-end analog input mode. 1 = Differential analog input mode. **Note:** In the differential mode, the input channel pair must be configured to EADC_CH15, EADC_CH14
[7:6]	**Reserved**	Reserved.
[5]	**ADCIEN3**	**Specific Sample Module EADC ADINT3 Interrupt Enable Bit** The EADC converter generates a conversion end ADIF3 (EADC_STATUS2[3]) upon the end of specific sample module EADC conversion. If EADCIEN3 bit is set then conversion end interrupt request ADINT3 is generated. 0 = Specific sample module EADC ADINT3 interrupt function Disabled. 1 = Specific sample module EADC ADINT3 interrupt function Enabled.
[4]	**ADCIEN2**	**Specific Sample Module EADC ADINT2 Interrupt Enable Bit** The EADC converter generates a conversion end ADIF2 (EADC_STATUS2[2]) upon the end of specific sample module EADC conversion. If EADCIEN2 bit is set then conversion end interrupt request ADINT2 is generated. 0 = Specific sample module EADC ADINT2 interrupt function Disabled. 1 = Specific sample module EADC ADINT2 interrupt function Enabled.
[3]	**ADCIEN1**	**Specific Sample Module EADC ADINT1 Interrupt Enable Bit** The EADC converter generates a conversion end ADIF1 (EADC_STATUS2[1]) upon the end of specific sample module EADC conversion. If EADCIEN1 bit is set then conversion end interrupt request ADINT1 is generated. 0 = Specific sample module EADC ADINT1 interrupt function Disabled. 1 = Specific sample module EADC ADINT1 interrupt function Enabled.

Bits	Description	
[2]	**ADCIEN0**	**Specific Sample Module EADC ADINT0 Interrupt Enable Bit** The EADC converter generates a conversion end ADIF0 (EADC_STATUS2[0]) upon the end of specific sample module EADC conversion. If ADCIEN0 bit is set then conversion end interrupt request ADINT0 is generated. 0 = Specific sample module EADC ADINT0 interrupt function Disabled. 1 = Specific sample module EADC ADINT0 interrupt function Enabled.
[1]	**ADCRST**	**EADC Converter Control Circuits Reset** 0 = No effect. 1 = Cause EADC control circuits reset to initial state, but not change the EADC registers value. **Note: EADC**RST bit remains 1 during EADC reset, when EADC reset end, the EADCRST bit is automatically cleared to 0.
[0]	**ADCEN**	**EADC Converter Enable Bit** 0 = EADC Disabled. 1 = EADC Enabled. **Note:** Before starting EADC conversion function, this bit should be set to 1. Clear it to 0 to disable EADC converter analog circuit power consumption.

EADC Sample Module Software Start Register (EADC_SWTRG)

Register	Offset	R/W	Description	Reset Value
EADC_SWTRG	EADC_BA+0x54	W	EADC Sample Module Software Start Register	0x0000_0000

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

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:0]	**SWTRG**	**EADC Sample Module 0~26 Software Force to Start EADC Conversion** 0 = No effect. 1 = Cause an EADC conversion when the priority is given to sample module. **Note:** After writing this register to start EADC 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.

EADC Sample Module Start of Conversion Pending Flag Register (EADC_PENDSTS)

Register	Offset	R/W	Description	Reset Value
EADC_PENDSTS	EADC_BA+0x58	R/W	EADC Start of Conversion Pending Flag Register	0x0000_0000

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

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:0]	**STPF**	**EADC Sample Module 0~26 Start of Conversion Pending Flag** Read Operation: 0 = There is no pending conversion for sample module. 1 = Sample module EADC start of conversion is pending. Write Operation: 1 = Clear pending flag & cancel the conversion for sample module. **Note:** This bit remains 1 during pending state, when the respective EADC conversion is end, the STPFn (n=0~26) bit is automatically cleared to 0.

EADC Sample Module Overrun Flag Register (EADC_OVSTS)

Register	Offset	R/W	Description	Reset Value
EADC_OVSTS	EADC_BA+0x5C	R/W	EADC Sample Module Start of Conversion Overrun Flag Register	0x0000_0000

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

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:0]	**SPOVF**	**EADC SAMPLE0~26 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.

EADC Control1 Register (EADC_CTL1)

Register	Offset	R/W	Description	Reset Value
EADC_CTL1	EADC_BA+0x60	R/W	EADC Control1 Register	0x0000_0000

31	30	29	28	27	26	25	24
OSR							
23	**22**	**23**	**22**	**19**	**18**	**17**	**16**
Reserved							**DECSET**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ULPDIV			**ULPEN**	**Reserved**			**FDETCHEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**DISCHEN**	**PRECHEN**

Bits	Description	
[31:24]	**OSR**	**Repeat Conversion Times Select** 8'b00000000 = ADC converts for 1 time 8'b00000001 = ADC converts for 2 times 8'b00000010 = ADC converts for 3 times 8'b00000011 = ADC converts for 4 times 8'b00000100 = ADC converts for 5 times : : : 8'b11111101 = ADC converts for 254 times 8'b11111110 = ADC converts for 255 times **Note:** The other steps of selection not listed above follow the same rule.
[23:17]	**Reserved**	Reserved.
[16]	**DECSET**	**High Speed Oversampling Mode Enable Bit** 0 = High speed oversampling mode Disabled. 1 = High speed oversampling mode Enabled.
[15:13]	**ULPDIV**	**Ultra Low Power Mode Prescalar selection** 000= ADC_CLK divided by 1 001= ADC_CLK divided by 2 010= ADC_CLK divided by 4 011= ADC_CLK divided by 8 100= ADC_CLK divided by 16 Others = Reserved
[12]	**ULPEN**	**Ultra Low Power Mode Enable Bit** 0 = Ultra low power mode Disabled. 1 = Ultra low power mode Enabled.
[11:9]	**Reserved**	Reserved.

[8]	**FDETCHEN**	**Floating Detect Channel Enable Bit** 0 = Floating Detect Channel Disabled. 1 = Floating Detect Channel Enabled. **Note:** if FDETCHEN is enabled, internal floating detect channel is always turn on.
[7:2]	**Reserved**	Reserved.
[1]	**DISCHEN**	**Discharge Enable** 0 = Channel discharge Disabled. 1 = Channel discharge Enabled. **Note:** Analog input voltage is 1/2 V_{REF} when PRECHEN and DISCHEN are all enabled.
[0]	**PRECHEN**	**Precharge Enable** 0 = Channel precharge Disabled. 1 = Channel precharge Enabled. **Note:** Analog input voltage is 1/2 V_{REF} when PRECHEN and DISCHEN are all enabled.

EADC Sample Module 0~3 Control Registers (EADC_SCTL0~EADC_SCTL3)

Register	Offset	R/W	Description	Reset Value
EADC_SCTL0	EADC_BA+0x80	R/W	EADC Sample Module 0 Control Register	0x0000_0000
EADC_SCTL1	EADC_BA+0x84	R/W	EADC Sample Module 1 Control Register	0x0000_0000
EADC_SCTL2	EADC_BA+0x88	R/W	EADC Sample Module 2 Control Register	0x0000_0000
EADC_SCTL3	EADC_BA+0x8C	R/W	EADC 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	EXTFEN	EXTREN	TRGSEL				
15	14	13	12	11	10	9	8
TRGDLYCNT							
7	6	5	4	3	2	1	0
TRGDLYDIV		INTPOS	CHSEL				

Bits	Description	
[31:24]	**EXTSMPT**	**EADC Sampling Time Extend** When EADC convertes at high conversion rate, the sampling time of analog input voltage may not be enough if input channel loading is heavy, and user can extend EADC sampling time after trigger source is coming to get enough sampling time. The range of start delay time is from 0~255 EADC clock.
[23]	**DBMEN**	**Double Buffer Mode Enable Bit** 0 = Sample has one sample result register (default). 1 = Sample has two sample result registers.
[22]	**EXTFEN**	**EADC External Trigger Falling Edge Enable Bit** 0 = Falling edge Disabled when EADC selects EADC0_ST as trigger source. 1 = Falling edge Enabled when EADC selects EADC0_ST as trigger source.
[21]	**EXTREN**	**EADC External Trigger Rising Edge Enable Bit** 0 = Rising edge Disabled when EADC selects EADC0_ST as trigger source. 1 = Rising edge Enabled when EADC selects EADC0_ST as trigger source.

Bits	Description	
[20:16]	**TRGSEL**	**EADC Sample Module Start of Conversion Trigger Source Selection** 0H = Disable trigger. 1H = External trigger from EADC0_ST pin input. 2H = EADC ADINT0 interrupt EOC (End of conversion) pulse trigger. 3H = EADC 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 = EPWM0TG0. 9H = EPWM0TG1. AH = EPWM0TG2. BH = EPWM0TG3. CH = EPWM0TG4. DH = EPWM0TG5. EH = EPWM1TG0. FH = EPWM1TG1. 10H = EPWM1TG2. 11H = EPWM1TG3. 12H = EPWM1TG4. 13H = EPWM1TG5. 14H = BPWM0TG. 15H = BPWM1TG. other = Reserved.
[15:8]	**TRGDLYCNT**	**EADC Sample Module Start of Conversion Trigger Delay Time** Trigger delay time = TRGDLYCNT x EADC_CLK period x n (n=1,2,4,16 from TRGDLYDIV setting).
[7:6]	**TRGDLYDIV**	**EADC Sample Module Start of Conversion Trigger Delay Clock Divider Selection** Trigger delay clock frequency: 00 = EADC_CLK/1. 01 = EADC_CLK/2. 10 = EADC_CLK/4. 11 = EADC_CLK/16.
[5]	**INTPOS**	**Interrupt Flag Position Select** 0 = Set ADIFn (EADC_STATUS2[n], n=0~3) at EADC end of conversion. 1 = Set ADIFn (EADC_STATUS2[n], n=0~3) at EADC start of conversion.

Bits	Description	
[4:0]	**CHSEL**	**EADC Sample Module Channel Selection** 00H = EADC_CH0. 01H = EADC_CH1. 02H = EADC_CH2. 03H = EADC_CH3. 04H = EADC_CH4. 05H = EADC_CH5. 06H = EADC_CH6. 07H = EADC_CH7. 08H = EADC_CH8. 09H = EADC_CH9. 0AH = EADC_CH10. 0BH = EADC_CH11. 0CH = EADC_CH12. 0DH = EADC_CH13. 0EH = EADC_CH14. 0FH = EADC_CH15. 10H = EADC_CH16. 11H = EADC_CH17. 12H = EADC_CH18. 13H = EADC_CH19. 14H = EADC_CH20. 15H = EADC_CH21. 16H = EADC_CH22. 17H = EADC_CH23.

EADC Sample Module 4~18 Control Registers (EADC_SCTL4~EADC_SCTL18)

Register	Offset	R/W	Description	Reset Value
EADC_SCTL4	EADC_BA+0x90	R/W	EADC Sample Module 4 Control Register	0x0000_0000
EADC_SCTL5	EADC_BA+0x94	R/W	EADC Sample Module 5 Control Register	0x0000_0000
EADC_SCTL6	EADC_BA+0x98	R/W	EADC Sample Module 6 Control Register	0x0000_0000
EADC_SCTL7	EADC_BA+0x9C	R/W	EADC Sample Module 7 Control Register	0x0000_0000
EADC_SCTL8	EADC_BA+0xA0	R/W	EADC Sample Module 8 Control Register	0x0000_0000
EADC_SCTL9	EADC_BA+0xA4	R/W	EADC Sample Module 9 Control Register	0x0000_0000
EADC_SCTL10	EADC_BA+0xA8	R/W	EADC Sample Module 10 Control Register	0x0000_0000
EADC_SCTL11	EADC_BA+0xAC	R/W	EADC Sample Module 11 Control Register	0x0000_0000
EADC_SCTL12	EADC_BA+0xB0	R/W	EADC Sample Module 12 Control Register	0x0000_0000
EADC_SCTL13	EADC_BA+0xB4	R/W	EADC Sample Module 13 Control Register	0x0000_0000
EADC_SCTL14	EADC_BA+0xB8	R/W	EADC Sample Module 14 Control Register	0x0000_0000
EADC_SCTL15	EADC_BA+0xBC	R/W	EADC Sample Module 15 Control Register	0x0000_0000
EADC_SCTL16	EADC_BA+0xC0	R/W	EADC Sample Module 16 Control Register	0x0000_0000
EADC_SCTL17	EADC_BA+0xC4	R/W	EADC Sample Module 17 Control Register	0x0000_0000
EADC_SCTL18	EADC_BA+0xC8	R/W	EADC Sample Module 18 Control Register	0x0000_0000

31	30	29	28	27	26	25	24
EXTSMPT							
23	22	21	20	19	18	17	16
Reserved	EXTFEN	EXTREN	TRGSEL				
15	14	13	12	11	10	9	8
TRGDLYCNT							
7	6	5	4	3	2	1	0
TRGDLYDIV		INTPOS	CHSEL				

Bits	Description	
[31:24]	**EXTSMPT**	**EADC Sampling Time Extend** When EADC is converting at high conversion rate, the sampling time of analog input voltage may not be enough if input channel loading is heavy, and software can extend EADC sampling time after trigger source is coming to get enough sampling time. The range of start delay time is from 0~255 EADC clock.
[23]	**Reserved**	Reserved.

Bits	Description	
[22]	**EXTFEN**	**EADC External Trigger Falling Edge Enable Bit** 0 = Falling edge Disabled when EADC selects EADC0_ST as trigger source. 1 = Falling edge Enabled when EADC selects EADC0_ST as trigger source.
[21]	**EXTREN**	**EADC External Trigger Rising Edge Enable Bit** 0 = Rising edge Disabled when EADC selects EADC0_ST as trigger source. 1 = Rising edge Enabled when EADC selects EADC0_ST as trigger source.
[20:16]	**TRGSEL**	**EADC Sample Module Start of Conversion Trigger Source Selection** 0H = Disable trigger. 1H = External trigger from EADC0_ST pin input. 2H = EADC ADINT0 interrupt EOC pulse trigger. 3H = EADC 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 = EPWM0TG0. 9H = EPWM0TG1. AH = EPWM0TG2. BH = EPWM0TG3. CH = EPWM0TG4. DH = EPWM0TG5. EH = EPWM1TG0. FH = EPWM1TG1. 10H = EPWM1TG2. 11H = EPWM1TG3. 12H = EPWM1TG4. 13H = EPWM1TG5. 14H = BPWM0TG. 15H = BPWM1TG. other = Reserved.
[15:8]	**TRGDLYCNT**	**EADC Sample Module Start of Conversion Trigger Delay Time** Trigger delay time = TRGDLYCNT x EADC_CLK period x n (n=1,2,4,16 from TRGDLYDIV setting).
[7:6]	**TRGDLYDIV**	**EADC Sample Module Start of Conversion Trigger Delay Clock Divider Selection** Trigger delay clock frequency: 00 = EADC_CLK/1. 01 = EADC_CLK/2. 10 = EADC_CLK/4. 11 = EADC_CLK/16.
[5]	**INTPOS**	**Interrupt Flag Position Select** 0 = Set ADIFn (EADC_STATUS2[n], n=0~3) at EADC end of conversion. 1 = Set ADIFn (EADC_STATUS2[n], n=0~3) at EADC start of conversion.

Bits	Description	
[4:0]	**CHSEL**	**EADC Sample Module Channel Selection** 00H = EADC_CH0. 01H = EADC_CH1. 02H = EADC_CH2. 03H = EADC_CH3. 04H = EADC_CH4. 05H = EADC_CH5. 06H = EADC_CH6. 07H = EADC_CH7. 08H = EADC_CH8. 09H = EADC_CH9. 0AH = EADC_CH10. 0BH = EADC_CH11. 0CH = EADC_CH12. 0DH = EADC_CH13. 0EH = EADC_CH14. 0FH = EADC_CH15. 10H = EADC_CH16. 11H = EADC_CH17. 12H = EADC_CH18. 13H = EADC_CH19. 14H = EADC_CH20. 15H = EADC_CH21. 16H = EADC_CH22. 17H = EADC_CH23.

EADC Interrupt Source Enable Control Registers (EADC_INTSRC0~EADC_INTSRC3)

Register	Offset	R/W	Description	Reset Value
EADC_INTSRC0	EADC_BA+0xD0	R/W	EADC Interrupt 0 Source Enable Control Register.	0x0000_0000
EADC_INTSRC1	EADC_BA+0xD4	R/W	EADC Interrupt 1 Source Enable Control Register.	0x0000_0000
EADC_INTSRC2	EADC_BA+0xD8	R/W	EADC Interrupt 2 Source Enable Control Register.	0x0000_0000
EADC_INTSRC3	EADC_BA+0xDC	R/W	EADC Interrupt 3 Source Enable Control Register.	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					SPLIE26	SPLIE25	SPLIE24
23	22	21	20	19	18	17	16
SPLIE23	SPLIE22	SPLIE21	SPLIE20	SPLIE19	SPLIE18	SPLIE17	SPLIE16
15	14	13	12	11	10	9	8
SPLIE15	SPLIE14	SPLIE13	SPLIE12	SPLIE11	SPLIE10	SPLIE9	SPLIE8
7	6	5	4	3	2	1	0
SPLIE7	SPLIE6	SPLIE5	SPLIE4	SPLIE3	SPLIE2	SPLIE1	SPLIE0

Bits	Description	
[31:27]	**Reserved**	Reserved
[26]	**SPLIE26**	**Sample Module 26 Interrupt Enable Bit** 0 = Sample Module 26 interrupt Disabled. 1 = Sample Module 26 interrupt Enabled.
[25]	**SPLIE25**	**Sample Module 25 Interrupt Enable Bit** 0 = Sample Module 25 interrupt Disabled. 1 = Sample Module 25 interrupt Enabled.
[24]	**SPLIE24**	**Sample Module 24 Interrupt Enable Bit** 0 = Sample Module 24 interrupt Disabled. 1 = Sample Module 24 interrupt Enabled.
[23]	**SPLIE23**	**Sample Module 23 Interrupt Enable Bit** 0 = Sample Module 23 interrupt Disabled. 1 = Sample Module 23 interrupt Enabled.
[22]	**SPLIE22**	**Sample Module 22 Interrupt Enable Bit** 0 = Sample Module 22 interrupt Disabled. 1 = Sample Module 22 interrupt Enabled.
[21]	**SPLIE21**	**Sample Module 21 Interrupt Enable Bit** 0 = Sample Module 21 interrupt Disabled. 1 = Sample Module 21 interrupt Enabled.

Bits	Description	
[20]	**SPLIE20**	**Sample Module 20 Interrupt Enable Bit** 0 = Sample Module 20 interrupt Disabled. 1 = Sample Module 20 interrupt Enabled.
[19]	**SPLIE19**	**Sample Module 19 Interrupt Enable Bit** 0 = Sample Module 19 interrupt Disabled. 1 = Sample Module 19 interrupt Enabled.
[18]	**SPLIE18**	**Sample Module 18 Interrupt Enable Bit** 0 = Sample Module 18 interrupt Disabled. 1 = Sample Module 18 interrupt Enabled.
[17]	**SPLIE17**	**Sample Module 17 Interrupt Enable Bit** 0 = Sample Module 17 interrupt Disabled. 1 = Sample Module 17 interrupt Enabled.
[16]	**SPLIE16**	**Sample Module 16 Interrupt Enable Bit** 0 = Sample Module 16 interrupt Disabled. 1 = Sample Module 16 interrupt Enabled.
[15]	**SPLIE15**	**Sample Module 15 Interrupt Enable Bit** 0 = Sample Module 15 interrupt Disabled. 1 = Sample Module 15 interrupt Enabled.
[14]	**SPLIE14**	**Sample Module 14 Interrupt Enable Bit** 0 = Sample Module 14 interrupt Disabled. 1 = Sample Module 14 interrupt Enabled.
[13]	**SPLIE13**	**Sample Module 13 Interrupt Enable Bit** 0 = Sample Module 13 interrupt Disabled. 1 = Sample Module 13 interrupt Enabled.
[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.

EADC 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	EADC Result Compare Register 0	0x0000_0000
EADC_CMP1	EADC_BA+0xE4	R/W	EADC Result Compare Register 1	0x0000_0000
EADC_CMP2	EADC_BA+0xE8	R/W	EADC Result Compare Register 2	0x0000_0000
EADC_CMP3	EADC_BA+0xEC	R/W	EADC 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.
[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 = EADCMPF0 (EADC_STATUS2[4]) will be set when EADC_CMP0 compared condition matched. EADCMPF2 (EADC_STATUS2[6]) will be set when EADC_CMP2 compared condition matched 1 = EADCMPF0 (EADC_STATUS2[4]) will be set when both EADC_CMP0 and EADC_CMP1 compared condition matched. EADCMPF2 (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.
[11:8]	**CMPMCNT**	**Compare Match Count** When the specified EADC 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 EADCMPFn (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. 01101 = Sample Module 13 conversion result EADC_DAT13 is selected to be compared. 01110 = Sample Module 14 conversion result EADC_DAT14 is selected to be compared. 01111 = Sample Module 15 conversion result EADC_DAT15 is selected to be compared. 10000 = Sample Module 16 conversion result EADC_DAT16 is selected to be compared. 10001 = Sample Module 17 conversion result EADC_DAT17 is selected to be compared. 10010 = Sample Module 18 conversion result EADC_DAT18 is selected to be compared. 10011 = Sample Module 19 conversion result EADC_DAT19 is selected to be compared. 10100 = Sample Module 20 conversion result EADC_DAT20 is selected to be compared. 10101 = Sample Module 21 conversion result EADC_DAT21 is selected to be compared. 10110 = Sample Module 22 conversion result EADC_DAT22 is selected to be compared. 10111 = Sample Module 23 conversion result EADC_DAT23 is selected to be compared. 11000 = Sample Module 24 conversion result EADC_DAT24 is selected to be compared. 11001 = Sample Module 25 conversion result EADC_DAT25 is selected to be compared. 11010 = Sample Module 26 conversion result EADC_DAT26 is selected to be compared. Others = reserved
[2]	**CMPCOND**	**Compare Condition** 0= Set the compare condition as that when a 12-bit EADC 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 EADC 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**	**EADC 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), EADCMPFn (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**	**EADC 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.

EADC Status Register 0 (EADC_STATUS0)

Register	Offset	R/W	Description	Reset Value
EADC_STATUS0	EADC_BA+0xF0	R	EADC Status Register 0	0x0000_0000

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

Bits	Description	
[31:16]	OV	**EADC_DAT0~15 Overrun Flag** It is a mirror to OV bit in sample module EADC result data register EADC_DATn. (n=0~15).
[15:0]	VALID	**EADC_DAT0~15 Data Valid Flag** It is a mirror of VALID bit in sample module EADC result data register EADC_DATn. (n=0~15).

EADC Status Register 1 (EADC_STATUS1)

Register	Offset	R/W	Description	Reset Value
EADC_STATUS1	EADC_BA+0xF4	R	EADC Status Register 1	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	
[31:27]	**Reserved**	Reserved.
[26:16]	**OV**	**EADC_DAT16~26 Overrun Flag** It is a mirror to OV bit in sample module EADC result data register EADC_DATn. (n=16~26).
[15:11]	**Reserved**	Reserved.
[10:0]	**VALID**	**EADC_DAT16~26 Data Valid Flag** It is a mirror of VALID bit in sample module EADC result data register EADC_DATn. (n=16~26).

EADC Status Register 2 (EADC_STATUS2)

Register	Offset	R/W	Description	Reset Value
EADC_STATUS2	EADC_BA+0xF8	R/W	EADC Status Register 2	0x0019_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.
[27]	**AOV**	**for All Sample Module EADC Result Data Register Overrun Flags Check (Read Only)** n=0~26. 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 EADC Result Data Register EADC_DAT Data Valid Flag Check (Read Only)** n=0~26. 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 EADC Sample Module Start of Conversion Overrun Flags Check (Read Only)** n=0~26. 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 EADC Interrupt Flag Overrun Bits Check (Read Only)** n=0~3. 0 = None of ADINT interrupt flag ADOVIFn, n=0~3 is overwritten to 1. 1 = Any one of ADINT interrupt flag ADOVIFn, n=0~3 is overwritten to 1. **Note:** This bit will keep 1 when any ADOVIFn Flag is equal to 1.
[23]	**BUSY**	**Busy/Idle (Read Only)** 0 = EADC is in idle state. 1 = EADC is busy at conversion. **Note:** this flag will be high after 4*EADC_CLK cycles, when the trigger source is coming.
[22:21]	**Reserved**	Reserved.

Bits	Description	
[20:16]	**CHANNEL**	**Current Conversion Channel (Read Only)** This filed reflects EADC current conversion channel when BUSY=1. It is read only. 00H = EADC_CH0. 01H = EADC_CH1. 02H = EADC_CH2. 03H = EADC_CH3. 04H = EADC_CH4. 05H = EADC_CH5. 06H = EADC_CH6. 07H = EADC_CH7. 08H = EADC_CH8. 09H = EADC_CH9. 0AH = EADC_CH10. 0BH = EADC_CH11. 0CH = EADC_CH12. 0DH = EADC_CH13. 0EH = EADC_CH14. 0FH = EADC_CH15. 10H = EADC_CH16 11H = EADC_CH17 12H = EADC_CH18 13H = EADC_CH19 14H = EADC_CH20 15H = EADC_CH21. 16H = EADC_CH22 17H = EADC_CH23 18H = EADC_CH24 (V_{BG}) 19H = EADC_CH25 (V_{TEMP}) 1AH = EADC_CH26 (DAC0_OUT)
[15]	**ADCMPO3**	**EADC 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 to CMPDAT3 setting.
[14]	**ADCMPO2**	**EADC 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 to CMPDAT2 setting.
[13]	**ADCMPO1**	**EADC 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 to CMPDAT1 setting.

Bits	Description	
[12]	**ADCMPO0**	**EADC 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.
[11]	**ADOVIF3**	**EADC 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**	**EADC 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**	**EADC 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**	**EADC 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**	**EADC Compare 3 Flag** When the specific sample module EADC 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**	**EADC Compare 2 Flag** When the specific sample module EADC 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**	**EADC Compare 1 Flag** When the specific sample module EADC 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**	**EADC Compare 0 Flag** When the specific sample module EADC 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**	**EADC ADINT3 Interrupt Flag** 0 = No ADINT3 interrupt pulse received. 1 = ADINT3 interrupt pulse has been received. **Note 1:** This bit is cleared by writing 1 to it. **Note 2:**This bit indicates whether an EADC conversion of specific sample module has been completed
[2]	**ADIF2**	**EADC ADINT2 Interrupt Flag** 0 = No ADINT2 interrupt pulse received. 1 = ADINT2 interrupt pulse has been received. **Note 1:** This bit is cleared by writing 1 to it. **Note 2:**This bit indicates whether an EADC conversion of specific sample module has been completed
[1]	**ADIF1**	**EADC ADINT1 Interrupt Flag** 0 = No ADINT1 interrupt pulse received. 1 = ADINT1 interrupt pulse has been received. **Note 1:** This bit is cleared by writing 1 to it. **Note 2:**This bit indicates whether an EADC conversion of specific sample module has been completed
[0]	**ADIF0**	**EADC ADINT0 Interrupt Flag** 0 = No ADINT0 interrupt pulse received. 1 = ADINT0 interrupt pulse has been received. **Note 1:** This bit is cleared by writing 1 to it. **Note 2:**This bit indicates whether an EADC conversion of specific sample module has been completed

EADC Status Register 3 (EADC_STATUS3)

Register	Offset	R/W	Description	Reset Value
EADC_STATUS3	EADC_BA+0xFC	R	EADC 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.
[4:0]	**CURSPL**	**EADC Current Sample Module (Read Only)** This register shows the current EADC is controlled by which sample module control logic modules. If the EADC is Idle, the bit filed will set to 0x1F.

EADC Double Data Register n for Sample Module n (EADC_DDAT0~3)

Register	Offset	R/W	Description	Reset Value
EADC_DDAT0	EADC_BA+0x100	R	EADC Double Data Register 0 for Sample Module 0	0x0000_0000
EADC_DDAT1	EADC_BA+0x104	R	EADC Double Data Register 1 for Sample Module 1	0x0000_0000
EADC_DDAT2	EADC_BA+0x108	R	EADC Double Data Register 2 for Sample Module 2	0x0000_0000
EADC_DDAT3	EADC_BA+0x10C	R	EADC 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	
[31:18]	**Reserved**	Reserved.
[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 = Double Data in RESULT (EADC_DDATn[15:0], n=0~3) is recent conversion result. 1 = Double Data in RESULT (EADC_DDATn[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**	**EADC Conversion Results** This field contains 12 bits conversion results. The 12-bit EADC conversion result with unsigned format will be filled in RESULT [11:0] and zero will be filled in RESULT [15:12].

EADC Calibration Control Register (EADC_CALCTL)

Register	Offset	R/W	Description	Reset Value
EADC_CALCTL	EADC_BA+0x114	R/W	EADC Calibration Control Register	0x0000_0020

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

Bits	Description	
[31:24]	**CALWRDATA**	Calibration Write Data
[23:13]	**Reserved**	Reserved.
[12:8]	**CALADDR**	**Calibration Data Address**
[7:2]	**Reserved**	Reserved.
[1]	**CALIE**	**Calibration Interrupt Enable Bit** 0 = Calibration interrupt Disabled. 1= Calibration interrupt Enabled.
[0]	**CAL**	**Calibration Enable Bit** 0 = = Calibration Disabled. 1 = = Calibration Enabled. **Note:** This bit is hardware auto cleared when calibration is done

EADC Calibration Status Register (EADC_CALSR)

Register	Offset	R/W	Description	Reset Value
EADC_CALSR	EADC_BA+0x118	R/W	EADC Calibration Status Register	0x0000_0000

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

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**CALIF**	**Calibration Finish Interrupt Flag** If calibration is finished, this flag will be set to 1. It is cleared by writing 1 to it.
[15:0]	**Reserved**	Reserved.

EADC PDMA Control Register (EADC_PDMACTL)

Register	Offset	R/W	Description	Reset Value
EADC_PDMACTL	EADC_BA+0x130	R/W	EADC PDMA Control Register	0x0000_0000

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

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:0]	**PDMATEN**	**PDMA Transfer Enable Bit** When EADC conversion is completed, the converted data is loaded into EADC_DATn (n: 0 ~ 26) 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 setting this bit field to 1, user must set EADCIENn (EADC_CTL[5:2], n=0~3) = 0 to disable interrupt.

EADC Sample Module 0~23 Control Registers1 (EADC_M0CTL1~EADC_M23CTL1)

Register	Offset	R/W	Description	Reset Value
EADC_M0CTL1	EADC_BA+0x140	R/W	EADC Sample Module0 Control Register 1	0x0000_0000
EADC_M1CTL1	EADC_BA+0x144	R/W	EADC Sample Module1 Control Register 1	0x0000_0000
EADC_M2CTL1	EADC_BA+0x148	R/W	EADC Sample Module2 Control Register 1	0x0000_0000
EADC_M3CTL1	EADC_BA+0x14C	R/W	EADC Sample Module3 Control Register 1	0x0000_0000
EADC_M4CTL1	EADC_BA+0x150	R/W	EADC Sample Module4 Control Register 1	0x0000_0000
EADC_M5CTL1	EADC_BA+0x154	R/W	EADC Sample Module5 Control Register 1	0x0000_0000
EADC_M6CTL1	EADC_BA+0x158	R/W	EADC Sample Module6 Control Register 1	0x0000_0000
EADC_M7CTL1	EADC_BA+0x15C	R/W	EADC Sample Module7 Control Register 1	0x0000_0000
EADC_M8CTL1	EADC_BA+0x160	R/W	EADC Sample Module8 Control Register 1	0x0000_0000
EADC_M9CTL1	EADC_BA+0x164	R/W	EADC Sample Module9 Control Register 1	0x0000_0000
EADC_M10CTL1	EADC_BA+0x168	R/W	EADC Sample Module10 Control Register 1	0x0000_0000
EADC_M11CTL1	EADC_BA+0x16C	R/W	EADC Sample Module11 Control Register 1	0x0000_0000
EADC_M12CTL1	EADC_BA+0x170	R/W	EADC Sample Module12 Control Register 1	0x0000_0000
EADC_M13CTL1	EADC_BA+0x174	R/W	EADC Sample Module13 Control Register 1	0x0000_0000
EADC_M14CTL1	EADC_BA+0x178	R/W	EADC Sample Module14 Control Register 1	0x0000_0000
EADC_M15CTL1	EADC_BA+0x17C	R/W	EADC Sample Module15 Control Register 1	0x0000_0000
EADC_M16CTL1	EADC_BA+0x180	R/W	EADC Sample Module16 Control Register 1	0x0000_0000
EADC_M17CTL1	EADC_BA+0x184	R/W	EADC Sample Module17 Control Register 1	0x0000_0000
EADC_M18CTL1	EADC_BA+0x188	R/W	EADC Sample Module18 Control Register 1	0x0000_0000
EADC_M19CTL1	EADC_BA+0x18C	R/W	EADC Sample Module19 Control Register 1	0x0000_0000
EADC_M20CTL1	EADC_BA+0x190	R/W	EADC Sample Module20 Control Register 1	0x0000_0000
EADC_M21CTL1	EADC_BA+0x194	R/W	EADC Sample Module21 Control Register 1	0x0000_0000
EADC_M22CTL1	EADC_BA+0x198	R/W	EADC Sample Module22 Control Register 1	0x0000_0000
EADC_M23CTL1	EADC_BA+0x19C	R/W	EADC Sample Module23 Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							

15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
ACU				Reserved		AVG	ALIGN

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:4]	**ACU**	**Number of Accumulated Conversion Results Selection** 0000 = 1 conversion result will be accumulated. 0001 = 2 conversion result will be accumulated. 0010 = 4 conversion result will be accumulated. 0011 = 8 conversion result will be accumulated. 0100 = 16 conversion result will be accumulated. 0101 = 32 conversion result will be accumulated. 0110 = 64 conversion result will be accumulated. 0111 = 128 conversion result will be accumulated. 1000 = 256 conversion result will be accumulated. Others = Reserved.
[3:2]	**Reserved**	Reserved.
[1]	**AVG**	**Average Mode Selection** 0 = Conversion results will be stored in data register without averaging. 1 = Conversion results in data register will be averaged. **Note:** This bit needs to work with ACU (EADC_MnCTL1[7:4], n=0~23).
[0]	**ALIGN**	**Alignment Selection** 0 = The conversion result will be right aligned in data register. 1 = The conversion result will be left aligned in data register.

EADC_Data Registers (EADC_DAT19~ EADC_DAT26)

Register	Offset	R/W	Description	Reset Value
EADC_DAT19	EADC_BA+0x200	R	EADC Data Register 19 for Sample Module 19	0x0000_0000
EADC_DAT20	EADC_BA+0x204	R	EADC Data Register 20 for Sample Module 20	0x0000_0000
EADC_DAT21	EADC_BA+0x208	R	EADC Data Register 21 for Sample Module 21	0x0000_0000
EADC_DAT22	EADC_BA+0x20C	R	EADC Data Register 22 for Sample Module 22	0x0000_0000
EADC_DAT23	EADC_BA+0x210	R	EADC Data Register 23 for Sample Module 23	0x0000_0000
EADC_DAT24	EADC_BA+0x214	R	EADC Data Register 24 for Sample Module 24	0x0000_0000
EADC_DAT25	EADC_BA+0x218	R	EADC Data Register 25 for Sample Module 25	0x0000_0000
EADC_DAT26	EADC_BA+0x21C	R	EADC Data Register 26 for Sample Module 26	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.
[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**	**EADC Conversion Result** This field contains 12 bits conversion result. The 12-bit EADC conversion result with unsigned format will be filled in RESULT[11:0] and zero will be filled in RESULT[15:12].

EADC Sample Module 19~23 Control Registers (EADC_SCTL19~EADC_SCTL23)

Register	Offset	R/W	Description	Reset Value
EADC_SCTL19	EADC_BA+0x220	R/W	EADC Sample Module 19 Control Register	0x0000_0000
EADC_SCTL20	EADC_BA+0x224	R/W	EADC Sample Module 20 Control Register	0x0000_0000
EADC_SCTL21	EADC_BA+0x228	R/W	EADC Sample Module 21 Control Register	0x0000_0000
EADC_SCTL22	EADC_BA+0x22C	R/W	EADC Sample Module 22 Control Register	0x0000_0000
EADC_SCTL23	EADC_BA+0x230	R/W	EADC Sample Module 23 Control Register	0x0000_0000

31	30	29	28	27	26	25	24
EXTSMPT							
23	22	21	20	19	18	17	16
Reserved	EXTFEN	EXTREN	TRGSEL				
15	14	13	12	11	10	9	8
TRGDLYCNT							
7	6	5	4	3	2	1	0
TRGDLYDIV		INTPOS	CHSEL				

Bits	Description	
[31:24]	**EXTSMPT**	**EADC Sampling Time Extend** When EADC is converting at high conversion rate, the sampling time of analog input voltage may not be enough if input channel loading is heavy, and software can extend EADC sampling time after trigger source is coming to get enough sampling time. The range of start delay time is from 0~255 EADC clock.
[23]	**Reserved**	Reserved.
[22]	**EXTFEN**	**EADC External Trigger Falling Edge Enable Bit** 0 = Falling edge Disabled when EADC selects EADC0_ST as trigger source. 1 = Falling edge Enabled when EADC selects EADC0_ST as trigger source.
[21]	**EXTREN**	**EADC External Trigger Rising Edge Enable Bit** 0 = Rising edge Disabled when EADC selects EADC0_ST as trigger source. 1 = Rising edge Enabled when EADC selects EADC0_ST as trigger source.

Bits	Description	
[20:16]	**TRGSEL**	**EADC Sample Module Start of Conversion Trigger Source Selection** 0H = Disable trigger. 1H = External trigger from EADC0_ST pin input. 2H = EADC ADINT0 interrupt EOC pulse trigger. 3H = EADC 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 = EPWM0TG0. 9H = EPWM0TG1. AH = EPWM0TG2. BH = EPWM0TG3. CH = EPWM0TG4. DH = EPWM0TG5. EH = EPWM1TG0. FH = EPWM1TG1. 10H = EPWM1TG2. 11H = EPWM1TG3. 12H = EPWM1TG4. 13H = EPWM1TG5. 14H = BPWM0TG. 15H = BPWM1TG. other = Reserved.
[15:8]	**TRGDLYCNT**	**EADC Sample Module Start of Conversion Trigger Delay Time** Trigger delay time = TRGDLYCNT x EADC_CLK period x n (n=1,2,4,16 from TRGDLYDIV setting).
[7:6]	**TRGDLYDIV**	**EADC Sample Module Start of Conversion Trigger Delay Clock Divider Selection** Trigger delay clock frequency: 00 = EADC_CLK/1. 01 = EADC_CLK/2. 10 = EADC_CLK/4. 11 = EADC_CLK/16.
[5]	**INTPOS**	**Interrupt Flag Position Select** 0 = Set ADIFn (EADC_STATUS2[n], n=0~3) at EADC end of conversion. 1 = Set ADIFn (EADC_STATUS2[n], n=0~3) at EADC start of conversion.

Bits	Description	
[4:0]	**CHSEL**	**EADC Sample Module Channel Selection** 00H = EADC_CH0. 01H = EADC_CH1. 02H = EADC_CH2. 03H = EADC_CH3. 04H = EADC_CH4. 05H = EADC_CH5. 06H = EADC_CH6. 07H = EADC_CH7. 08H = EADC_CH8. 09H = EADC_CH9. 0AH = EADC_CH10. 0BH = EADC_CH11. 0CH = EADC_CH12. 0DH = EADC_CH13. 0EH = EADC_CH14. 0FH = EADC_CH15. 10H = EADC_CH16. 11H = EADC_CH17. 12H = EADC_CH18. 13H = EADC_CH19. 14H = EADC_CH20. 15H = EADC_CH21. 16H = EADC_CH22. 17H = EADC_CH23.

EADC Sample Module 24~26 Control Registers (EADC_SCTL24~EADC_SCTL26)

Register	Offset	R/W	Description	Reset Value
EADC_SCTL24	EADC_BA+0x234	R/W	EADC Sample Module 24 Control Register	0x0000_0000
EADC_SCTL25	EADC_BA+0x238	R/W	EADC Sample Module 25 Control Register	0x0000_0000
EADC_SCTL26	EADC_BA+0x23C	R/W	EADC Sample Module 26 Control Register	0x0000_0000

31	30	29	28	27	26	25	24
EXTSMPT							
23	**22**	**21**	**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	
[31:24]	**EXTSMPT**	**EADC Sampling Time Extend** When EADC is converting at high conversion rate, the sampling time of analog input voltage may not be enough if input channel loading is heavy, and software can extend EADC sampling time after trigger source is coming to get enough sampling time. The range of start delay time is from 0~255 EADC clock.
[23:0]	**Reserved**	Reserved.

EADC Reference Voltage Control Register (EADC_VREF)

Register	Offset	R/W	Description	Reset Value
EADC_VREF	EADC_BA+0xFF8	R/W	EADC Reference Voltage 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						**REFSEL**	**Reserved**

Bits	Description	
[31:2]	**Reserved**	**Reserved.**
[1]	**REFSEL**	**Positive Reference Voltage Source Selection** **0 =** Positive Reference Voltage Source from AV_{DD} 1 = Positive Reference Voltage Source from V_{REF} or INT_VREF
[0]	**Reserved**	**Reserved.**

6.21 Digital to Analog Converter (DAC)

6.21.1 Overview

The DAC module is a 8-bit, voltage output digital-to-analog converter. It can be configured to used in conjunction with the PDMA controller. The DAC integrates a voltage output buffer that can be used to reduce output impendence and drive external loads directly without having to add an external operational amplifier.

6.21.2 Features

- Analog output voltage range: 0~AV_{DD}.
- Supports 8-bit output mode.
- Rail to rail settle time 5us.
- Supports up to one 8-bit 200 KSPS voltage type DAC without buffer.
- Reference voltage from internal reference voltage (INT_VREF), V_{REF} pin.
- Supports software and hardware trigger, including Timer0~3, EPWM0, EPWM1, and external trigger pin to start DAC conversion.
- Supports PDMA mode.

6.21.3 Block Diagram

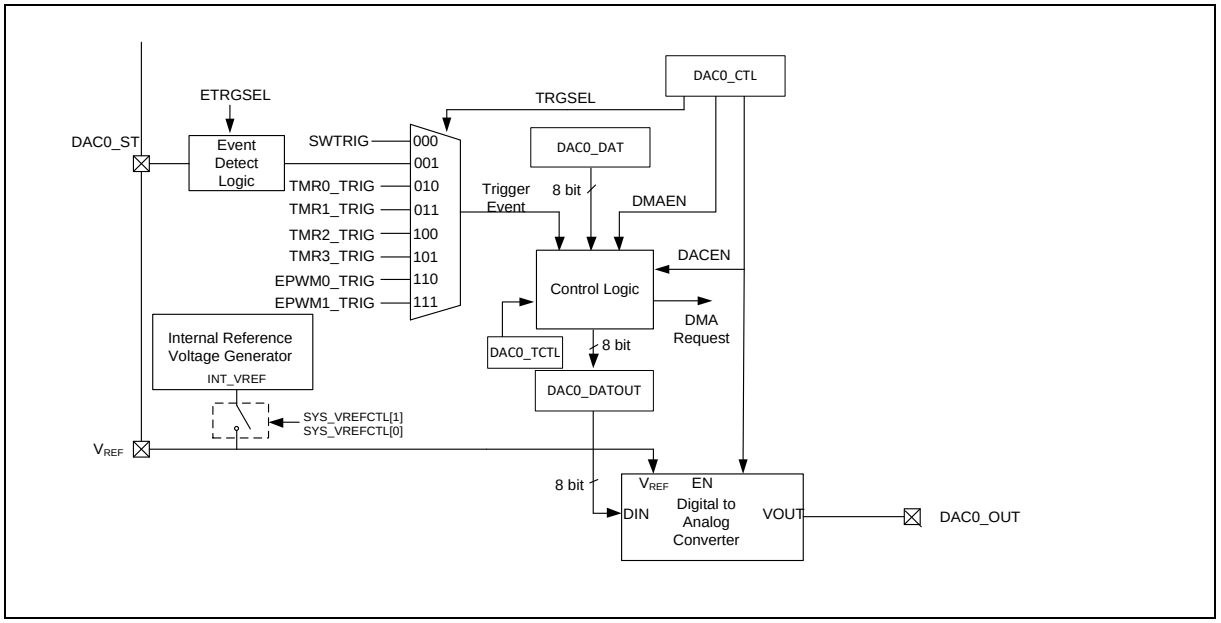


Figure 6.21-1 Digital-to-Analog Converter Block Diagram

6.21.4 Basic Configuration

6.21.4.1 DAC0 Basic Configuration

- Clock Source Configuration
  - Enable DAC0 peripheral clock in DACCKEN (CLK_APBCLK1[12]).
- Reset Configuration

- Reset DAC0 controller in DACRST (SYS_IPRST2[12]).

- Pin Configuration

Group	Pin Name	GPIO	MFP
DAC0	DAC0_OUT	PB.12	MFP1
	DAC0_ST	PA.10	MFP14
		PA.0	MFP15

6.21.5 Functional Description

6.21.5.1 DAC Output

The DAC is a 8-bit voltage output digital-to-analog converter and can be configured as 8-bit operation mode. The DAC integrates a voltage output buffer that can be used to reduce output impendence and drive external loads directly without having to add an external operational amplifier.The maximum DAC output voltage is limited to the selected reference voltage source.

6.21.5.2 DAC Reference Voltage

The DAC reference voltage is shared with EADC reference voltage and it is configured by VREFCTL (SYS_VREFCTL[4:0]) in system manager control registers. The reference voltage for the DAC can be configured from external reference voltage pin (V_{REF}) or internal reference voltage generator (INT_VREF).

6.21.5.3 DAC Conversion

Any data transfer to the DAC channel is performed by loading the data into DACn_DAT register. Figure 6.21-2 shows the DAC conversion started by software write operation. When user writes the conversion data to data holding register DACn_DAT, the data is loaded into data output register DACn_DATOUT by hardware and DAC starts data conversion after one PCLK (APB clock) clock cycle. Figure 6.21-3 shows the DAC conversion started by hardware trigger (external pin DACn_ST, timer trigger event or EPWM timer trigger event). The data stored in the DACn_DAT register is automatically transferred to the data output buffer DACn_DATOUT after occurring one PCLK (APB clock) the event.

When DAC data output register DACn_DATOUT is loaded with the DACn_DAT contents, the analog output voltage becomes available after specified conversion settling time. The conversion settling time is 5us when 8- bit input code transition from lowest code (0x00) to highest code (0xFF). Two adjacent codes conversion settling time is 1us. The DAC controller provides a 10-bit time counter for user to count the conversion time period. In continuous conversion operation, user needs to write appropriate value to SETTLET (DACn_TCTL[9:0]) to define DAC conversion time period. The value must be longer than DAC conversion settling time which is specified in DAC electric characteristic table. For example, when DAC controller APB clock speed is 80 MHz and DAC conversion settling time is 5us, the selected SETTLET value must be greater than 0x190. When the conversion is started, the conversion finish flag FINISH (DACn_STATUS[0]) is cleared to 0 by hardware and set to 1 after the time counter counts to SETTLET. Note that n=0,1.

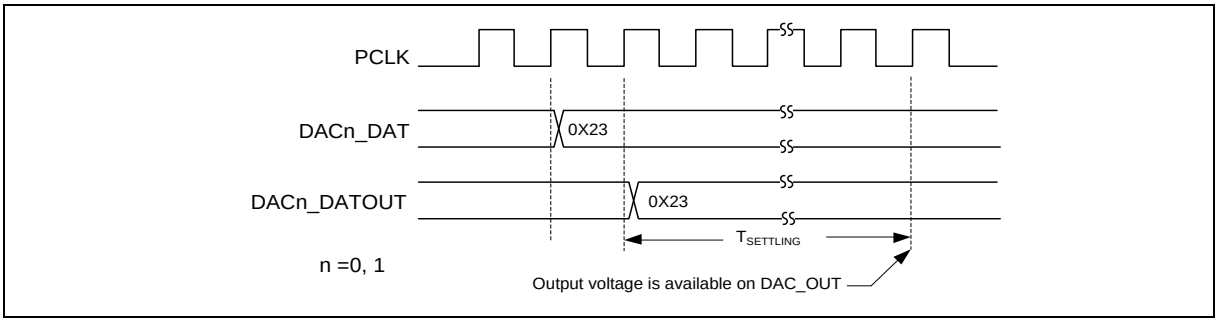


Figure 6.21-2 DAC Conversion Started by Software Write Trigger

6.21.5.4 DAC Output Voltage

Digital inputs are converted to output voltage on a linear conversion between 0 and reference voltage V_{REF}. The analog output voltage on DAC pin is determined by the following equation:

$DACOUT = V_{REF} * \frac{DATnOUT[7:0]}{256}$, n=0,1

6.21.5.5 DAC Trigger Selection

The DAC conversion can be started by writing DACn_DAT, software trigger or hardware trigger. When TRGEN (DACn_CTL[4]) is 0, the data conversion is started by writing DACn_DAT register. When TRGEN (DACn_CTL[4]) is 1, the data conversion is started by external DACn_ST pin, timer event, or EPWM timer event. If the software trigger is selected, the conversion starts once the SWTRG (DACn_SWTRG[0]) is set to 1. The SWTRG is cleared to 0 by hardware automatically when DACn_DATOUT has been loaded with DACDAT content. The TRGSEL (DACn_CTL[7:5]) determines which one of eight events is selected to start the conversion.

When DAC detects a rising edge on the selected trigger event input, the last data stored in DACDAT is transferred into the DACn_DATOUT[7:0] and DAC starts converting after one PCLK (APB clock) clock cycle. Note that n=0,1.

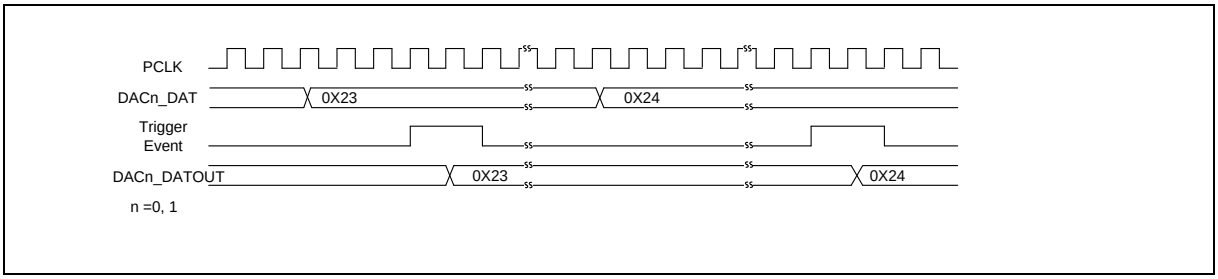


Figure 6.21-3 DAC Conversion Started by Hardware Trigger Event

6.21.5.6 DMA Operation

A DAC DMA request is generated when a hardware trigger event occurs while DMAEN (DACn_CTL[2]) is set. The content of DACn_DAT is transferred to the DACn_DATOUT[7:0] and DAC starts data conversion after one PCLK (APB clock) clock cycle. The new transferred data by PDMA in DACn_DAT will be converted when next trigger event arrives. Figure 6.21-4 shows the DAC PDMA under-run condition, when the second DMA request trigger event arrives before the first conversion finish, then no new PDMA request is issued and DMA under-run flag DMAUDR (DACn_STATUS[1]) is set 1 to report the error condition. DMA data transfers are then disabled and no further DMA request is treated and DAC continues to convert last data. An interrupt is also generated if the corresponding DMAURIEN (DACn_CTL[3]) is enabled. User has to change the trigger event frequency in timer or EPWM timer and then start DAC conversion again. Note that n=0,1.

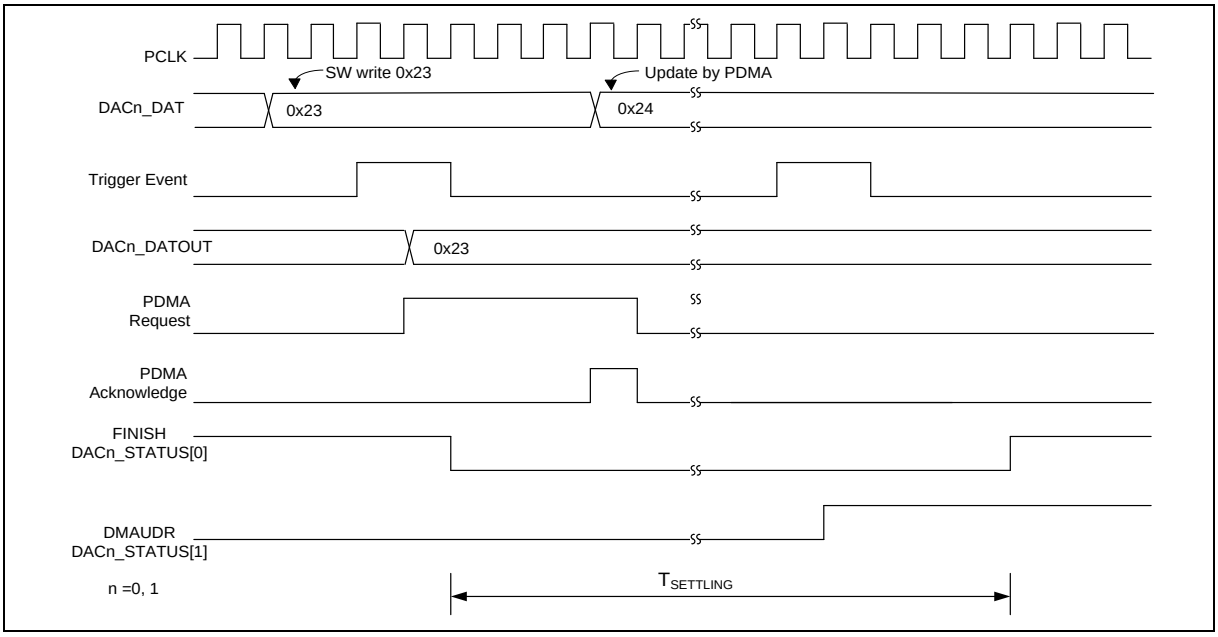


Figure 6.21-4 DAC PDMA Under-Run Condition Example

The DMA request can also be generated by software enable, user sets DMAEN (DACn_CTL[2]) to 1 and TRGEN (DACn_CTL[4]) to 0, DMA request is generated periodically according to the conversion time defined by SETTLET (DACn_TCTL[9:0]) value. DAC output is updated periodically. When user clears DMAEN (DACn_CTL[2]) to 0, DAC controller will stop issuing next new PDMA transfer request. Figure 6.21-5 provides an example of DAC continuous conversion with software PDMA mode. Note that n = 0,1.

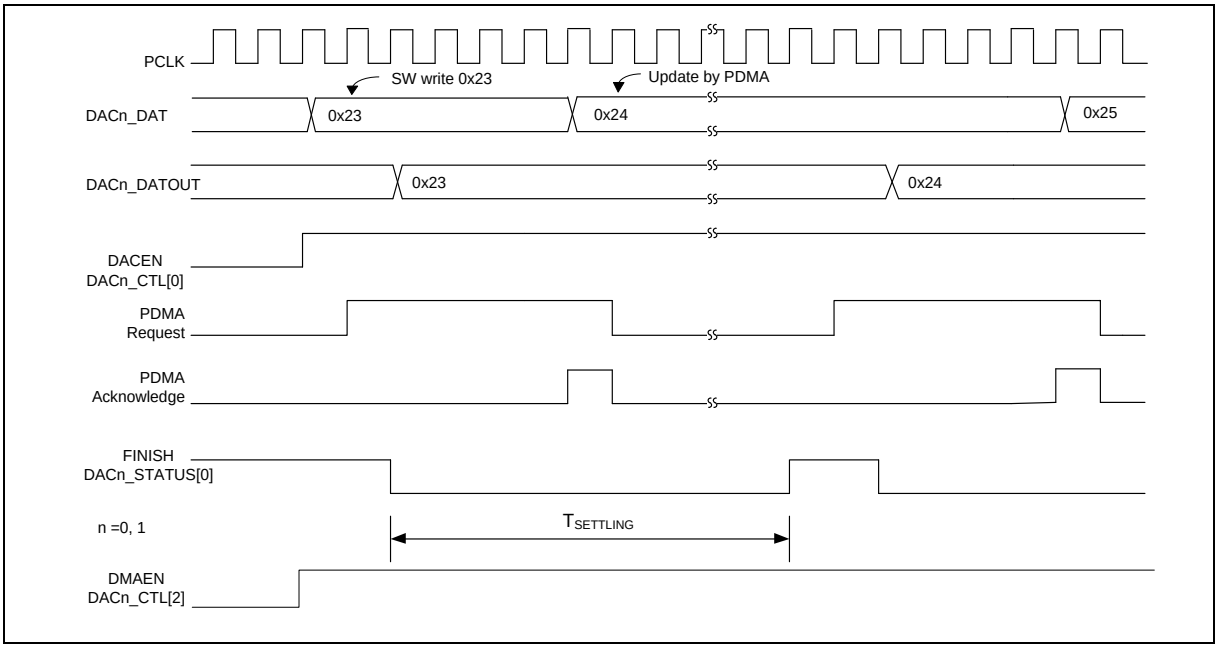


Figure 6.21-5 DAC Continuous Conversion with Software PDMA Mode

6.21.5.7 Interrupt Sources

There are two interrupt sources in the DAC controller, one is DAC data conversion finish interrupt and the other is DMA under-run interrupt as shown in Figure 6.21-6. When DAC conversion is finished, the

FINISH (DACn_STATUS[0]) is set to 1 and an interrupt occurs while DACIEN (DACn_CTL[1]) is enabled. If a new DMA trigger event occurs during DAC data conversion period, the DMA under-run flag DMAUDR (DACn_STATUS[1]) is generated and an interrupt occurs if DMAURIEN (DACn_CTL[3]) is enabled. Note that n = 0,1.

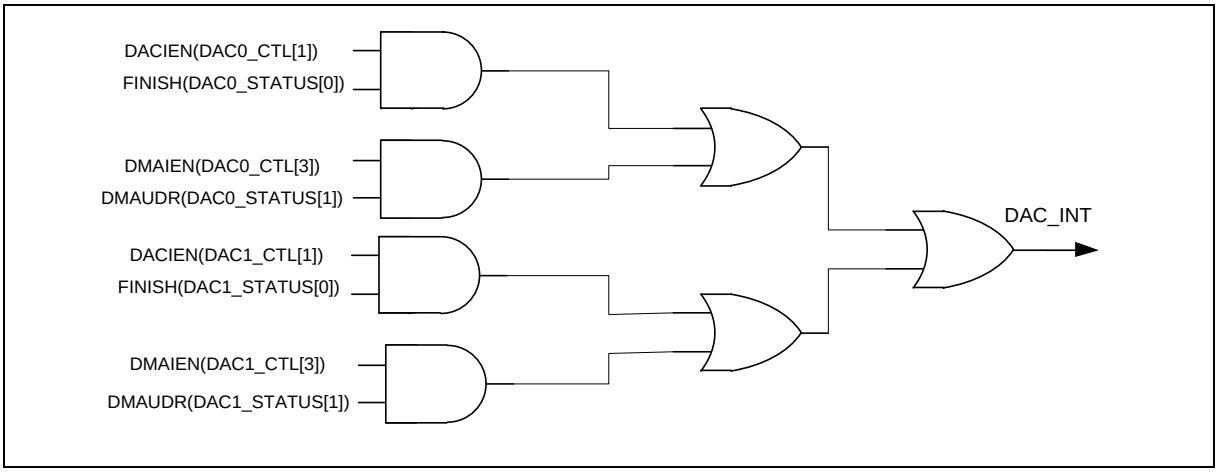


Figure 6.21-6 DAC Interrupt Source

6.21.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
DAC Base Address: **DAC_BA = 0x4004_7000**				
DAC0_CTL	DAC_BA+0x00	R/W	DAC0 Control Register	0x0000_0000
DAC0_SWTRG	DAC_BA+0x04	R/W	DAC0 Software Trigger Control Register	0x0000_0000
DAC0_DAT	DAC_BA+0x08	R/W	DAC0 Data Holding Register	0x0000_0000
DAC0_DATOUT	DAC_BA+0x0C	R	DAC0 Data Output Register	0x0000_0000
DAC0_STATUS	DAC_BA+0x10	R/W	DAC0 Status Register	0x0000_0000
DAC0_TCTL	DAC_BA+0x14	R/W	DAC0 Timing Control Register	0x0000_0000
DAC0_VREF	DAC_BA+0x20	R/W	DAC0 Reference Voltage Control Register	0x0000_0000

6.21.7 Register Description

DAC0 Control Register (DAC0_CTL)

Register	Offset	R/W	Description	Reset Value
DAC0_CTL	DAC_BA+0x00	R/W	DAC0 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		ETRGSEL		Reserved			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TRGSEL			TRGEN	DMAURIEN	DMAEN	DACIEN	DACEN

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13:12]	**ETRGSEL**	**External Pin Trigger Selection** 00 = Low level trigger. 01 = High level trigger. 10 = Falling edge trigger. 11 = Rising edge trigger.
[11:8]	**Reserved**	Reserved.
[7:5]	**TRGSEL**	**Trigger Source Selection** 000 = Software trigger. 001 = External pin DAC0_ST trigger. 010 = Timer 0 trigger. 011 = Timer 1 trigger. 100 = Timer 2 trigger. 101 = Timer 3 trigger. 110 = EPWM0 trigger. 111 = EPWM1 trigger.
[4]	**TRGEN**	**Trigger Mode Enable Bit** 0 = DAC event trigger mode Disabled. 1 = DAC event trigger mode Enabled.
[3]	**DMAURIEN**	**DMA Under-run Interrupt Enable Bit** 0 = DMA under-run interrupt Disabled. 1 = DMA under-run interrupt Enabled.
[2]	**DMAEN**	**DMA Mode Enable Bit** 0 = DMA mode Disabled. 1 = DMA mode Enabled.

[1]	**DACIEN**	**DAC Interrupt Enable Bit** 0 = DAC interrupt Disabled. 1 = DAC interrupt Enabled.
[0]	**DACEN**	**DAC Enable Bit** 0 = DAC Disabled. 1 = DAC Enabled.

DAC0 Software Trigger Control Register (DAC0_SWTRG)

Register	Offset	R/W	Description	Reset Value
DAC0_SWTRG	DAC_BA+0x04	R/W	DAC0 Software Trigger Control Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**SWTRG**	**Software Trigger** 0 = Software trigger Disabled. 1 = Software trigger Enabled. **Note:** User writes this bit to generate one shot pulse and this bit is cleared to 0 by hardware automatically; reading this bit will always get 0.

DAC0 Data Holding Register (DAC0_DAT)

Register	Offset	R/W	Description	Reset Value
DAC0_DAT	DAC_BA+0x08	R/W	DAC0 Data Holding Register	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**DACDAT**	**DAC 8-bit Holding Data** These bits are written by user software which specifies 8-bit conversion data for DAC output.

DAC0 Data Output Register (DAC0_DATOUT)

Register	Offset	R/W	Description	Reset Value
DAC0_DATOUT	DAC_BA+0x0C	R	DAC0 Data Output Register	0x0000_0000

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

Bits	Description	
[31:12]	Reserved	Reserved.
[11:0]	DATOUT	**DAC 8-bit Output Data** These bits are current digital data for DAC output conversion. It is loaded from DAC_DAT register and user cannot write it directly.

DAC0 Status Register (DAC0_STATUS)

Register	Offset	R/W	Description	Reset Value
DAC0_STATUS	DAC_BA+0x10	R/W	DAC0 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							BUSY
7	6	5	4	3	2	1	0
Reserved						DMAUDR	FINISH

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**BUSY**	**DAC Busy Flag (Read Only)** 0 = DAC is ready for next conversion. 1 = DAC is busy in conversion.
[7:2]	**Reserved**	Reserved.
[1]	**DMAUDR**	**DMA Under-run Interrupt Flag** 0 = No DMA under-run error condition occurred. 1 = DMA under-run error condition occurred. **Note:** User writes 1 to clear this bit.
[0]	**FINISH**	**DAC Conversion Complete Finish Flag** 0 = DAC is in conversion state. 1 = DAC conversion finish. **Note:** This bit is set to 1 when conversion time counter counts to SETTLET. It is cleared to 0 when DAC starts a new conversion. User writes 1 to clear this bit to 0.

DAC0 Timing Control Register (DAC0_TCTL)

Register	Offset	R/W	Description	Reset Value
DAC0_TCTL	DAC_BA+0x14	R/W	DAC0 Timing 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						SETTLET	
7	6	5	4	3	2	1	0
SETTLET							

Bits	Description	
[31:10]	Reserved	Reserved.
[9:0]	SETTLET	**DAC Output Settling Time** User software needs to write appropriate value to these bits to meet DAC conversion settling time base on PCLK (APB clock) speed. For example, DAC controller clock speed is 80 MHz and DAC conversion settling time is 1 us, SETTLETvalue must be greater than 0x50. SELTTLET = DAC controller clock speed x settling time.

DAC0 Reference Voltage Control Register (DAC0_VREF)

Register	Offset	R/W	Description	Reset Value
DAC0_VREF	DAC_BA+0x20	R/W	DAC0 Reference Voltage 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		**OUTFLOAT**	**$SELV_{REF}$**	**Reserved**			

Bits	Description	
[31:1]	**Reserved**	Reserved.
[5]	**OUTFLOAT**	**DAC Output Floating Selection** 0 = DAC_OUT output DAC_ROUT 1 = DAC_OUT output Hi-z
[4]	**$SELV_{REF}$**	**DAC Reference Voltage Selection** 0 = DAC reference voltage is from AV_{DD} 1 = DAC reference voltage is from VREFP
[3:0]	**Reserved**	Reserved.

6.22 Analog Comparator Controller (ACMP)

6.22.1 Overview

The chip provides two comparators. The comparator output is logic 1 when positive input is greater than negative input; otherwise, the output is 0. Each comparator can be configured to generate an interrupt when the comparator output value changes.

6.22.2 Features

- Analog input voltage range: 0 ~ AV_{DD} (voltage of AV_{DD} pin)
- Up to two rail-to-rail analog comparators
- Supports hysteresis function
  - Supports programmable hysteresis window: 0mV, 10mV, 20mV, 30mV, 40mV, 50mV
- Supports wake-up function
- Selectable input sources of positive input and negative input
- ACMP0 supports:
  - 4 multiplexed I/O pins at positive sources:
    - ACMP0_P0, ACMP0_P1, ACMP0_P2, or ACMP0_P3
  - 4 negative sources:
    - ACMP0_N
    - Supports Comparator Reference Voltage (CRV)
    - Internal band-gap voltage (VBG)
    - DAC0 output (DAC0_OUT)
- ACMP1 supports:
  - 4 multiplexed I/O pins at positive sources:
    - ACMP1_P0, ACMP1_P1, ACMP1_P2, or ACMP1_P3
  - 4 negative sources:
    - ACMP1_N
    - Supports Comparator Reference Voltage (CRV)
    - Internal band-gap voltage (VBG)
    - DAC0 output (DAC0_OUT)
- Shares one ACMP interrupt vector for all comparators
- Interrupts generated when compare results change (Interrupt event condition is programmable)
- Supports triggers for break events and cycle-by-cycle control for EPWM
- Supports window compare mode and window latch mode
- Supports offset calibration

6.22.3 Block Diagram

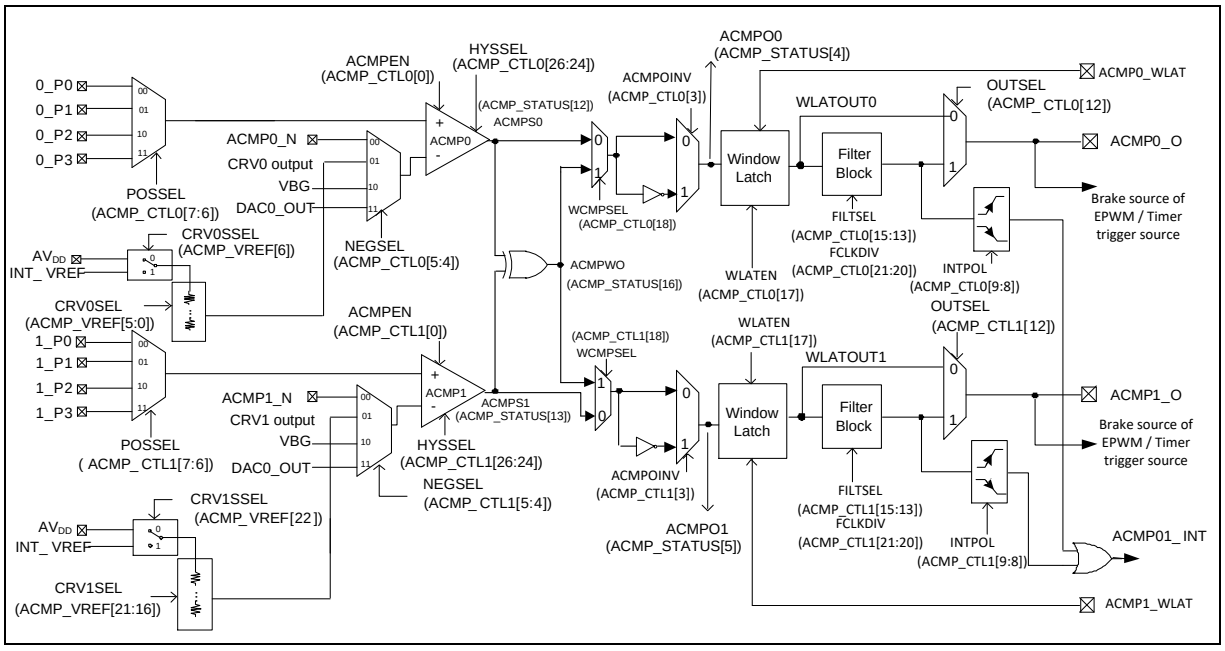


Figure 6.22-1 Analog Comparator Block Diagram

6.22.4 Basic Configuration

6.22.4.1 ACMP0 Basic Configuration

- Clock Source Configuration
  - Enable ACMP0 peripheral clock in ACMP01CKEN (CLK_APBCLK0[7]).
- Reset Configuration
  - Reset ACMP0 controller in ACMP01RST (SYS_IPRST1[7]).

6.22.4.2 ACMP1 Basic Configuration

- Clock Source Configuration
  - Enable ACMP1 peripheral clock in ACMP01CKEN (CLK_APBCLK0[7]).
- Reset Configuration
  - Reset ACMP1 controller in ACMP01RST (SYS_IPRST1[7]).

6.22.5 Functional Description

6.22.5.1 Hysteresis Function

The analog comparator provides the hysteresis function to make the comparator to have a stable output transition and it can refer to Figure 6.22-2. If comparator output is 0, it will not be changed to 1 until the positive input voltage exceeds the negative input voltage by a high threshold voltage. Similarly, if comparator output is 1, it will not be changed to 0 until the positive input voltage drops below the negative input voltage by a low threshold voltage.

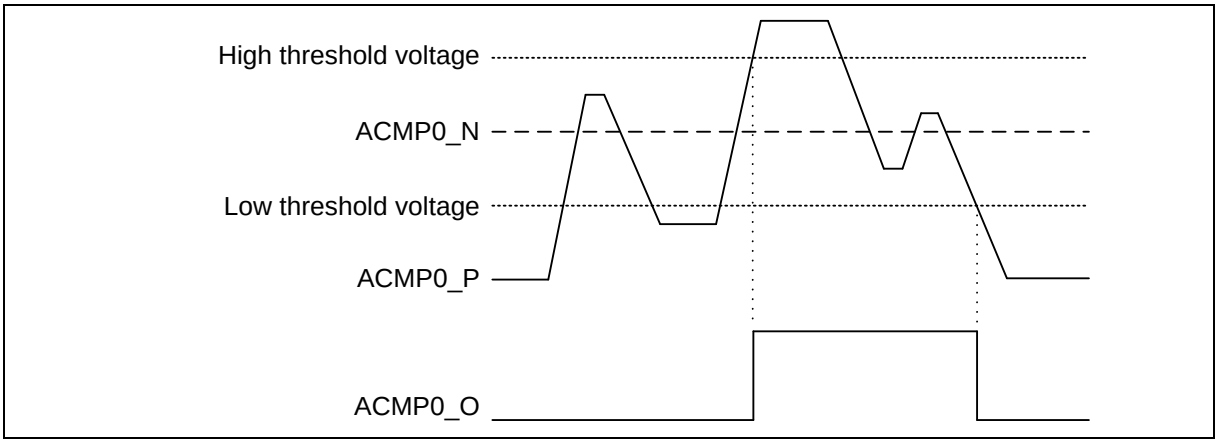


Figure 6.22-2 Comparator Hysteresis Function of ACMP0

6.22.5.2 Window Latch Mode

Figure 6.22-3 shows the comparator operation in window latch mode. Window latch mode can be enabled by setting WLATEN (ACMP_CTL0/1[17]) to 1. When window latch function enabled, ACMP0/1_WLAT pin is used to control the output WLATOUT0/1 .When ACMP0/1_WLAT pin is high, ACMPO0/1 passes through to WLATOUT0/1. When ACMP0/1_WLAT pin is low, WLATOUT0/1 will keep last state of WLATOUT0/1.

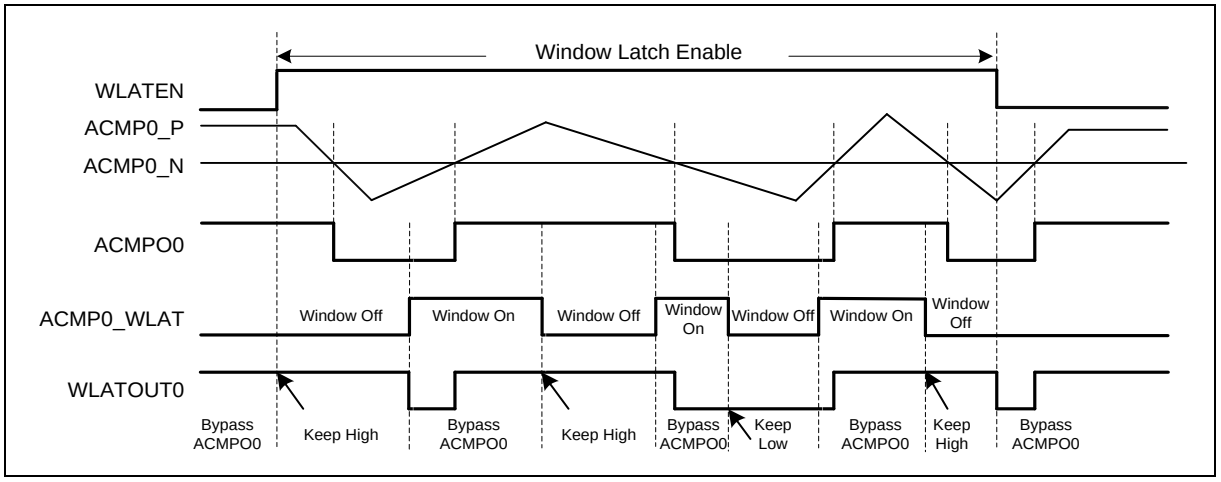


Figure 6.22-3 Window Latch Mode

6.22.5.3 Filter Function

The analog comparator provides filter function to avoid the un-stable state of comparator output.

By setting FILTSEL (ACMP_CTL0[15:13], ACMP_CTL1[15:13]), the comparator output would be sampled by consecutive PCLKs. With longer sample clocks, the comparator output would be more stable. But the sensitivity of comparator output would be reduced. Under the condition of setting FILTSEL to 3'h7 the comparator output is sampled 64 consecutive PCLKs, filter sample clock can be divided by 2 or 4 by setting FCLKDIV(ACMP_CTL0[21:20], ACMP_CTL1[21:20]) to get sampled 128, 256 consecutive PCLKs.

Figure 6.22-4 shows an example of filter function of ACMP0 with FILTSEL = 3 (4 PCLK). In this example, the comparing result is sampled by PCLK. All result must keep for 4 PCLK clocks before it

can be output to ACMPO0. If the comparing result is shorter than 4 PCLK, it will be filtered.

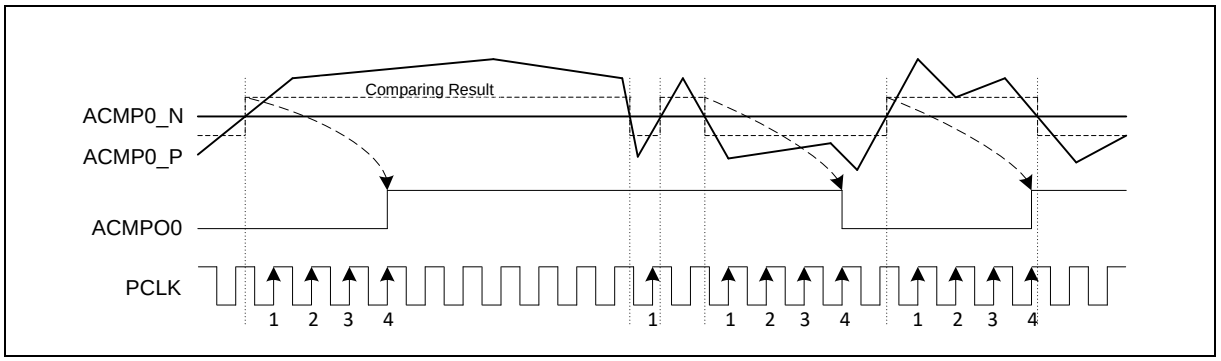


Figure 6.22-4 Example of Filter Function

6.22.5.4 Interrupt Sources

The outputs of ACMP0 and ACMP1 are reflected at ACMPO0 (ACMP_STATUS[4]) and ACMPO1 (ACMP_STATUS[5]) respectively. Then they are processed by window latch and filter functions. Finally, the output signal could be utilized to assert interrupts. Refer to Figure 6.22-5, if ACMPIE of ACMP_CTL0/1 register is set to 1, the interrupt will be enabled. If the output state ACMPO0/1 is changed as the setting of INTPOL (ACMP_CTL0/1[9:8]), the comparator interrupt will be asserted and the corresponding flag, ACMPIF0 (ACMP_STATUS[0]) and ACMPIF1 (ACMP_STATUS[1]), will be set to 1. The interrupt flag can be cleared to 0 by writing 1.

WKIF0/1(ACMP_STATUS[8], ACMP_STATUS[9]) will be set according to the setting of INTPOL (ACMP_CTL0/1[9:8]) if ACMP wakeup function is enabled. These two flags also cause interrupt rising that makes system wake-up from power down by ACMP. Figure 6.22-5 shows the interrupt sources of ACMP.

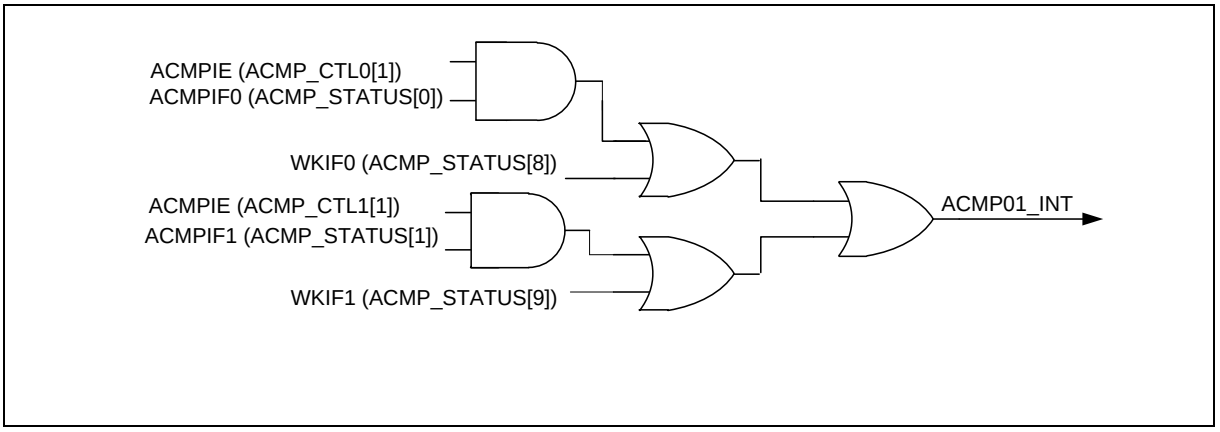


Figure 6.22-5 Comparator Controller Interrupt

6.22.5.5 Comparator Reference Voltage (CRV)

The comparator reference voltage (CRV) module is responsible for generating reference voltage for comparators. The CRV module consists of resistor ladder and analog switch. User can set the CRV output voltage by setting CRV0SEL (ACMP_VREF[5:0]), CRV1SEL (ACMP_VREF[21:16]). The CRV output voltage can be selected as the negative input of comparator by setting NEGSEL (ACMP_CTL0[5:4], ACMP_CTL1[5:4]). Figure 6.22-6 shows the block diagram of Comparator Reference Voltage.

The resistor ladder will be disabled by hardware to reduce power consumption when NEGSEL (ACMP_CTL0[5:4], ACMP_CTL1[5:4]) is not selected to CRV module. The reference voltage of resistor ladder can be the voltage of AV_{DD} pin or the INT_VREF voltage which is controlled by

SYS_VREFCTL register. Figure 6.22-6 shows comparator0 CRV0 output. For example, if user wants to get comparator1 CRV1 output, CRV1SSEL (ACMP_VREF[22]), CRV1SEL (ACMP_VREF[21:16]) must be set.

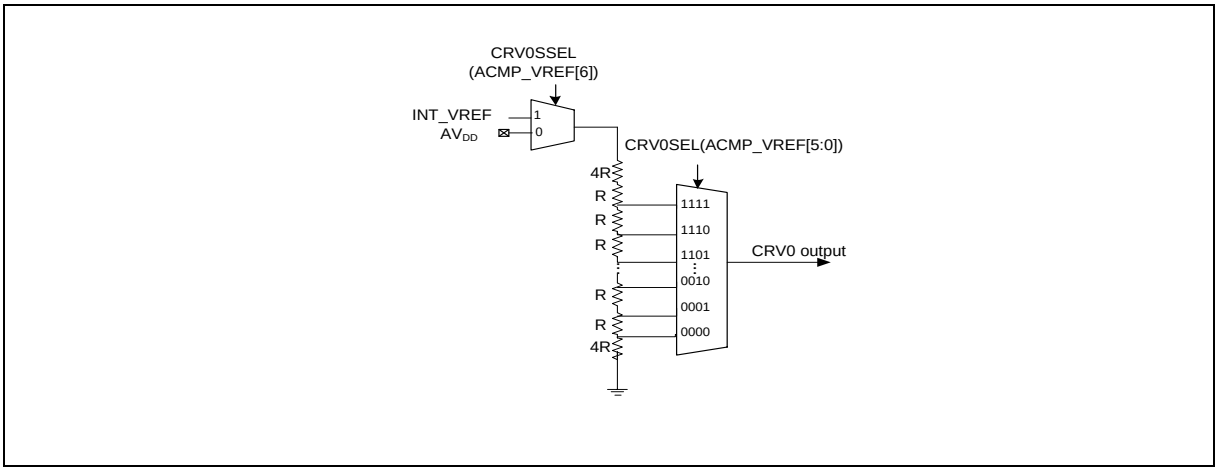


Figure 6.22-6 Comparator0 Reference Voltage Block Diagram

6.22.5.6 Window Compare Mode

The comparator provides window compare mode. When window compare mode is enabled by setting WCMPSEL (ACMP_CTL0/1[18]) to 1, user can monitor a specific analog voltage source with a designated range. User can connect the specific analog voltage source to either the positive inputs of both comparators or the negative inputs of both comparators. The upper bound and lower bound of the designated range are determined by the voltages applied to the other inputs of both comparators. If the output of a comparator is low and the other comparator outputs high, which means two comparators implies the upper and lower bound. User can directly monitor a specific analog voltage source via ACMPWO (ACMP_STATUS[16]). If ACMPWO is high, it implies a specific analog voltage source is in the range of upper and lower bound, which are called as the analog voltage is in the window.

Figure 6.22-7 illustrates an example of window compare mode. In this example, once window compare mode is selected, user can choose one of four positive input sources of each comparator and connect these two inputs together outside the chip.

If ACMPS0 outputs high and ACMPS1 outputs low, it means the voltage source is in the range of lower bound and upper bound, which are called as the voltage source in the window. Otherwise, the voltage source is outside the window.

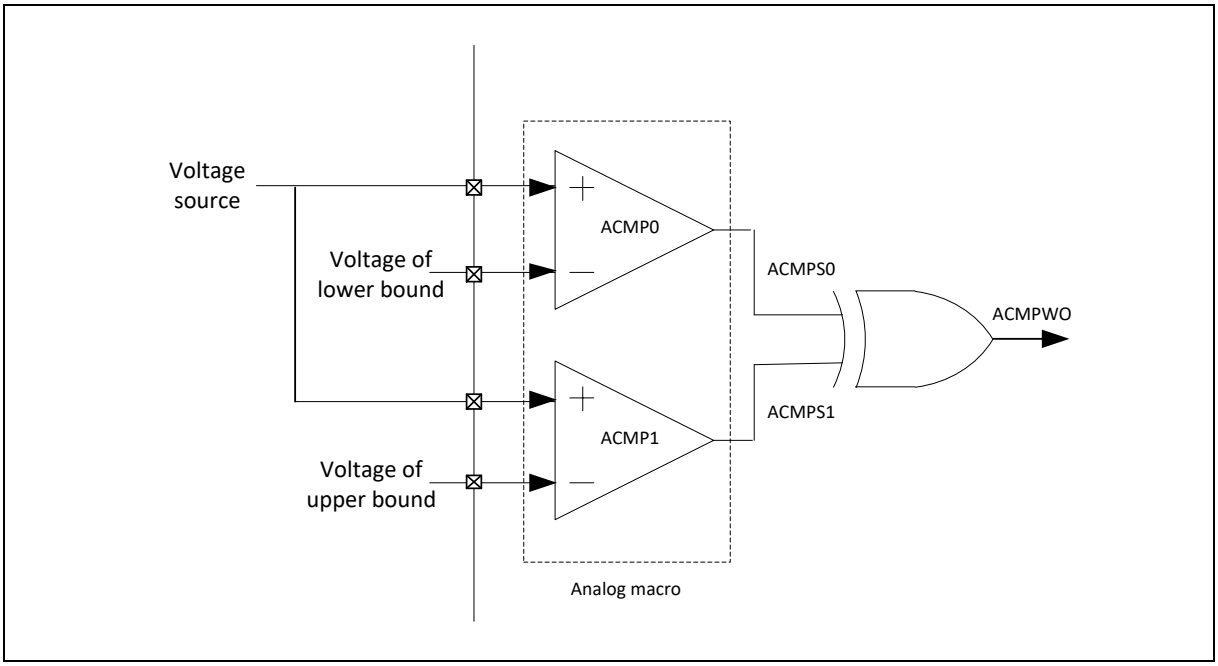


Figure 6.22-7 Example of Window Compare Mode

The comparator window output (ACMPWO) can be shown in ACMP_STATUS[16] and the truth table of window compare logic is shown in Table 6.22-1.

ACMPS0	ACMPS1	ACMPWO
0	0	0
0	1	1
1	1	0
1	0	1

Table 6.22-1 Truth Table of Window Compare Logic

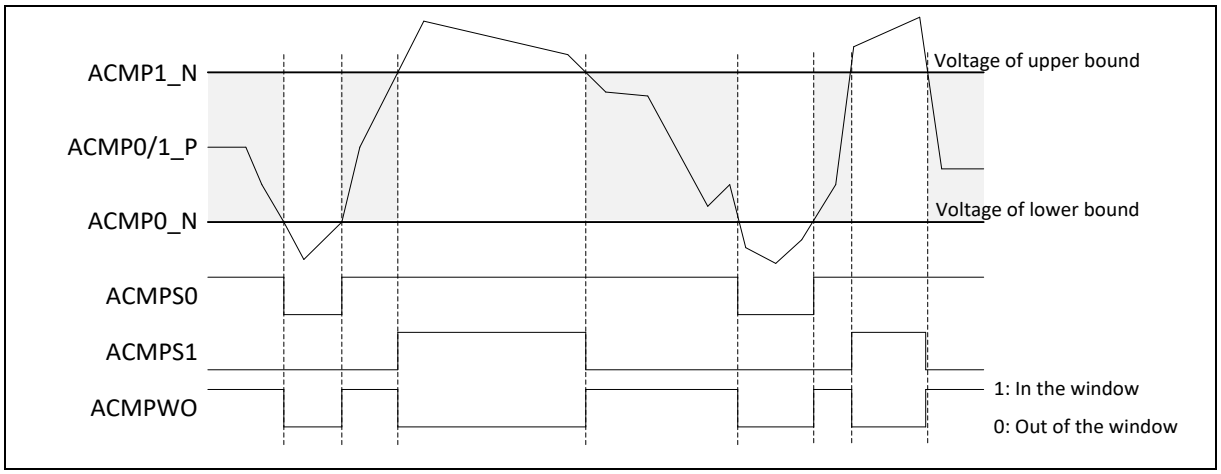


Figure 6.22-8 Example of Window Compare Mode

As shown in Figure 6.22-8, if ACMPWO equals 1, it means positive input voltage is inside the window. Otherwise, the positive input voltage is outside the window. Therefore, ACMPWO can be used to

monitor voltage transition of external analog pin. Furthermore, ACMPWO still can be applied to window latch, filter functions and interrupt of ACMP.

Note that negative inputs must choose different source. Otherwise, the function will be meaningless.

6.22.5.7 Calibration Function

The two comparators have its own four trim bits which can be used to calibrate the offset voltage. The offset voltage comes from both mismatch of NMOS-type differential and PMOS-type differential input stages. Calibration can be started by setting CALTRG0 (ACMP_CALCTL[0]), CALTRG1 (ACMP_CALCTL[1]).

The common mode input range is achieved by using an NMOS and a PMOS differential pairs connected in parallel. After calibration, reading the ACMP_CALSR register can monitor calibration status, including calibration done status, NMOS, and PMOS calibration result status. Take a brief calibration flow for example: Once ACMPEN (ACMP_CTLx[0], x=0,1) is enabled, hardware will automatically load default calibrated trim values to compensate offset voltage. Therefore, user can directly use comparator without doing additional calibration action. If user wants to start calibration function again, the newer calibrated trim value will be updated after calibration done. Noted that every time ACMPEN is set, user must set CALTRG0 (ACMP_CALCTL[0]) or CALTRG1 (ACMP_CALCTL[1]) and NEGSEL (ACMP_CTLx[5:4]), x= 0,1 equal to 2'b01 to start calibration function getting newer offset trim values, or comparator will operating with elder offset trim values.

6.22.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
ACMP Base Address: **ACMP01_BA = 0x4004_5000**				
ACMP_CTL0	ACMP01_BA+0x00	R/W	Analog Comparator 0 Control Register	0x0000_0000
ACMP_CTL1	ACMP01_BA+0x04	R/W	Analog Comparator 1 Control Register	0x0000_0000
ACMP_STATUS	ACMP01_BA+0x08	R/W	Analog Comparator Status Register	0x0000_0000
ACMP_VREF	ACMP01_BA+0x0C	R/W	Analog Comparator Reference Voltage Control Register	0x0000_0000
ACMP_CALCTL	ACMP01_BA+0x10	R/W	Analog Comparator Calibration Control Register	0x0000_0050
ACMP_CALSR	ACMP01_BA+0x14	R	Analog Comparator Calibration Status Register	0x0000_0000

6.22.7 Register Description

Analog Comparator 0 Control Register (ACMP_CTL0)

Register	Offset	R/W	Description	Reset Value
ACMP_CTL0	ACMP01_BA+0x00	R/W	Analog Comparator 0 Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					HYSSEL		
23	22	21	20	19	18	17	16
Reserved		FCLKDIV		Reserved	WCMPSEL	WLATEN	WKEN
15	14	13	12	11	10	9	8
FILTSEL			OUTSEL	Reserved		INTPOL	
7	6	5	4	3	2	1	0
POSSEL		NEGSEL		ACMPOINV	Reserved	ACMPIE	ACMPEN

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:24]	**HYSSEL**	**Hysteresis Mode Selection** 000 = Hysteresis is 0mV. 010 = Hysteresis is 20mV. 100 = Hysteresis is 40mV Others = Reserved
[23:22]	**Reserved**	Reserved.
[21:20]	**FCLKDIV**	**Comparator Output Filter Clock Divider** 00 = Comparator output filter clock = PCLK 01 = Comparator output filter clock = PCLK/2 10 = Comparator output filter clock = PCLK/4 11 = Reserved **Note:** Use FCLKDIV under the condition of FILTSEL = 3'h7, then set FCLKDIV to get the effect of filtering 128,256 consecutive PCLKs.
[19]	**Reserved**	Reserved.
[18]	**WCMPSEL**	**Window Compare Mode Selection** 0 = Window Compare Mode Disabled. 1 = Window Compare Mode is Selected.
[17]	**WLATEN**	**Window Latch Mode Enable Bit** 0 = Window Latch Mode Disabled. 1 = Window Latch Mode Enabled.
[16]	**WKEN**	**Power-down Wake-up Enable Bit** 0 = Wake-up function Disabled. 1 = Wake-up function Enabled.
[15:13]	**FILTSEL**	**Comparator Output Filter Count Selection**

		000 = Filter function is Disabled. 001 = ACMP0 output is sampled 1 consecutive PCLK. 010 = ACMP0 output is sampled 2 consecutive PCLKs. 011 = ACMP0 output is sampled 4 consecutive PCLKs. 100 = ACMP0 output is sampled 8 consecutive PCLKs. 101 = ACMP0 output is sampled 16 consecutive PCLKs. 110 = ACMP0 output is sampled 32 consecutive PCLKs. 111 = ACMP0 output is sampled 64 consecutive PCLKs.
[12]	**OUTSEL**	**Comparator Output Select** 0 = Comparator 0 output to ACMP0_O pin is unfiltered comparator output. 1 = Comparator 0 output to ACMP0_O pin is from filter output.
[11:10]	**Reserved**	Reserved.
[9:8]	**INTPOL**	**Interrupt Condition Polarity Selection** ACMPIF0 will be set to 1 when comparator output edge condition is detected. 00 = Rising edge or falling edge. 01 = Rising edge. 10 = Falling edge. 11 = Reserved.
[7:6]	**POSSEL**	**Comparator Positive Input Selection** 00 = Input from ACMP0_P0. 01 = Input from ACMP0_P1. 10 = Input from ACMP0_P2. 11 = Input from ACMP0_P3.
[5:4]	**NEGSEL**	**Comparator Negative Input Selection** 00 = ACMP0_N pin. 01 = Internal comparator reference voltage (CRV0). 10 = Band-gap voltage. 11 = DAC0 output. **Note:** NEGSEL must select 2'b01 in calibration mode.
[3]	**ACMPOINV**	**Comparator Output Inverse** 0 = Comparator 0 output inverse Disabled. 1 = Comparator 0 output inverse Enabled.
[2]	**Reserved**	Reserved.
[1]	**ACMPIE**	**Comparator Interrupt Enable Bit** 0 = Comparator 0 interrupt Disabled. 1 = Comparator 0 interrupt Enabled. If WKEN (ACMP_CTL0[16]) is set to 1, the wake-up interrupt function will be enabled as well.
[0]	**ACMPEN**	**Comparator Enable Bit** 0 = Comparator 0 Disabled. 1 = Comparator 0 Enabled.

Analog Comparator 1 Control Register (ACMP_CTL1)

Register	Offset	R/W	Description	Reset Value
ACMP_CTL1	ACMP01_BA+0x04	R/W	Analog Comparator 1 Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					HYSSEL		
23	22	21	20	19	18	17	16
Reserved		FCLKDIV		Reserved	WCMPSEL	WLATEN	WKEN
15	14	13	12	11	10	9	8
FILTSEL			OUTSEL	Reserved		INTPOL	
7	6	5	4	3	2	1	0
POSSEL		NEGSEL		ACMPOINV	Reserved	ACMPIE	ACMPEN

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:24]	**HYSSEL**	**Hysteresis Mode Selection** 000 = Hysteresis is 0mV. 010 = Hysteresis is 20mV. 100 = Hysteresis is 40mV Others = Reserved
[23:22]	**Reserved**	Reserved.
[21:20]	**FCLKDIV**	**Comparator Output Filter Clock Divider** 00 = comparator output filter clock = PCLK 01 = comparator output filter clock = PCLK/2 10 = comparator output filter clock = PCLK/4 11 = Reserved
[19]	**Reserved**	Reserved.
[18]	**WCMPSEL**	**Window Compare Mode Selection** 0 = Window Compare Mode Disabled. 1 = Window Compare Mode is Selected.
[17]	**WLATEN**	**Window Latch Mode Enable Bit** 0 = Window Latch Mode Disabled. 1 = Window Latch Mode Enabled.
[16]	**WKEN**	**Power-down Wakeup Enable Bit** 0 = Wake-up function Disabled. 1 = Wake-up function Enabled.

Bits	Description	
[15:13]	**FILTSEL**	**Comparator Output Filter Count Selection** 000 = Filter function is Disabled. 001 = ACMP1 output is sampled 1 consecutive PCLK. 010 = ACMP1 output is sampled 2 consecutive PCLKs. 011 = ACMP1 output is sampled 4 consecutive PCLKs. 100 = ACMP1 output is sampled 8 consecutive PCLKs. 101 = ACMP1 output is sampled 16 consecutive PCLKs. 110 = ACMP1 output is sampled 32 consecutive PCLKs. 111 = ACMP1 output is sampled 64 consecutive PCLKs.
[12]	**OUTSEL**	**Comparator Output Select** 0 = Comparator 1 output to ACMP1_O pin is unfiltered comparator output. 1 = Comparator 1 output to ACMP1_O pin is from filter output.
[11:10]	**Reserved**	Reserved.
[9:8]	**INTPOL**	**Interrupt Condition Polarity Selection** ACMPIF1 will be set to 1 when comparator output edge condition is detected. 00 = Rising edge or falling edge. 01 = Rising edge. 10 = Falling edge. 11 = Reserved.
[7:6]	**POSSEL**	**Comparator Positive Input Selection** 00 = Input from ACMP1_P0. 01 = Input from ACMP1_P1. 10 = Input from ACMP1_P2. 11 = Input from ACMP1_P3.
[5:4]	**NEGSEL**	**Comparator Negative Input Selection** 00 = ACMP1_N pin. 01 = Internal comparator reference voltage (CRV1). 10 = Band-gap voltage. 11 = DAC0 output. **Note:** NEGSEL must select 2'b01 in calibration mode.
[3]	**ACMPOINV**	**Comparator Output Inverse Control** 0 = Comparator 1 output inverse Disabled. 1 = Comparator 1 output inverse Enabled.
[2]	**Reserved**	Reserved.
[1]	**ACMPIE**	**Comparator Interrupt Enable Bit** 0 = Comparator 1 interrupt Disabled. 1 = Comparator 1 interrupt Enabled. If WKEN (ACMP_CTL1[16]) is set to 1, the wake-up interrupt function will be enabled as well.
[0]	**ACMPEN**	**Comparator Enable Bit** 0 = Comparator 1 Disabled. 1 = Comparator 1 Enabled.

Analog Comparator Status Register (ACMP_STATUS)

Register	Offset	R/W	Description	Reset Value
ACMP_STATUS	ACMP01_BA+0x08	R/W	Analog Comparator Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							ACMPWO
15	14	13	12	11	10	9	8
Reserved		ACMPS1	ACMPS0	Reserved		WKIF1	WKIF0
7	6	5	4	3	2	1	0
Reserved		ACMPO1	ACMPO0	Reserved		ACMPIF1	ACMPIF0

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**ACMPWO**	**Comparator Window Output** This bit shows the output status of window compare mode 0 = The positive input voltage is outside the window. 1 = The positive input voltage is in the window.
[15:14]	**Reserved**	Reserved.
[13]	**ACMPS1**	**Comparator 1 Status** Synchronized to the PCLK to allow reading by software. Cleared when the comparator 1 is disabled, i.e. ACMPEN (ACMP_CTL1[0]) is cleared to 0.
[12]	**ACMPS0**	**Comparator 0 Status** Synchronized to the PCLK to allow reading by software. Cleared when the comparator 0 is disabled, i.e. ACMPEN (ACMP_CTL0[0]) is cleared to 0.
[11:10]	**Reserved**	Reserved.
[9]	**WKIF1**	**Comparator 1 Power-down Wake-up Interrupt Flag** This bit will be set to 1 when ACMP1 wake-up interrupt event occurs. 0 = No power-down wake-up occurred. 1 = Power-down wake-up occurred. **Note:** Write 1 to clear this bit to 0.
[8]	**WKIF0**	**Comparator 0 Power-down Wake-up Interrupt Flag** This bit will be set to 1 when ACMP0 wake-up interrupt event occurs. 0 = No power-down wake-up occurred. 1 = Power-down wake-up occurred. **Note:** Write 1 to clear this bit to 0.
[7:6]	**Reserved**	Reserved.
[5]	**ACMPO1**	**Comparator 1 Output** Synchronized to the PCLK to allow reading by software. Cleared when the comparator 1 is disabled, i.e.

Bits	Description	
		ACMPEN (ACMP_CTL1[0]) is cleared to 0.
[4]	**ACMPO0**	**Comparator 0 Output** Synchronized to the PCLK to allow reading by software. Cleared when the comparator 0 is disabled, i.e. ACMPEN (ACMP_CTL0[0]) is cleared to 0.
[3:2]	**Reserved**	Reserved.
[1]	**ACMPIF1**	**Comparator 1 Interrupt Flag** This bit is set by hardware when the edge condition defined by INTPOL (ACMP_CTL1[9:8]) is detected on comparator 1 output. This will cause an interrupt if ACMPIE (ACMP_CTL1[1]) is set to 1. **Note:** Write 1 to clear this bit to 0.
[0]	**ACMPIF0**	**Comparator 0 Interrupt Flag** This bit is set by hardware when the edge condition defined by INTPOL (ACMP_CTL0[9:8]) is detected on comparator 0 output. This will generate an interrupt if ACMPIE (ACMP_CTL0[1]) is set to 1. **Note:** Write 1 to clear this bit to 0.

ACMP Reference Voltage Control Register (ACMP_VREF)

Register	Offset	R/W	Description	Reset Value
ACMP_VREF	ACMP01_BA+0x0C	R/W	Analog Comparator Reference Voltage Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							CRV1EN
23	22	21	20	19	18	17	16
Reserved	CRV1SSEL	CRV1SEL					
15	14	13	12	11	10	9	8
Reserved							CRV0EN
7	6	5	4	3	2	1	0
Reserved	CRV0SSEL	CRV0SEL					

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**CRV1EN**	**CRV1 Enable Bit** 0 = CRV1 Disabled. 1 = CRV1 Enabled.
[23]	**Reserved**	Reserved.
[22]	**CRV1SSEL**	**CRV1 Source Voltage Selection** 0 = AV_{DD} is selected as CRV1 source voltage. 1 = The reference voltage defined by SYS_VREFCTL register is selected as CRV1 source voltage.
[21:16]	**CRV1SEL**	**Comparator1 Reference Voltage Setting** CRV1 = CRV1 source voltage * (ACMP_VREF[21:16])/ 63.
[15:9]	**Reserved**	Reserved.
[8]	**CRV0EN**	**CRV0 Enable Bit** 0 = CRV0 Disabled. 1 = CRV0 Enabled.
[7]	**Reserved**	Reserved.
[6]	**CRV0SSEL**	**CRV0 Source Voltage Selection** 0 = AV_{DD} is selected as CRV0 source voltage. 1 = The reference voltage defined by SYS_VREFCTL register is selected as CRV0 source voltage.
[5:0]	**CRV0SEL**	**Comparator0 Reference Voltage Setting** CRV0 = CRV0 source voltage * (ACMP_VREF[5:0])/ 63.

Analog Comparator Calibration Control Register (ACMP_CALCTL)

Register	Offset	R/W	Description	Reset Value
ACMP_CALCTL	ACMP01_BA+0x10	R/W	Analog Comparator Calibration Control Register	0x0000_0050

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						CALTRG1	CALTRG0

Bits	Description	
[31:2]	Reserved	Reserved.
[1]	CALTRG1	**Comparator1 Calibration Trigger Bit** 0 = Calibration is stopped. 1 = Calibration is triggered. **Note 1:** Before this bit is enabled, ACMPEN(ACMP_CTL1[0]) should be set and the internal high speed RC oscillator (HIRC) should be enabled in advance. **Note 2:** Hardware will auto clear this bit when the next calibration is triggered by software. **Note 3:** If user must trigger calibration twice or more times, the second trigger has to wait at least 300us after the previous calibration is done.
[0]	CALTRG0	**Comparator0 Calibration Trigger Bit** 0 = Calibration is stopped. 1 = Calibration is triggered. **Note 1:** Before this bit is enabled,ACMPEN(ACMP_CTL0[0]) should be set and the internal high speed RC oscillator (HIRC) should be enabled in advance. **Note 2:** Hardware will auto clear this bit when the next calibration is triggered by software. **Note 3:** If user must trigger calibration twice or more times, the second trigger has to wait at least 300us after the previous calibration is done.

Analog Comparator Calibration Status Register (ACMP_CALSR)

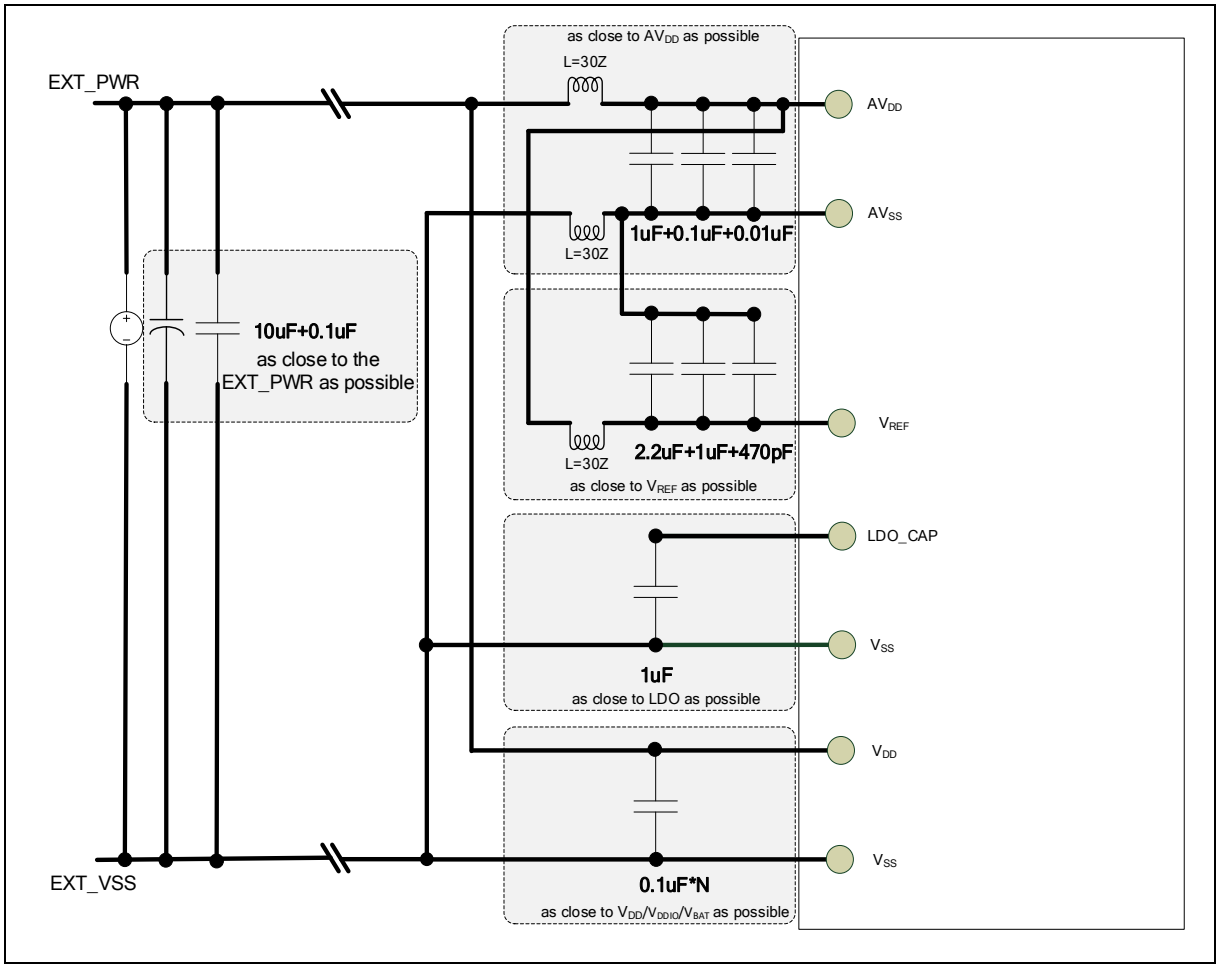
Register	Offset	R/W	Description	Reset Value
ACMP_CALSR	ACMP01_BA+0x14	R	Analog Comparator Calibration Status Register	0x0000_0000

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	CALPS1	CALNS1	DONE1	Reserved	CALPS0	CALNS0	DONE0

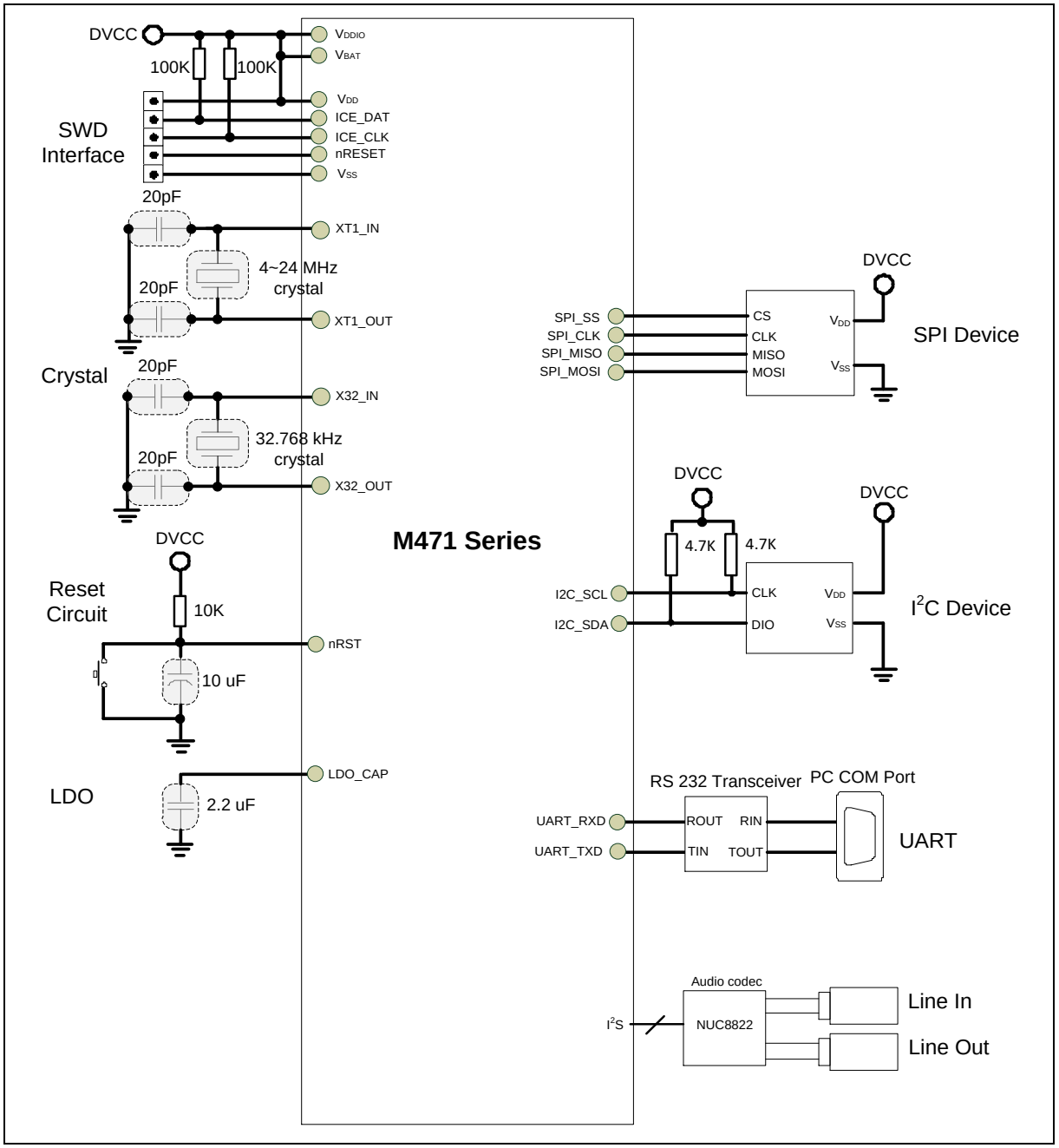
Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**CALPS1**	**Comparator1 Calibration Result Status for PMOS** 0 = Pass. 1 = Fail.
[5]	**CALNS1**	**Comparator1 Calibration Result Status for NMOS** 0 = Pass. 1 = Fail.
[4]	**DONE1**	**Comparator1 Calibration Done Status** 0 = Calibrating. 1 = Calibration done.
[3]	**Reserved**	Reserved.
[2]	**CALPS0**	**Comparator0 Calibration Result Status for PMOS** 0 = Pass. 1 = Fail.
[1]	**CALNS0**	**Comparator0 Calibration Result Status for NMOS** 0 = Pass. 1 = Fail.
[0]	**DONE0**	**Comparator0 Calibration Done Status** 0 = Calibrating. 1 = Calibration done.

7 APPLICATION CIRCUIT

7.1 Power Supply Scheme



7.2 Peripheral Application Scheme



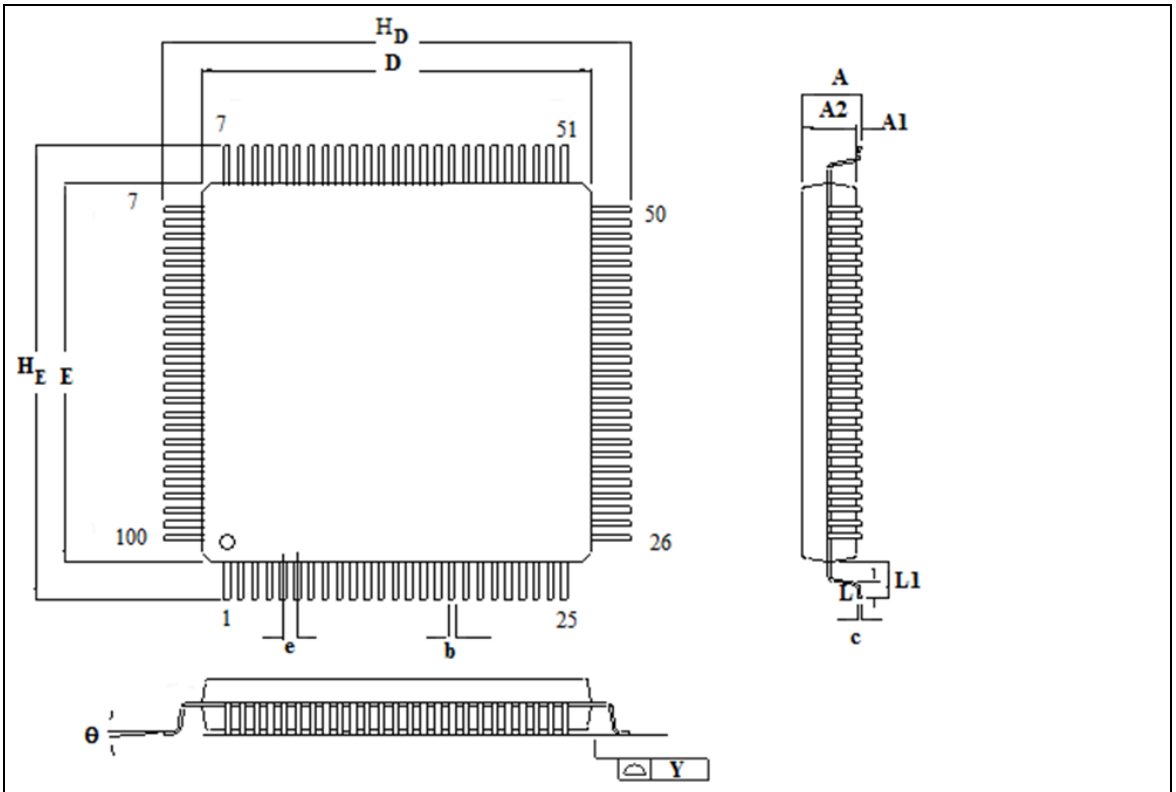
8 ELECTRICAL CHARACTERISTICS

Please refer to the relative Datasheet for detailed information about the M471V/M471K/M471C electrical characteristics.

9 PACKAGE DIMENSIONS

Package is Halogen-free, RoHS-compliant and TSCA-compliant.

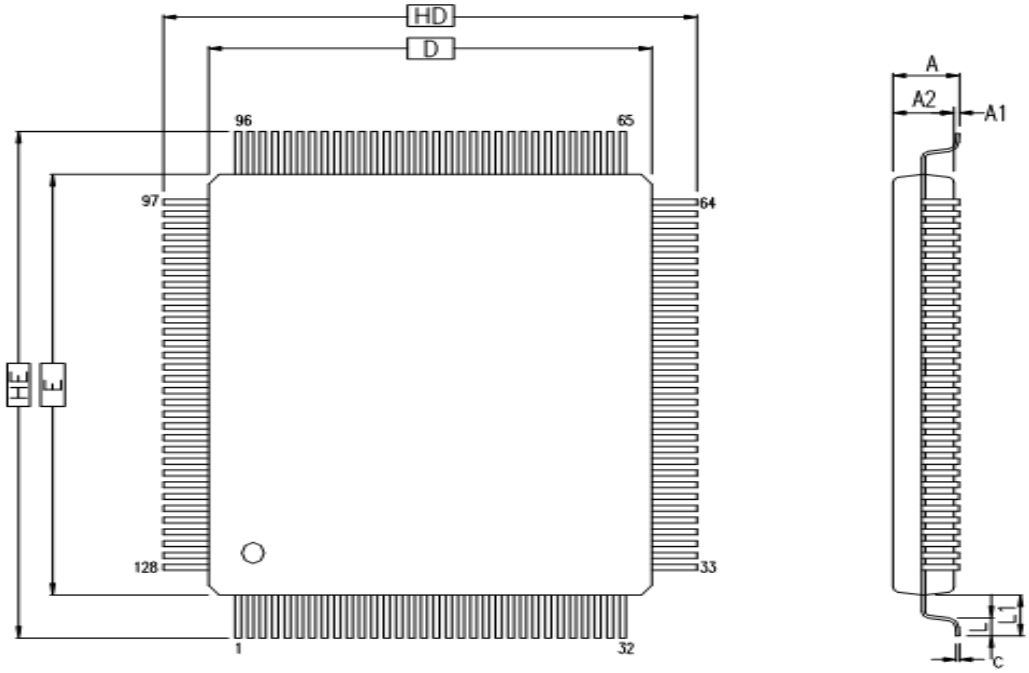
9.1 LQFP 100L (14x14x1.4 mm^3 footprint 2.0 mm)



Controlling Dimension : Millimeters

SYMBOLS	MIN.	NOM.	MAX.
A	-	-	1.60
A1	0.05	-	-
A2	1.35	1.40	1.45
b	0.17	0.22	0.27
c	0.10	0.15	0.20
D	13.90	14.00	14.10
E	13.90	14.00	14.10
e	-	0.50	-
HD	15.80	16.00	16.20
HE	15.80	16.00	16.20
L	0.45	0.60	0.75
L1	-	1	-
y	-	-	0.10
θ	0°	-	7°

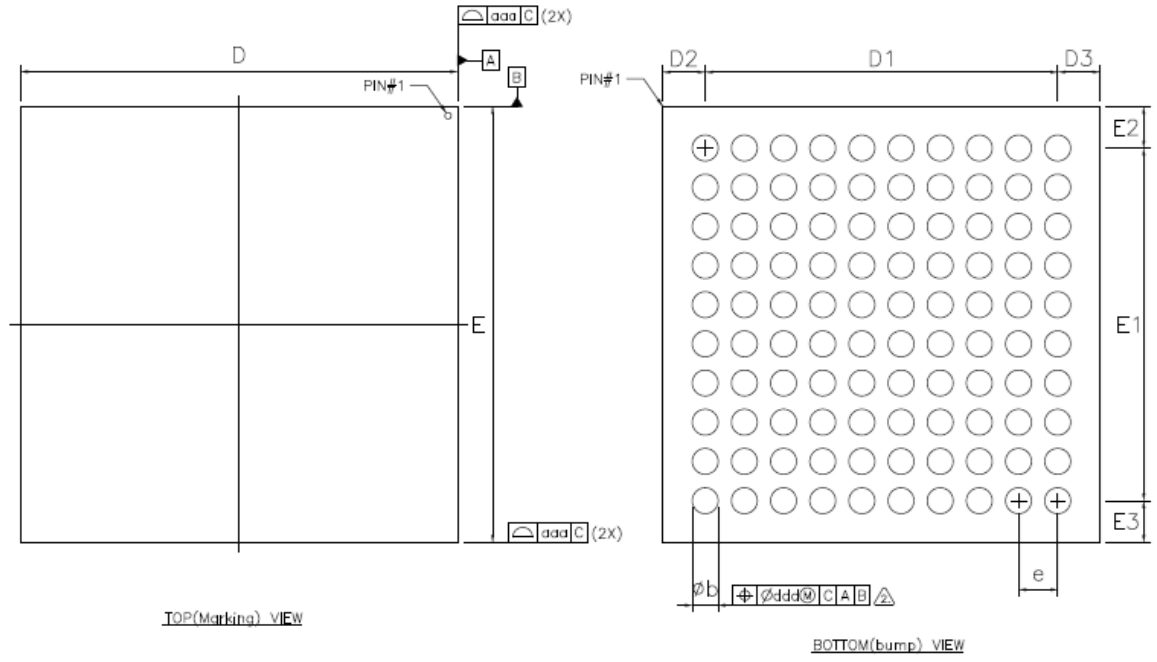
9.2 LQFP 128L (14x14x1.4 mm^3 footprint 2.0 mm)

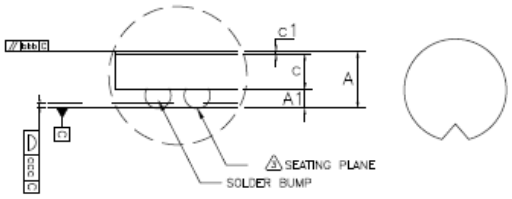


COTROL DIMENSIONS ARE IN MILLIMETERS.

SYMBOL	MILLIMETER			INCH		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
A	—	—	1.60	—	—	0.063
A1	0.05	—	0.15	0.002	—	0.006
A2	1.35	1.40	1.45	0.053	0.055	0.057
HD	16.00 BSC.			0.630 BSC.		
D	14.00 BSC.			0.551 BSC.		
HE	16.00 BSC.			0.630 BSC.		
E	14.00 BSC.			0.551 BSC.		
b	0.13	0.16	0.23	0.005	0.006	0.009
e	0.40 BSC.			0.016 BSC.		
θ	0°	3.5°	7°	0°	3.5°	7°
c	0.09	—	0.20	0.004	—	0.008
L	0.45	0.60	0.75	0.018	0.024	0.030
L_1	1.00 REF			0.039 REF		
y	-	-	0.1	-	-	0.004

9.3 WLCSP100 (4.4648x 4.4480x 0.58 mm^3 , ball pitch 0.4 mm)





Symbol	Dimensions in mm			Dimensions in inch		
	MIN	NOM	MAX	MIN	NOM	MAX
A	0.5320	0.5800	0.6280	0.0209	0.0228	0.0247
A1	0.1800	0.2000	0.2200	0.0071	0.0079	0.0087
c1	0.0220	0.0250	0.0280	0.0009	0.0010	0.0011
c	0.3300	0.3550	0.3800	0.0130	0.0140	0.0150
D	4.4448	4.4648	4.4848	0.1750	0.1758	0.1766
E	4.4280	4.4480	4.4680	0.1743	0.1751	0.1759
b	0.2236	0.2630	0.3025	0.0088	0.0104	0.0119
D1	---	3.6000	---	---	0.1417	---
D2	---	0.4324	---	---	0.0170	---
D3	---	0.4324	---	---	0.0170	---
E1	---	3.6000	---	---	0.1417	---
E2	---	0.4240	---	---	0.0167	---
E3	---	0.4240	---	---	0.0167	---
e	---	0.4000	---	---	0.0157	---
aaa	---	0.0200	---	---	0.0008	---
bbb	---	0.0600	---	---	0.0024	---
ccc	---	0.0500	---	---	0.0020	---
ddd	---	0.0150	---	---	0.0006	---

10 ABBREVIATIONS

10.1 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
CCAP	Camera Capture Interface
DAP	Debug Access Port
DES	Data Encryption Standard
EADC	Enhanced Analog-to-Digital Converter
EBI	External Bus Interface
EMAC	Ethernet MAC Controller
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	12 MHz Internal High Speed RC Oscillator
HXT	4~24 MHz External High Speed Crystal Oscillator
IAP	In Application Programming
ICP	In Circuit Programming
ISP	In System Programming
LDO	Low Dropout Regulator
LIN	Local Interconnect Network
LIRC	10 kHz internal low speed RC oscillator (LIRC)
MPU	Memory Protection Unit
NVIC	Nested Vectored Interrupt Controller
PCLK	The Clock of Advanced Peripheral Bus
PDMA	Peripheral Direct Memory Access
PLL	Phase-Locked Loop

PWM	Pulse Width Modulation
QEI	Quadrature Encoder Interface
SD	Secure Digital
SPI	Serial Peripheral Interface
SPS	Samples per Second
TDES	Triple Data Encryption Standard
TK	Touch Key
TMR	Timer Controller
UART	Universal Asynchronous Receiver/Transmitter
UCID	Unique Customer ID
USB	Universal Serial Bus
WDT	Watchdog Timer
WWDT	Window Watchdog Timer

Table 10.1-1 List of Abbreviations

11 REVISION HISTORY

Date	Revision	Description
2020.09.15	1.00	1. Initial version.
2021.05.15	1.10	1. Added section 6.7 PDMA Controller (PDMA).
2021.06.15	1.20	1. Updated cover page. 2. Updated chapter 1 GENERAL DESCRIPTION. 3. Updated section 3.3 M471V/M471K/M471C Series Selection Guide. 4. Updated section 3.2 M471V/M471K/M471C Series Naming Rule. 5. Updated section 5.1 NuMicro® M471V/M471K/M471C Block Diagram.
2021.12.22	1.30	1. Added a note: "CPU access operation only can read word alignment data in Data Flash memory." in section 6.5.4.3. 2. Updated ADC reference voltage MUX description in "Figure 6.20-1" and section 6.20.5.18 3. Updated UART TX delay description in section 6.14.5.4. 4. Added a note: "Repeat START (Sr) is sent immediately when I^2C triggers Repeat START signal. It should wait 0.5 I^2C clock cycle to trigger Repeat START signal." in section 6.16.5.1
2022.12.22	1.40	1. Added new part number M471CI8AE. 2. Added "Package is Halogen-free, RoHS-compliant and TSCA-compliant." in Chapter 3 and 9.

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.