Arm® Cortex®-M

32-bit Microcontroller

NuMicro® Family
M031/M032 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® M031/M032 series 32-bit microcontroller is based on Arm® Cortex®-M0 core with 32-bit hardware multiplier/divider. It features 1.8 ~ 3.6 V operating voltage, 5 V I/O tolerant, and runs up to 48/72 MHz within -40°C ~105°C.

The M031/M032 series provides a solution for the applications that need 1.8 V low-voltage interface connection with enhanced fast 2 MSPS conversion rate 12-bit ADC, comparators and up-to 24-ch 96/144 MHz PWM control. It supports a fast and precise data conversion for the voltage, current, and sensor data, then fast response control to the external device. Additionally, the M031/M032 series also provides plenty of peripherals including Universal Serial Control Interface(USCI) that can be set as UART/SPI/I^2C flexibly, up to 10 sets of UART, 4 sets of SPI, 4 sets of I^2C, and 1-wire UART interface for data communication between master and slave devices.

The M031/M032 series provides Flash size from 16 Kbytes to 512 Kbytes, SRAM size from 2 Kbytes to 96 Kbytes. Supported packages from small form factor TSSOP 20-pin, TSSOP 28-pin, QFN 33-pin, LQFP 48-pin to LQFP 64-pin and LQFP 128-pin with pin-compatible for different part numbers makes the system design and parts change easily.

Part Numbers with the M032 series are all based on the M031 series and enhanced with the crystal-less USB 2.0 full-speed device feature for USB related applications.

For the development, Nuvoton provides the NuMaker-PFM 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	UART	I^2C	SPI/ I^2S	QSPI	USCI	Timer	PWM	RTC	PDMA	EBI	ADC	ACMP	Divider	USBD	IEC60730
M031/M032	8	2	1	1	2	4	24	1	9	1	16	2	1	√	√

Table 1-1 NuMicro® M031/M032 Series Key Features Support Table

The NuMicro® M031/M032 series is suitable for a wide range of applications such as:

- Laser Distance Meter
- Air Detector/Cleaner
- Mobile LCD Panel Controller
- IoT Sensing Device
- HMI Controller
- Micro Printer
- Gaming Keyboard and Mouse
- WPC Wireless Charger

2 FEATURES

2.1 M031/M032 Features

Core and System	
Arm® Cortex®-M0	• Arm® Cortex®-M0 processor, running up to 72 MHz – 72 MHz at 2.0V-3.6V – 48 MHz at 1.8V-3.6V • Built-in Nested Vectored Interrupt Controller (NVIC) • 24-bit system tick timer • Programmble and maskable interrupt • Low Power Sleep mode by WFI and WFE instructions
Brown-out Detector (BOD)	• Two-level BOD with brown-out interrupt and reset option. (2.5V/2.0V)
Low Voltage Reset (LVR)	• LVR with 1.7V threshold voltage level.
Security	• 96-bit Unique ID (UID). • 128-bit Unique Customer ID (UCID).
32-bit H/W Divider(HDIV)	• Signed (two's complement) integer calculation • 32-bit dividend with 16-bit divisor calculation capacity • 32-bit quotient and 32-bit remainder outputs (16-bit remainder with sign extends to 32-bit)
Memories	
Flash	• Dual bank 512 KB on-chip Application ROM (APROM) for Over-The-Air (OTA) upgrade. • Single bank up to 256 KB on-chip Application ROM (APROM). • Up to 8 KB on-chip Flash for user-defined loader (LDROM) • Up to 2048 bytes execution-only Security Protection ROM (SPROM) • All on-chip Flash support 512 bytes or 2048 bytes page erase • Fast Flash programming verification with CRC-32 checksum calculation • On-chip Flash programming with In-Chip Programming (ICP), In-System Programming (ISP) and In-Application Programming (IAP) capabilities • 2-wired ICP Flash updating through SWD/ICE interface
SRAM	• Up to 96 KB on-chip SRAM – 32 KB SRAM located in bank 0 that supports hardware parity check and retenion mode – 32/32 KB SRAM located in bank 1 and bank 2

	• Byte-, half-word- and word-access • 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 DMA to write data with performing CRC operation
Peripheral DMA (PDMA)	• Up to 9 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
Clocks	
External Clock Source	• 4~32 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	• 48 MHz High-speed Internal RC oscillator (HIRC) dedicated for crystal-less USB. • 38.4 kHz Low-speed Internal RC oscillator (LIRC) for watchdog timer and wakeup operation • Up to 144 MHz on-chip PLL, sourced from HIRC or HXT, allows CPU operation up to the maximim CPU frequency without the need for a high-frequency crystal
Real-Time Clock (RTC)	• The RTC clock source includes Low-speed external crystal oscillator (LXT) • Able to wake up CPU from idle or power-down mode • Supports ±5ppm 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	• Up to 4 sets of 32-bit timers with 24-bit up counter and one 8-bit pre-scale counter from independent clock source • One-shot, Periodic, Toggle and Continuous Counting operation modes • Supports event counting function to count the event from external pins • Supports external capture pin for interval measurement and resetting 24-bit up counter • Supports chip wake-up function, if a timer interrupt signal is generated
PWM (PWM)	• Up to two PWM modules, each module provides three 16-bit counter and 6 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 • Counter synchronous start function • Brake function with auto recovery mechanism • Mask function and tri-state output for each PWM channel • Able to trigger ADC to start conversion
Basic PWM (BPWM)	• Two 16-bit counters with 12-bit clock prescale for twelve 144 MHz PWM output channels. • Up to 6 independent input capture channels with 16-bit resolution counter • Up, down or up-down PWM counter type • Counter synchronous start function • Mask function and tri-state output for each PWM channel • Able to trigger ADC to start conversion
Watchdog	• 20-bit free running up counter for WDT time-out interval • Supports multiple clock sources from LIRC (default selection), HCLK/2048 and LXT with 9 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	
ADC	• Analog input voltage range: 0 ~ AV_{DD} • One 12-bit, 2 MSPS SAR ADC with up to 16 single-ended input channels or 8 differential input pairs; 10-bit accuracy is guaranteed. • Internal channels for band-gap VBG input. • Supports external V_{REF} pin. • Supports calibration capability. • Four operation modes: Single mode, Burst mode, Single-cycle Scan mode and Continuous Scan mode. • Analog-to-Digital conversion can be triggered by software enable (ADST), external pin (STADC), Timer 0~3 overflow pulse trigger, PWM trigger or BPWM trigger. • Each conversion result is held in data register of each channel with valid and overrun indicators. • Supports conversion result monitor by compare mode function. • Configurable ADC external sampling time. • PDMA operation. • Supports floating detect function.
Analog Comparator (ACMP)	• Two Analog Comparators • Supports four multiplexed I/O pins at positive input • Supports I/O pins, band-gap, and 16-level Voltage divider from AV_{DD} or V_{REF} at negative input • Supports wake up from Power-down by interrupt • Supports triggers for brake events and cycle-by-cycle control for PWM • Supports window compare mode and window latch mode • Supports hysteresis function • Supports calibration function
Communication Interfaces	
Low-power UART	• Low-power UARTs with up to 7.2 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.

	• 16-byte FIFOs with programmable level trigger • Auto flow control (nCTS and nRTS) • Supports IrDA (SIR) function • 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 enables 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. • Supports Single-wire function mode.
I^2C	• Three sets of I^2C devices with Master/Slave mode. • Supports Standard mode (100 kbps), Fast mode (400 kbps), Fast mode plus (1 Mbps) • Supports 7 bits mode • Programmable clocks allowing for versatile rate control • Supports multiple address recognition (four slave address with mask option) • Supports multi-address power-down wake-up function • PDMA operation • I^2C Port0 supports SMBus and PMBus
Quad SPI	• SPI Quad controller with Master/Slave mode. • Up to 24 MHz in Master mode and up to 16 MHz in Slave mode at 1.8V~3.6V system voltage. • Supports Dual and Quad I/O Transfer mode. • Supports one data channel half-duplex transfer. • Supports receive-only mode. • Configurable bit length of a transfer word from 8 to 32-bit. • Provides separate 8-level depth transmit and receive FIFO buffers. • Supports MSB first or LSB first transfer sequence. • Supports the byte reorder function. • Supports Byte or Word Suspend mode. • Supports 3-wired, no slave select signal, bi-direction interface. • PDMA operation. • Supports 2-bit Transfer mode.

SPI/I²S	• SPI/I²S controllers with Master/Slave mode. **SPI** • Up to 24 MHz in Master mode and up to 16 MHz in Slave mode at 1.8V~3.6V system voltage. • Configurable bit length of a transfer word from 8 to 32-bit. • Provides separate 4-level of 32-bit (or 8-level of 16-bit) transmit and receive FIFO buffers. • 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. • PDMA operation. **I²S** • Supports mono and stereo audio data with 8-, 16-, 24- and 32-bit audio data sizes. • Provides separate 4-level depth transmit and receive FIFO buffers. • Supports PCM mode A, PCM mode B, I²S and MSB justified data format. • PDMA operation. • Generates interrupt requests when buffer levels cross as programmable boundary
Universal Serial Control Interface (USCI)	• Two sets of USCI, configured as UART, SPI or I²C function. • Supports single byte TX and RX buffer mode **UART** • Supports one transmit buffer and two receive buffers for data payload. • Supports hardware auto flow control function and programmable flow control trigger level. • 9-bit Data Transfer. • Baud rate detection by built-in capture event of baud rate generator. • Supports wake-up function. • PDMA operation. **SPI** • Supports Master or Slave mode operation. • Supports one transmit buffer and two receive buffer for data payload. • Configurable bit length of a transfer word from 4 to 16-bit. • Supports MSB first or LSB first transfer sequence. • Supports Word Suspend function. • Supports 3-wire, no slave select signal, bi-direction interface. • Supports wake-up function: input slave select transition.

	• PDMA operation. • Supports one data channel half-duplex transfer. **I^2C** • Supports master and slave device capability. • Supports one transmit buffer and two receive buffer for data payload. • Communication in standard mode (100 kbps), fast mode (up to 400 kbps), and Fast mode plus (1 Mbps). • Supports 7-bit mode (10-bit mode not supported). • Supports 10-bit bus time out capability. • Supports bus monitor mode. • Supports power-down wake-up by data toggle or address match. • Supports multiple address recognition. • Supports device address flag. • Programmable setup/hold time.
External Bus Interface (EBI)	• Supports up to two memory banks with individual adjustment of timing parameter. • Each bank supports dedicated external chip select pin with polarity control and up to 1 MB addressing space. • 8-/16-bit data width. • Supports byte write in 16-bit data width mode. • Configurable idle cycle for different access condition: Idle of Write command finish (W2X) and Idle of Read-to-Read (R2R). • Supports Address/Data multiplexed mode. • Supports address bus and data bus separate mode. • Supports LCD interface i80 mode. • PDMA operation.
GPIO	• Supports four I/O modes: Quasi bi-direction, Push-Pull output, Open-Drain output and Input only with high impendence mode. • Configured as interrupt source with edge/level trigger setting. • I/O pin internal pull-up resistor enabled only in Quasi-bidirectional I/O mode. • Supports 5V-tolerance function except analog IO (PA.10, PA.11, PB.0~PB.15, PF.2~PF.5). • Enabling the pin interrupt function will also enable the wake-up function. • Input schmitt trigger function.
Advanced Connectivity	
USB 2.0 Full Speed with on-chip transceiver	• Compliant with USB Revision 2.0 Specification. • Supports suspend function when no bus activity existing for 3 ms. • 8 configurable endpoints for configurable Isochronous, Bulk,

Interrupt and Control transfer types.

- 512 bytes configurable RAM for endpoint buffer.
- Remote wake-up capability.
- Supports Crystal-less function
- Start of Frame (SOF) locked clock pulse generation
- USB 2.0 link power management.

3 PARTS INFORMATION

3.1 Package Type

Part No.	TSSOP20	TSSOP28	QFN33	LQFP48	LQFP64	LQFP128
M031xB	M031FB0AE	M031EB0AE	M031TB0AE			
M031xC	M031FC1AE	M031EC1AE	M031TC1AE	M031LC2AE	M031SC2AE	
M031xD			M031TD2AE	M031LD2AE	M031SD2AE	
M031xE			M031TE3AE	M031LE3AE	M031SE3AE	
M031xG				M031LG6AE M031LG8AE	M031SG6AE M031SG8AE	M031KG6AE M031KG8AE
M031xI					M031SIAAE	M031KIAAE
M032xC	M032FC1AE	M032EC1AE	M032TC1AE	M032LC2AE		
M032xD			M032TD2AE	M032LD2AE		
M032xE				M032LE3AE	M032SE3AE	
M032xG				M032LG6AE M032LG8AE	M032SG6AE M032SG8AE	M032KG6AE M032KG8AE
M032xI					M032SIAAE	M032KIAAE

3.2 M031/M032 Series Selection Guide

3.2.1 M031 Base Series (M031Fx / M031Ex / M031Tx)

Part Number		M031							
		FB0AE	FC1AE	EB0AE	EC1AE	TB0AE	TC1AE	TD2AE	TE3AE
Flash (KB)		16	32	16	32	16	32	64	128
SRAM (KB)		2	4	2	4	2	4	8	16
LDROM (KB)		2	2	2	2	2	2	2	4
SPROM (Bytes)		512							
System Frequency (MHz)		48							
PLL (MHz)		-	-	-	-	-	-	96	96
I/O		15	15	23	23	27	27	27	27
32-bit Timer		2	4	2	4	2	4	4	4
Connectivity	USCI	-	-	-	-	-	-	1	1
	UART	3							
	SPI/I^{2}S	1							
	QSPI	-							
	I^{2}C/SMBus	2/0							
	USB FS	-							
PWM		6	6	6	6	6	6	12	12
BPWM		-							
PDMA		-	2	-	2	-	2	5	5
EBI		-							
HDIV		√							
CRC		√							
IEC-60730		-							
HXT		√							
LXT		-	-	-	-	-	√	√	√
RTC		-							
Analog Comparator		-	-	-	-	-	-	2	2
12-bit SAR ADC		7	7	9	9	10	10	10	10
Package		TSSOP20	TSSOP20	TSSOP28	TSSOP28	QFN33	QFN33	QFN33	QFN33

3.2.2 M031 Base Series (M031Lx)

Part Number		M031				
		LC2AE	LD2AE	LE3AE	LG6AE	LG8AE
Flash (KB)		32	64	128	256	256
SRAM (KB)		8	8	16	32	64
LDROM (KB)		2	2	4	4	4
SPROM (Bytes)		512	512	512	2048	2048
System Frequency (MHz)		48	48	48	72	72
PLL (MHz)		96	96	96	144	144
I/O		42				
32-bit Timer		4				
Connectivity	USCI	1	1	1	2	2
	UART	3	3	3	6	6
	SPI/I^{2}S	1				
	QSPI	-	-	-	1	1
	I²C/SMBus	2/0	2/0	2/0	2/1	2/1
	USB FS	-	-	-	-	-
PWM		12				
BPWM		-	-	-	12	12
PDMA		5	5	5	7	7
EBI		-	-	√	√	√
CRC		√				
HDIV		√				
IEC-60730		-	-	-	√	√
HXT		√				
LXT		√				
RTC		-	-	-	√	√
Analog Comparator		2				
12-bit SAR ADC		12				
Package		LQFP48				

3.2.3 M031 Base Series (M031Sx)

Part Number		M031					
		SC2AE	SD2AE	SE3AE	SG6AE	SG8AE	SIAAE
Flash (KB)		32	64	128	256	256	512
SRAM (KB)		8	8	16	32	64	96
LDROM (KB)		2	2	4	4	4	8
SPROM (Bytes)		512	512	512	2048	2048	2048
System Frequency (MHz)		48	48	48	72	72	72
PLL (MHz)		96	96	96	144	144	144
I/O		55					
32-bit Timer		4					
Connectivity	USCI	1	1	1	2	2	2
	UART	3	3	3	6	6	8
	SPI/I^2S	1					
	QSPI	-	-	-	1	1	1
	I²C/SMBus	2/0	2/0	2/0	2/1	2/1	2/1
	USB FS	-					
PWM		12					
BPWM		-	-	-	12	12	12
PDMA		5	5	5	7	7	9
EBI		-	-	√	√	√	√
CRC		√					
HDIV		√					
IEC-60730		-	-	-	√	√	√
HXT		√					
LXT		√					
RTC		-	-	-	√	√	√
Analog Comparator		2					
12-bit SAR ADC		16					
Package		LQFP64					

3.2.4 M031 Base Series (M031Kx)

Part Number		M031		
		KG6AE	KG8AE	KIAAE
Flash (KB)		256	256	512
SRAM (KB)		32	64	96
LDROM (KB)		4	4	8
SPROM (Bytes)		2048		
System Frequency (MHz)		72		
PLL (MHz)		144		
I/O		111		
32-bit Timer		4		
Connectivity	USCI	2		
	UART	6	6	8
	SPI/I²S	1		
	QSPI	1		
	I²C/SMBus	2/1		
	USB FS	-		
PWM		12		
BPWM		12		
PDMA		7	7	9
EBI		√		
CRC		√		
HDIV		√		
IEC-60730		√		
HXT		√		
LXT		√		
RTC		√		
Analog Comparator		2		
12-bit SAR ADC		16		
Package		LQFP128		

3.2.5 M032 USB Series (M032Fx / M032Ex / M032Tx)

Part Number		M032			
		FC1AE	EC1AE	TC1AE	TD2AE
Flash (KB)		32	32	32	64
SRAM (KB)		4	4	4	8
LDROM (KB)		2			
SPROM (Bytes)		512			
System Frequency (MHz)		48			
PLL (MHz)		-			
I/O		11	19	23	23
32-bit Timer		2	2	2	4
Connectivity	USCI	1	1	1	2
	UART	1			
	SPI/I²S	1			
	QSPI	-	-	-	1
	I²C/SMBus	-			
	USB FS	√			
PWM		-			
BPWM		6	6	6	12
PDMA		-	-	-	5
EBI		-			
CRC		-			
HDIV		-	-	-	√
IEC-60730		-			
HXT		-			
LXT		-			
RTC		-			
Analog Comparator		-			
12-bit SAR ADC		3	9	10	10
Package		TSSOP20	TSSOP28	QFN33	QFN33

3.2.6 M032 USB Series (M032Lx)

Part Number		M032				
		LC2AE	LD2AE	LE3AE	LG6AE	LG8AE
Flash (KB)		32	64	128	256	256
SRAM (KB)		8	8	16	32	64
LDROM (KB)		2	2	4	4	4
SPROM (Bytes)		512	512	512	2048	2048
System Frequency (MHz)		48	48	48	72	72
PLL (MHz)		-	-	96	144	144
I/O		38				
32-bit Timer		4				
Connectivity	USCI	2	2	1	2	2
	UART	1	1	3	6	6
	SPI/I²S	1				
	QSPI	1	1	-	1	1
	I²C/SMBus	-	-	2/0	2/1	2/1
	USB FS	√				
PWM		-	-	12	12	12
BPWM		12	12	-	12	12
PDMA		5	5	5	7	7
EBI		-	-	√	√	√
CRC		-	-	√	√	√
HDIV		√				
IEC-60730		-	-	-	√	√
HXT		-	-	√	√	√
LXT		-	-	√	√	√
RTC		-	-	-	√	√
Analog Comparator		-	-	2	2	2
12-bit SAR ADC		12				
Package		LQFP48				

3.2.7 M032 USB Series (M032Sx)

Part Number		M032			
		SE3AE	SG6AE	SG8AE	SIAAE
Flash (KB)		128	256	256	512
SRAM (KB)		16	32	64	96
LDROM (KB)		4	4	4	8
SPROM (Bytes)		512	2048	2048	2048
System Frequency (MHz)		48	72	72	72
PLL (MHz)		96	144	144	144
I/O		51			
32-bit Timer		4			
Connectivity	USCI	1	2	2	2
	UART	3	6	6	8
	SPI/I^{2}S	1			
	QSPI	-	1	1	1
	I^{2}C/SMBus	2/0	2/1	2/1	2/1
	USB FS	√			
PWM		12			
BPWM		-	12	12	12
PDMA		5	7	7	9
EBI		√			
CRC		√			
HDIV		√			
IEC-60730		-	√	√	√
HXT		√			
LXT		√			
RTC		-	√	√	√
Analog Comparator		2			
12-bit SAR ADC		16			
Package		LQFP64			

3.2.8 M032 USB Series (M032Kx)

Part Number		M032		
		KG6AE	KG8AE	KIAAE
Flash (KB)		256	256	512
SRAM (KB)		32	64	96
LDROM (KB)		4	4	8
SPROM (Bytes)		2048		
System Frequency (MHz)		72		
PLL (MHz)		144		
I/O		107		
32-bit Timer		4		
Connectivity	USCI	2		
	UART	6	6	8
	SPI/I²S	1		
	QSPI	1		
	I²C/SMBus	2/1		
	USB FS	√		
PWM		12		
BPWM		12		
PDMA		7	7	9
EBI		√		
CRC		√		
HDIV		√		
IEC-60730		√		
HXT		√		
LXT		√		
RTC		√		
Analog Comparator		2		
12-bit SAR ADC		16		
Package		LQFP128		

3.2.9 Naming Rule

M0	32	K	I	A	A	E
Core	**Line**	**Package**	**Flash**	**SRAM**	**Reserve**	**Temperature**
Cortex®-M0	31: Base 32: USB	F: TSSOP20 (4.4x6.5 mm) E: TSSOP28 (4.4x9.7 mm) T: QFN33 (4x4 mm) L: LQFP48 (7x7 mm) S: LQFP64 (7x7 mm) K: LQFP128 (14x14 mm)	B: 16 KB C: 32 KB D: 64 KB E: 128 KB G: 256 KB I: 512 KB	0: 2 KB 1: 4 KB 2: 8/12 KB 3: 16 KB 6: 32 KB 8: 64 KB A: 96 KB		E:-40°C ~ 105°C

3.3 M031/M032 Series Feature Comparison Table

Section	Sub-section	-	M031xB/C/D/E	M031xG/I
		M032xC/D	M032xE	M032xG/I
System Manager	6.3.6 SRAM Memory Organization	●	●	-
	6.3.7 SRAM Memory Organization with parity function	-	-	●
FMC	6.4.4.3 Physical and Virtual Address Concept	-	-	●
	6.4.4.4 APROM Reboot Address Operation Model Selection	-	-	-/●
	6.4.4.14 Cache Memory Controller	-	-	●
	6.4.4.15 Embedded Flash Memory Programming 64-bit Programming and Multi-word Programming	-	-	●
	6.4.4.17 Flash All-One Verification	-	-	●
	ISP Control Register (FMC_ISPCTL) INTEN (FMC_ISPCTL[24])	-	●	
ADC	6.25.5.11 PWM trigger	-	●	●
	6.25.5.12 BPWM trigger	●	-	●
	6.25.5.17 Floating Detect Function	●	-	●
I^2C	6.16.5.2 Operation Modes - Bus Management (SMBus/PMBus Compatiable) - Device Identification – Slave Address - Bus Protocols - Address Resolution Protocol (ARP) - Received Command and Data acknowledge control - Host Notify Protocol - Bus Management Alert - Packet Error Checking - Time-out - Bus Management Time-out: - Bus Clock Low Time-out: - Bus Idle Detection	-	-	●
ACMP	6.26.5.7 Calibration function	-	-/-/-/●	●
EBI	6.21.5.3 EBI Data Width Connection - Address Bus and Data Bus Separate Mode	-	-	●
	6.21.5.4 EBI Operating Control - Continuous Data Access Mode	-	-	●
USBD	6.22.7 Register Description USB Configuration Register (USB_CFGx)	●	-	-

Section	Sub-section	-	M031xB/C/D/E	M031xG/I
		M032xC/D	M032xE	M032xG/I
	DSQSYNC OUT Token Transaction			

4 PIN CONFIGURATION

Users can find pin configuaration informations in chapter 4 or by using NuTool - PinConfig. The NuTool - PinConfigure 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 M031 Series Pin Diagram

4.1.1.1 M031 Series TSSOP 20-Pin Diagram

Corresponding Part Number: M031FB0AE, M031FC1AE

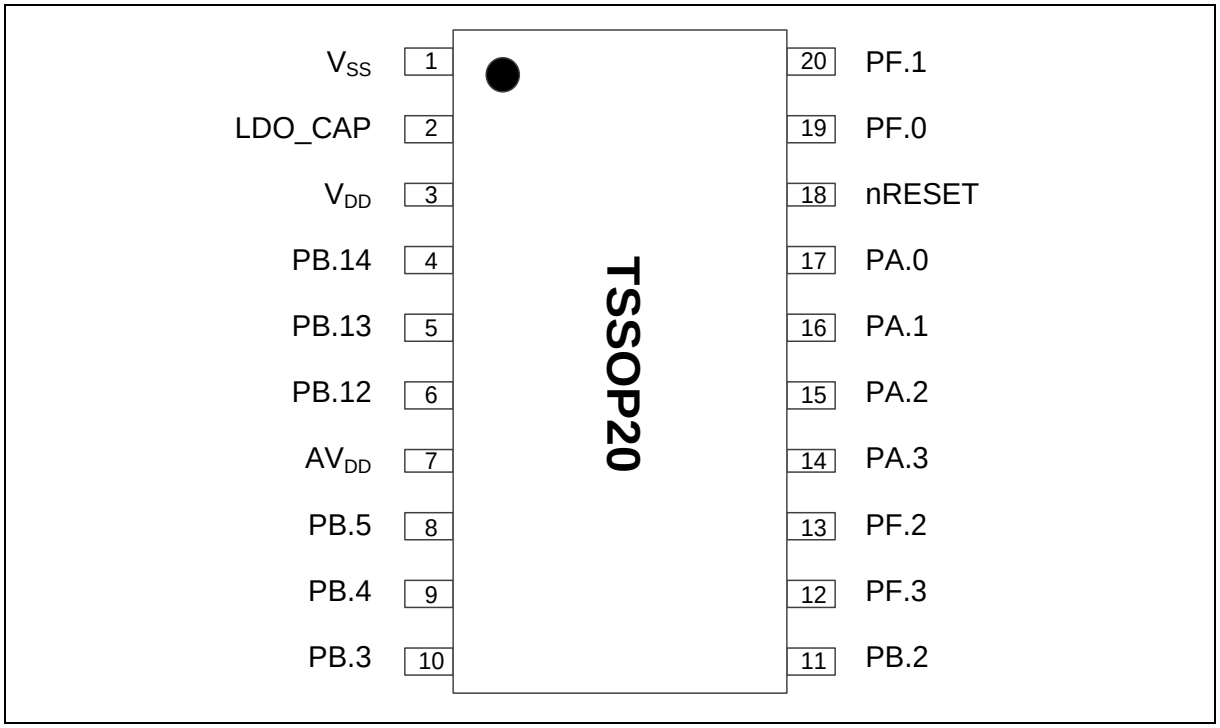


Figure 4.1-1 M031 Series TSSOP 20-pin Diagram

4.1.1.2 M031 Series TSSOP 28-Pin Diagram

Corresponding Part Number: M031EB0AE, M031EC1AE

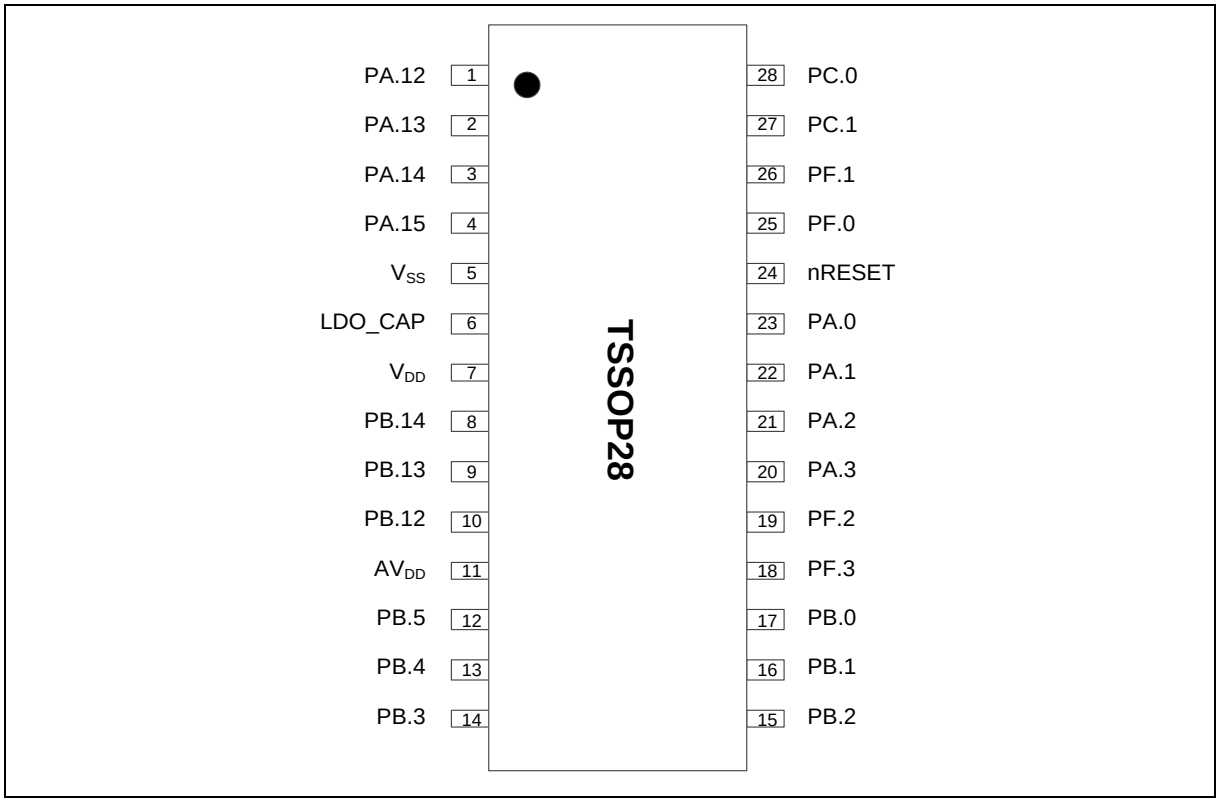


Figure 4.1-2 M031 Series TSSOP 28-pin Diagram

4.1.1.3 M031 Series QFN 33-Pin Diagram

Corresponding Part Number: M031TB0AE, M031TC1AE, M031TD2AE, M031TE3AE

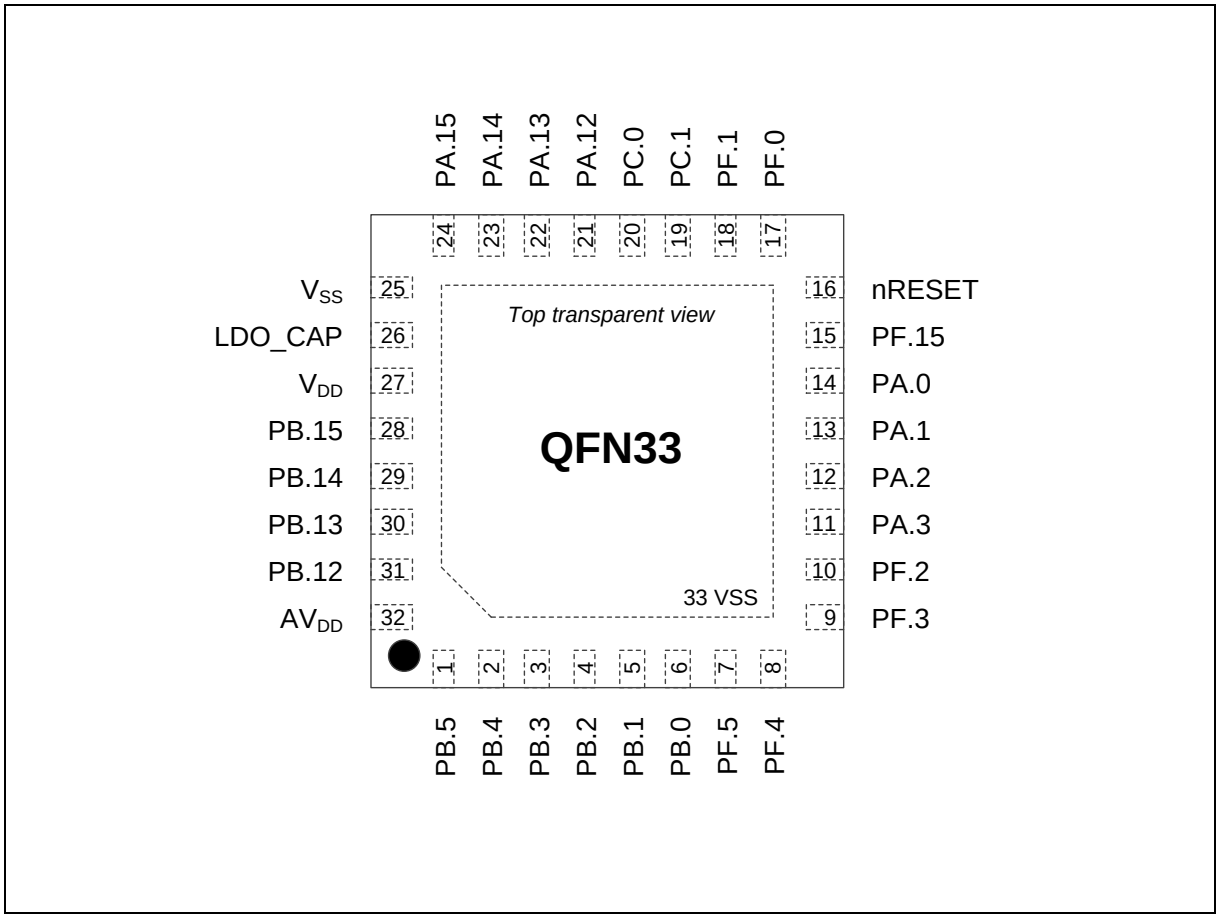


Figure 4.1-3 M031 Series QFN 33-pin Diagram

4.1.1.4 M031 Series LQFP 48-Pin Diagram

Corresponding Part Number: M031LC2AE, M031LD2AE, M031LE3AE, M031LG6AE, M031LG8AE

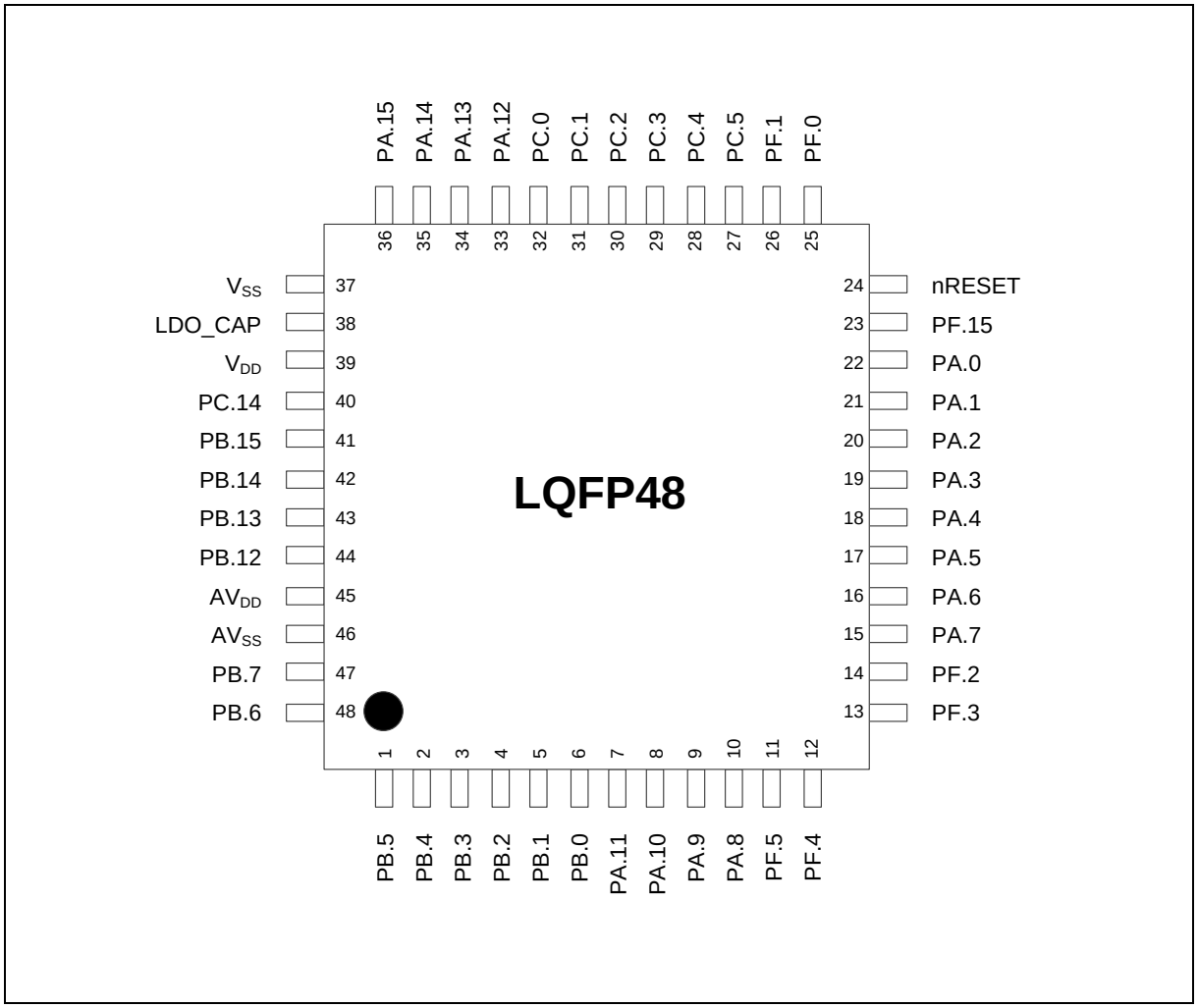


Figure 4.1-4 M031 Series LQFP 48-pin Diagram

4.1.1.5 M031 Series LQFP 64-Pin Diagram

Corresponding Part Number: M031SC2AE, M031SD2AE, M031SE3AE, M031SG6AE, M031SG8AE, M031SIAAE

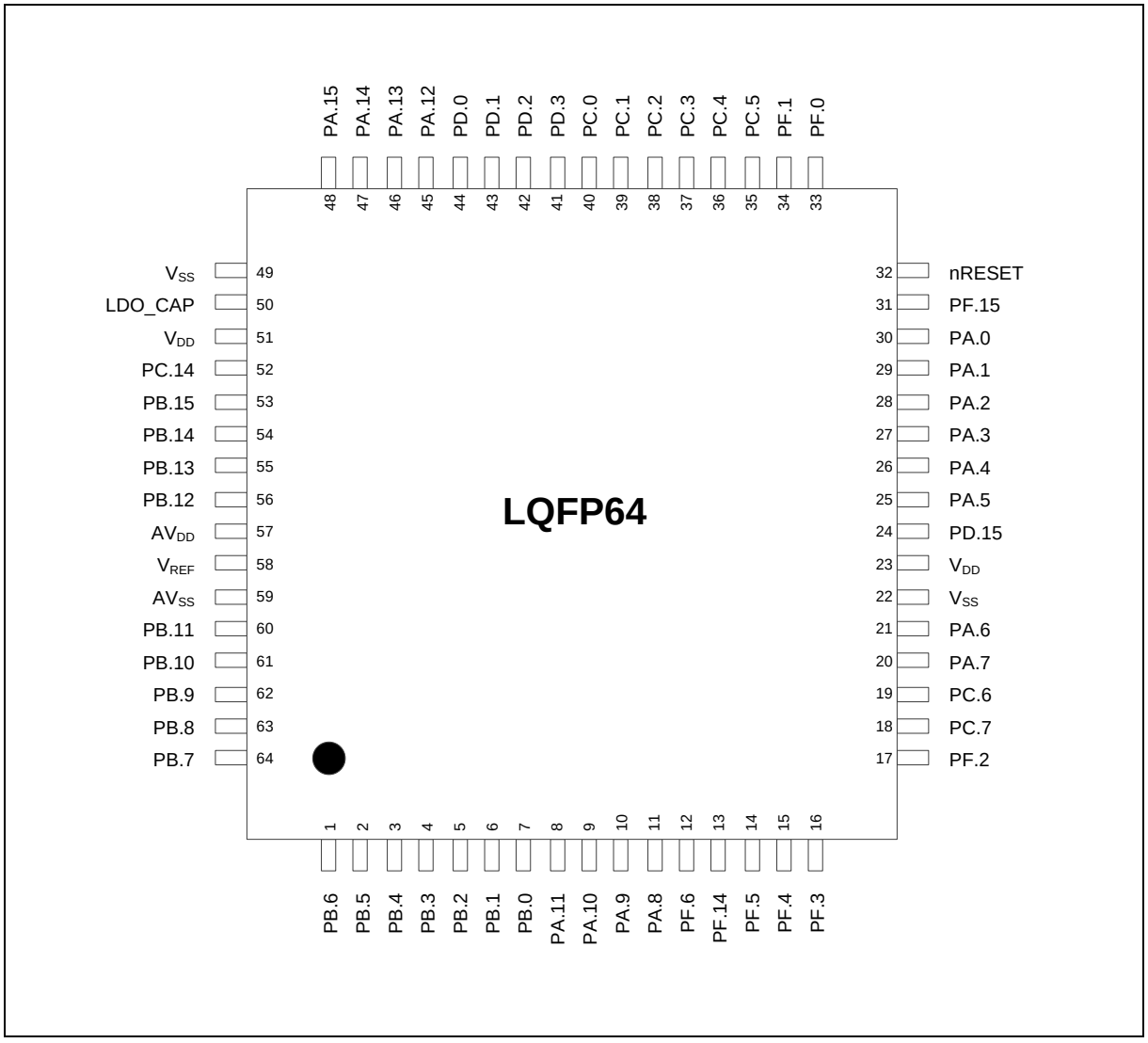


Figure 4.1-5 M031 Series LQFP 64-pin Diagram

4.1.1.6 M031 Series LQFP 128-Pin Diagram

Corresponding Part Number: M031KG6AE, M031KG8AE, M031KIAAE

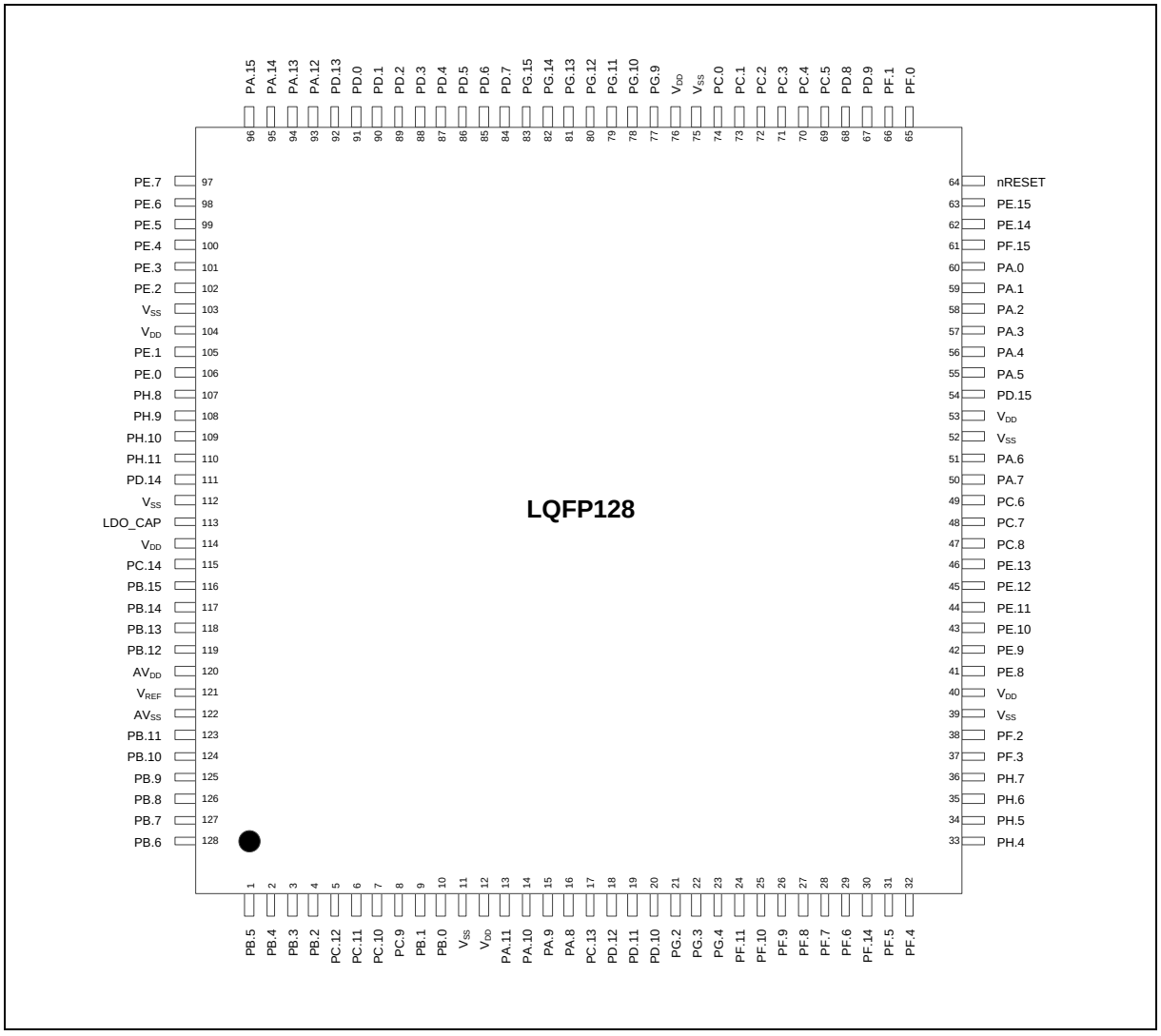


Figure 4.1-6 M031 Series LQFP 128-pin Diagram

4.1.2 M031 Series Multi-function Pin Diagram

4.1.2.1 M031 Series TSSOP 20-Pin Multi-function Pin Diagram

Corresponding Part Number: M031FB0AE, M031FC1AE

M031FB0AE

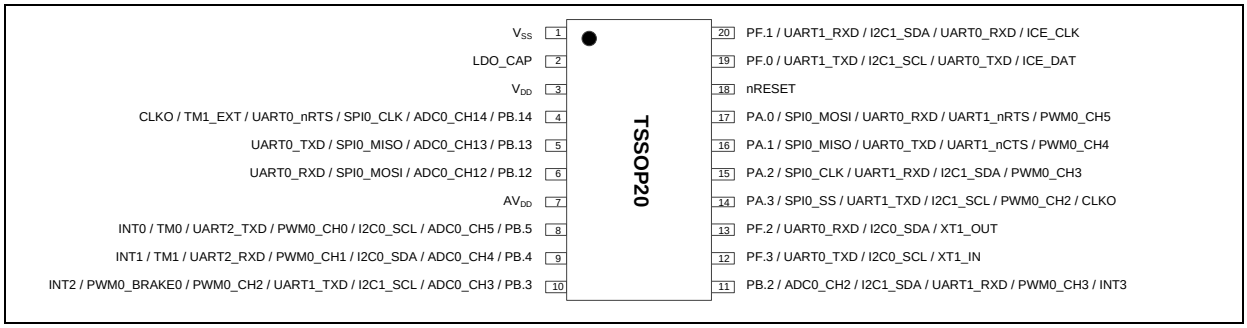


Figure 4.1-7 M031FB0AE Multi-function Pin Diagram

Pin	M031FB0AE Pin Function
1	VSS
2	LDO_CAP
3	V_{DD}
4	PB.14 / ADC0_CH14 / SPI0_CLK / UART0_nRTS / TM1_EXT / CLKO
5	PB.13 / ADC0_CH13 / SPI0_MISO / UART0_TXD
6	PB.12 / ADC0_CH12 / SPI0_MOSI / UART0_RXD
7	AV_{DD}
8	PB.5 / ADC0_CH5 / I2C0_SCL / PWM0_CH0 / UART2_TXD / TM0 / INT0
9	PB.4 / ADC0_CH4 / I2C0_SDA / PWM0_CH1 / UART2_RXD / TM1 / INT1
10	PB.3 / ADC0_CH3 / I2C1_SCL / UART1_TXD / PWM0_CH2 / PWM0_BRAKE0 / INT2
11	PB.2 / ADC0_CH2 / I2C1_SDA / UART1_RXD / PWM0_CH3 / INT3
12	PF.3 / UART0_TXD / I2C0_SCL / XT1_IN
13	PF.2 / UART0_RXD / I2C0_SDA / XT1_OUT
14	PA.3 / SPI0_SS / UART1_TXD / I2C1_SCL / PWM0_CH2 / CLKO
15	PA.2 / SPI0_CLK / UART1_RXD / I2C1_SDA / PWM0_CH3
16	PA.1 / SPI0_MISO / UART0_TXD / UART1_nCTS / PWM0_CH4
17	PA.0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / PWM0_CH5
18	nRESET
19	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / ICE_DAT
20	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / ICE_CLK

Table 4.1-1 M031FB0AE Multi-function Pin Table

M031FC1AE

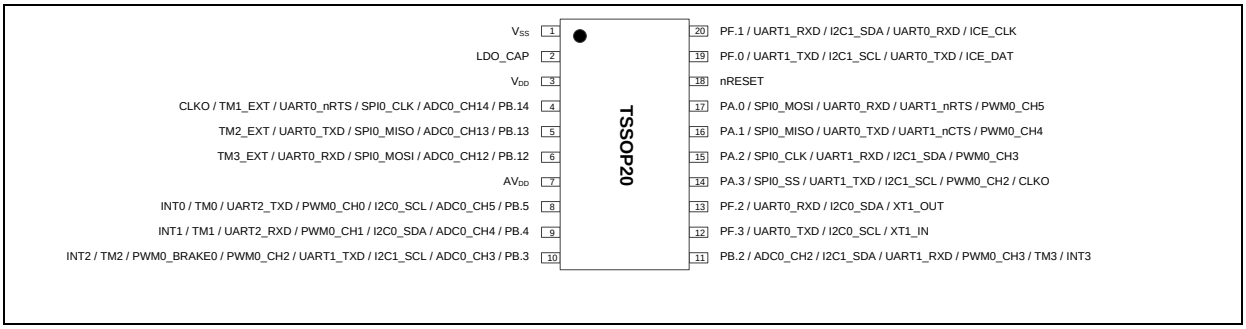


Figure 4.1-8 M031FC1AE Multi-function Pin Diagram

Pin	M031FC1AE Pin Function
1	VSS
2	LDO_CAP
3	V_{DD}
4	PB.14 / ADC0_CH14 / SPI0_CLK / UART0_nRTS / TM1_EXT / CLKO
5	PB.13 / ADC0_CH13 / SPI0_MISO / UART0_TXD / TM2_EXT
6	PB.12 / ADC0_CH12 / SPI0_MOSI / UART0_RXD / TM3_EXT
7	AV_{DD}
8	PB.5 / ADC0_CH5 / I2C0_SCL / PWM0_CH0 / UART2_TXD / TM0 / INT0
9	PB.4 / ADC0_CH4 / I2C0_SDA / PWM0_CH1 / UART2_RXD / TM1 / INT1
10	PB.3 / ADC0_CH3 / I2C1_SCL / UART1_TXD / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2
11	PB.2 / ADC0_CH2 / I2C1_SDA / UART1_RXD / PWM0_CH3 / TM3 / INT3
12	PF.3 / UART0_TXD / I2C0_SCL / XT1_IN
13	PF.2 / UART0_RXD / I2C0_SDA / XT1_OUT
14	PA.3 / SPI0_SS / UART1_TXD / I2C1_SCL / PWM0_CH2 / CLKO
15	PA.2 / SPI0_CLK / UART1_RXD / I2C1_SDA / PWM0_CH3
16	PA.1 / SPI0_MISO / UART0_TXD / UART1_nCTS / PWM0_CH4
17	PA.0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / PWM0_CH5
18	nRESET
19	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / ICE_DAT
20	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / ICE_CLK

Table 4.1-2 M031FC1AE Multi-function Pin Table

4.1.2.2 M031 Series TSSOP 28-Pin Multi-function Pin Diagram

Corresponding Part Number: M031EB0AE, M031EC1AE

M031EB0AE

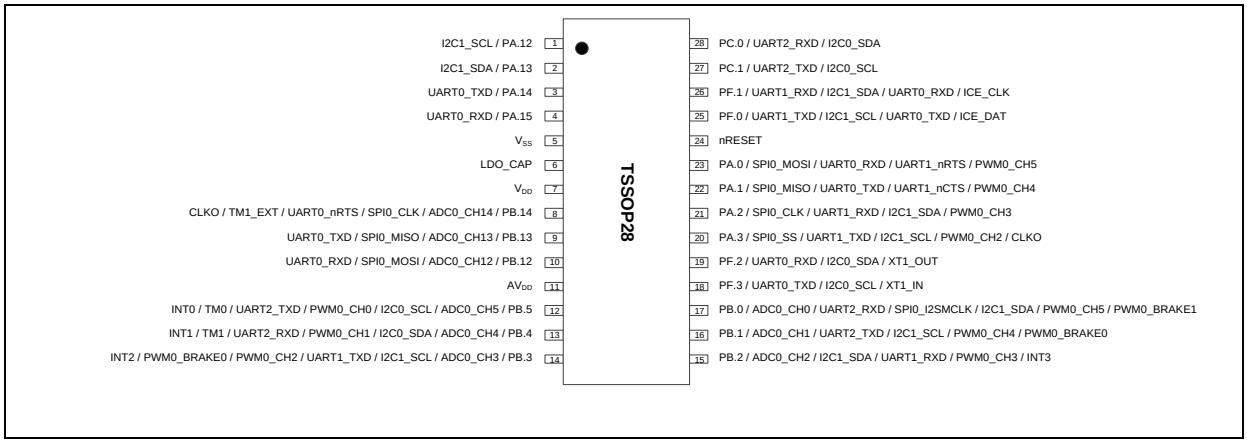


Figure 4.1-9 M031EB0AE Multi-function Pin Diagram

Pin	M031EB0AE Pin Function
1	PA.12 / I2C1_SCL
2	PA.13 / I2C1_SDA
3	PA.14 / UART0_TXD
4	PA.15 / UART0_RXD
5	VSS
6	LDO_CAP
7	V_{DD}
8	PB.14 / ADC0_CH14 / SPI0_CLK / UART0_nRTS / TM1_EXT / CLKO
9	PB.13 / ADC0_CH13 / SPI0_MISO / UART0_TXD
10	PB.12 / ADC0_CH12 / SPI0_MOSI / UART0_RXD
11	AV_{DD}
12	PB.5 / ADC0_CH5 / I2C0_SCL / PWM0_CH0 / UART2_TXD / TM0 / INT0
13	PB.4 / ADC0_CH4 / I2C0_SDA / PWM0_CH1 / UART2_RXD / TM1 / INT1
14	PB.3 / ADC0_CH3 / I2C1_SCL / UART1_TXD / PWM0_CH2 / PWM0_BRAKE0 / INT2
15	PB.2 / ADC0_CH2 / I2C1_SDA / UART1_RXD / PWM0_CH3 / INT3
16	PB.1 / ADC0_CH1 / UART2_TXD / I2C1_SCL / PWM0_CH4 / PWM0_BRAKE0
17	PB.0 / ADC0_CH0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / PWM0_CH5 / PWM0_BRAKE1
18	PF.3 / UART0_TXD / I2C0_SCL / XT1_IN
19	PF.2 / UART0_RXD / I2C0_SDA / XT1_OUT
20	PA.3 / SPI0_SS / UART1_TXD / I2C1_SCL / PWM0_CH2 / CLKO
21	PA.2 / SPI0_CLK / UART1_RXD / I2C1_SDA / PWM0_CH3

Pin	M031EB0AE Pin Function
22	PA.1 / SPI0_MISO / UART0_TXD / UART1_nCTS / PWM0_CH4
23	PA.0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / PWM0_CH5
24	nRESET
25	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / ICE_DAT
26	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / ICE_CLK
27	PC.1 / UART2_TXD / I2C0_SCL
28	PC.0 / UART2_RXD / I2C0_SDA

Table 4.1-3 M031EB0AE Multi-function Pin Table

M031EC1AE

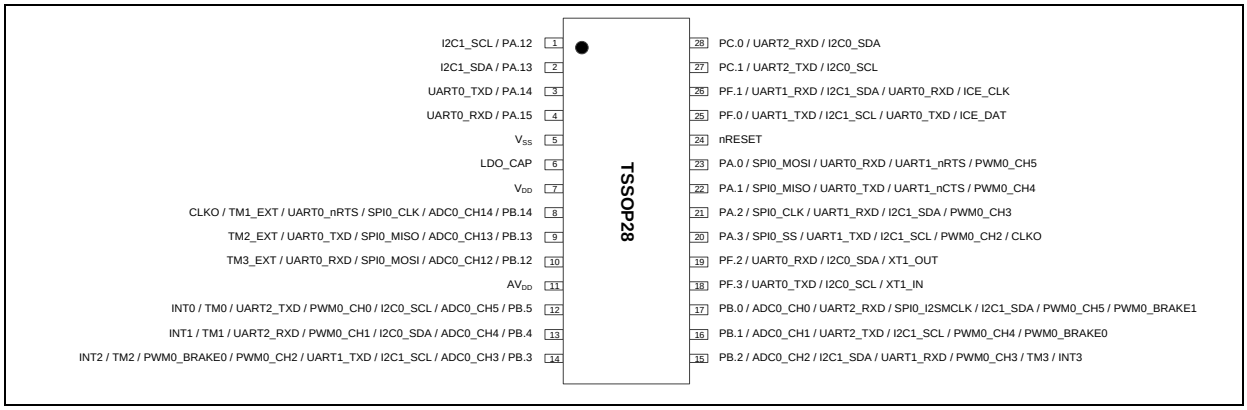


Figure 4.1-10 M031EC1AE Multi-function Pin Diagram

Pin	M031EC1AE Pin Function
1	PA.12 / I2C1_SCL
2	PA.13 / I2C1_SDA
3	PA.14 / UART0_TXD
4	PA.15 / UART0_RXD
5	VSS
6	LDO_CAP
7	V_{DD}
8	PB.14 / ADC0_CH14 / SPI0_CLK / UART0_nRTS / TM1_EXT / CLKO
9	PB.13 / ADC0_CH13 / SPI0_MISO / UART0_TXD / TM2_EXT
10	PB.12 / ADC0_CH12 / SPI0_MOSI / UART0_RXD / TM3_EXT
11	AV_{DD}
12	PB.5 / ADC0_CH5 / I2C0_SCL / PWM0_CH0 / UART2_TXD / TM0 / INT0
13	PB.4 / ADC0_CH4 / I2C0_SDA / PWM0_CH1 / UART2_RXD / TM1 / INT1
14	PB.3 / ADC0_CH3 / I2C1_SCL / UART1_TXD / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2

Pin	M031EC1AE Pin Function
15	PB.2 / ADC0_CH2 / I2C1_SDA / UART1_RXD / PWM0_CH3 / TM3 / INT3
16	PB.1 / ADC0_CH1 / UART2_TXD / I2C1_SCL / PWM0_CH4 / PWM0_BRAKE0
17	PB.0 / ADC0_CH0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / PWM0_CH5 / PWM0_BRAKE1
18	PF.3 / UART0_TXD / I2C0_SCL / XT1_IN
19	PF.2 / UART0_RXD / I2C0_SDA / XT1_OUT
20	PA.3 / SPI0_SS / UART1_TXD / I2C1_SCL / PWM0_CH2 / CLKO
21	PA.2 / SPI0_CLK / UART1_RXD / I2C1_SDA / PWM0_CH3
22	PA.1 / SPI0_MISO / UART0_TXD / UART1_nCTS / PWM0_CH4
23	PA.0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / PWM0_CH5
24	nRESET
25	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / ICE_DAT
26	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / ICE_CLK
27	PC.1 / UART2_TXD / I2C0_SCL
28	PC.0 / UART2_RXD / I2C0_SDA

Table 4.1-4 M031EC1AE Multi-function Pin Table

4.1.2.3 M031 Series QFN 33-Pin Multi-function Pin Diagram

Corresponding Part Number: M031TB0AE, M031TC1AE, M031TD2AE

M031TB0AE

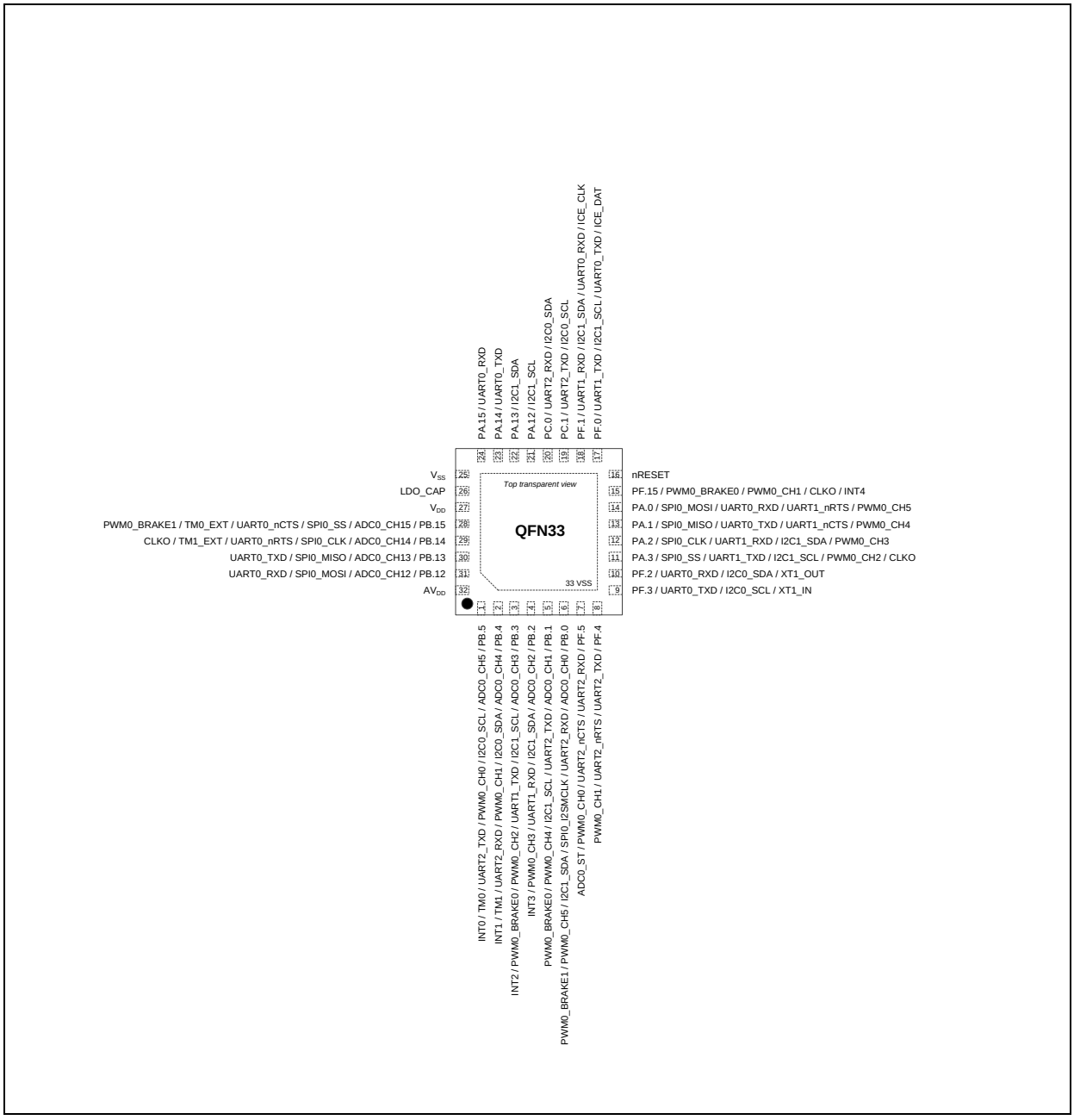


Figure 4.1-11 M031TB0AE Multi-function Pin Diagram

Pin	M031TB0AE Pin Function
1	PB.5 / ADC0_CH5 / I2C0_SCL / PWM0_CH0 / UART2_TXD / TM0 / INT0
2	PB.4 / ADC0_CH4 / I2C0_SDA / PWM0_CH1 / UART2_RXD / TM1 / INT1
3	PB.3 / ADC0_CH3 / I2C1_SCL / UART1_TXD / PWM0_CH2 / PWM0_BRAKE0 / INT2

Pin	M031TB0AE Pin Function
4	PB.2 / ADC0_CH2 / I2C1_SDA / UART1_RXD / PWM0_CH3 / INT3
5	PB.1 / ADC0_CH1 / UART2_TXD / I2C1_SCL / PWM0_CH4 / PWM0_BRAKE0
6	PB.0 / ADC0_CH0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / PWM0_CH5 / PWM0_BRAKE1
7	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / ADC0_ST
8	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1
9	PF.3 / UART0_TXD / I2C0_SCL / XT1_IN
10	PF.2 / UART0_RXD / I2C0_SDA / XT1_OUT
11	PA.3 / SPI0_SS / UART1_TXD / I2C1_SCL / PWM0_CH2 / CLKO
12	PA.2 / SPI0_CLK / UART1_RXD / I2C1_SDA / PWM0_CH3
13	PA.1 / SPI0_MISO / UART0_TXD / UART1_nCTS / PWM0_CH4
14	PA.0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / PWM0_CH5
15	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / CLKO / INT4
16	nRESET
17	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / ICE_DAT
18	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / ICE_CLK
19	PC.1 / UART2_TXD / I2C0_SCL
20	PC.0 / UART2_RXD / I2C0_SDA
21	PA.12 / I2C1_SCL
22	PA.13 / I2C1_SDA
23	PA.14 / UART0_TXD
24	PA.15 / UART0_RXD
25	VSS
26	LDO_CAP
27	V_{DD}
28	PB.15 / ADC0_CH15 / SPI0_SS / UART0_nCTS / TM0_EXT / PWM0_BRAKE1
29	PB.14 / ADC0_CH14 / SPI0_CLK / UART0_nRTS / TM1_EXT / CLKO
30	PB.13 / ADC0_CH13 / SPI0_MISO / UART0_TXD
31	PB.12 / ADC0_CH12 / SPI0_MOSI / UART0_RXD
32	AV_{DD}

Table 4.1-5 M031TB0AE Multi-function Pin Table

M031TC1AE

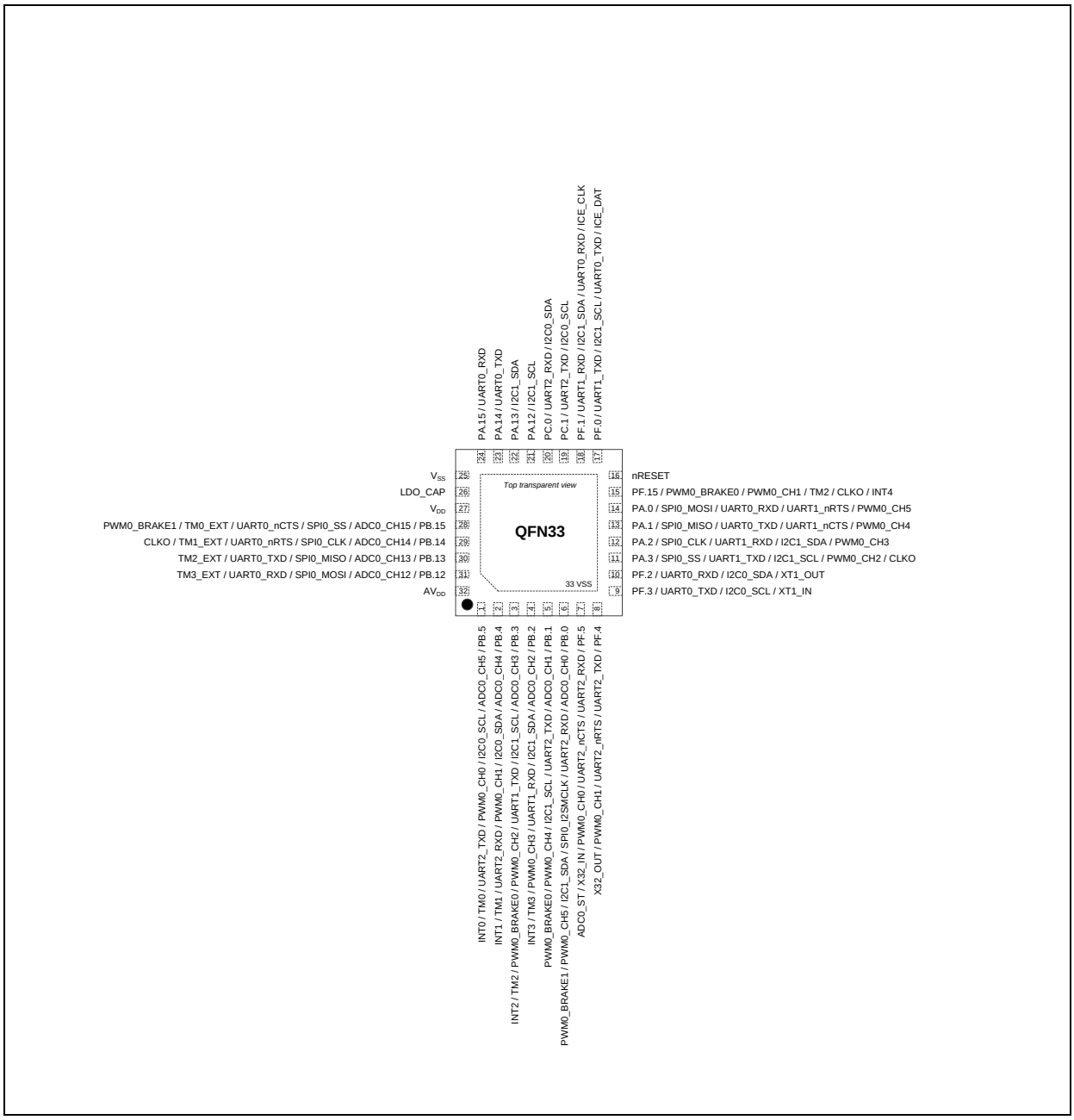


Figure 4.1-12 M031TC1AE Multi-function Pin Diagram

Pin	M031TC1AE Pin Function
1	PB.5 / ADC0_CH5 / I2C0_SCL / PWM0_CH0 / UART2_TXD / TM0 / INT0
2	PB.4 / ADC0_CH4 / I2C0_SDA / PWM0_CH1 / UART2_RXD / TM1 / INT1
3	PB.3 / ADC0_CH3 / I2C1_SCL / UART1_TXD / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2
4	PB.2 / ADC0_CH2 / I2C1_SDA / UART1_RXD / PWM0_CH3 / TM3 / INT3

Pin	M031TC1AE Pin Function
5	PB.1 / ADC0_CH1 / UART2_TXD / I2C1_SCL / PWM0_CH4 / PWM0_BRAKE0
6	PB.0 / ADC0_CH0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / PWM0_CH5 / PWM0_BRAKE1
7	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / X32_IN / ADC0_ST
8	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1 / X32_OUT
9	PF.3 / UART0_TXD / I2C0_SCL / XT1_IN
10	PF.2 / UART0_RXD / I2C0_SDA / XT1_OUT
11	PA.3 / SPI0_SS / UART1_TXD / I2C1_SCL / PWM0_CH2 / CLKO
12	PA.2 / SPI0_CLK / UART1_RXD / I2C1_SDA / PWM0_CH3
13	PA.1 / SPI0_MISO / UART0_TXD / UART1_nCTS / PWM0_CH4
14	PA.0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / PWM0_CH5
15	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / TM2 / CLKO / INT4
16	nRESET
17	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / ICE_DAT
18	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / ICE_CLK
19	PC.1 / UART2_TXD / I2C0_SCL
20	PC.0 / UART2_RXD / I2C0_SDA
21	PA.12 / I2C1_SCL
22	PA.13 / I2C1_SDA
23	PA.14 / UART0_TXD
24	PA.15 / UART0_RXD
25	VSS
26	LDO_CAP
27	V_{DD}
28	PB.15 / ADC0_CH15 / SPI0_SS / UART0_nCTS / TM0_EXT / PWM0_BRAKE1
29	PB.14 / ADC0_CH14 / SPI0_CLK / UART0_nRTS / TM1_EXT / CLKO
30	PB.13 / ADC0_CH13 / SPI0_MISO / UART0_TXD / TM2_EXT
31	PB.12 / ADC0_CH12 / SPI0_MOSI / UART0_RXD / TM3_EXT
32	AV_{DD}

Table 4.1-6 M031TC1AE Multi-function Pin Table

M031TD2AE

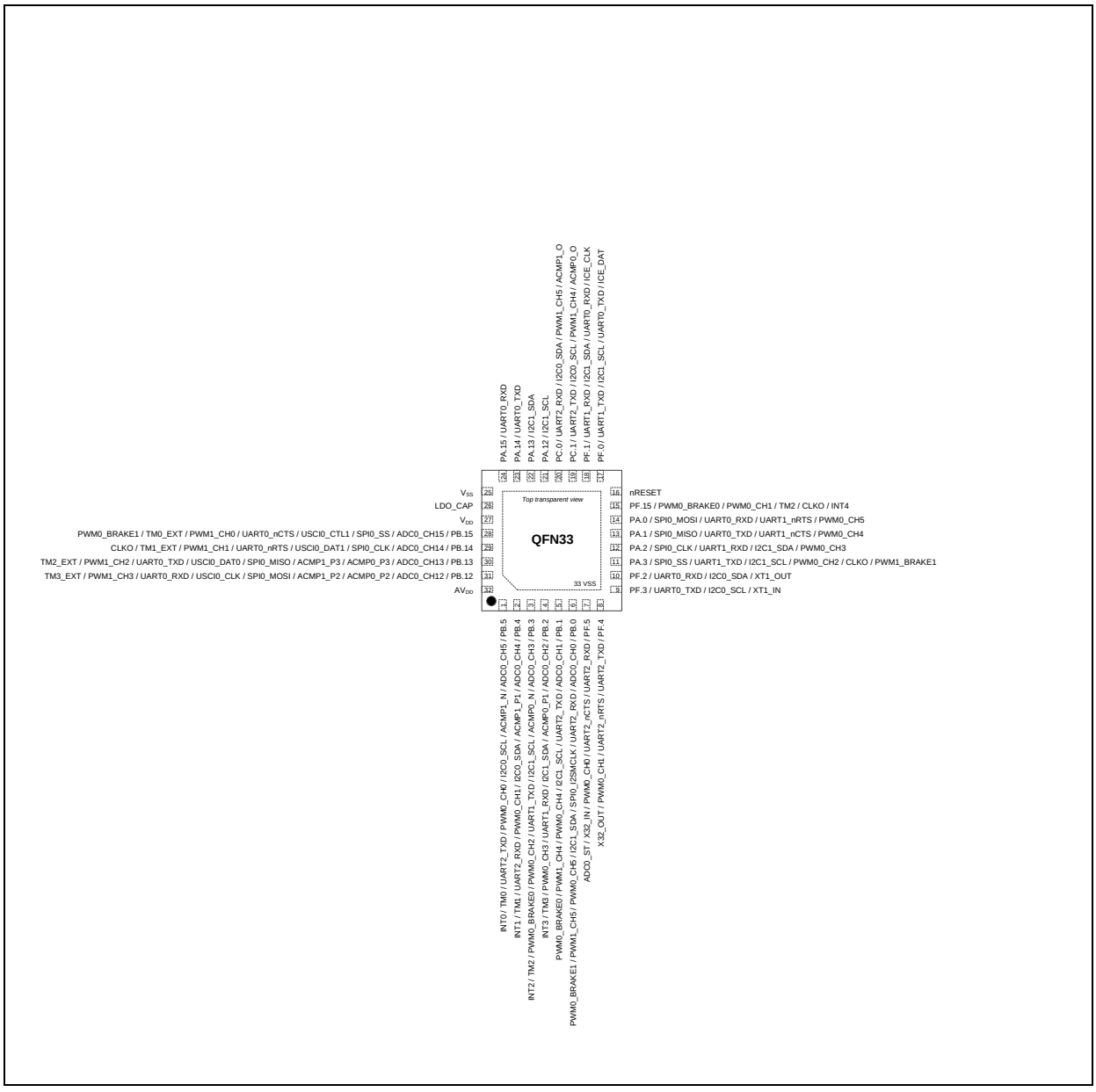


Figure 4.1-13 M031TD2AE Multi-function Pin Diagram

Pin	M031TD2AE Pin Function
1	PB.5 / ADC0_CH5 / ACMP1_N / I2C0_SCL / PWM0_CH0 / UART2_TXD / TM0 / INT0
2	PB.4 / ADC0_CH4 / ACMP1_P1 / I2C0_SDA / PWM0_CH1 / UART2_RXD / TM1 / INT1
3	PB.3 / ADC0_CH3 / ACMP0_N / I2C1_SCL / UART1_TXD / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2
4	PB.2 / ADC0_CH2 / ACMP0_P1 / I2C1_SDA / UART1_RXD / PWM0_CH3 / TM3 / INT3
5	PB.1 / ADC0_CH1 / UART2_TXD / I2C1_SCL / PWM0_CH4 / PWM1_CH4 / PWM0_BRAKE0
6	PB.0 / ADC0_CH0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / PWM0_CH5 / PWM1_CH5 / PWM0_BRAKE1

Pin	M031TD2AE Pin Function
7	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / X32_IN / ADC0_ST
8	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1 / X32_OUT
9	PF.3 / UART0_TXD / I2C0_SCL / XT1_IN
10	PF.2 / UART0_RXD / I2C0_SDA / XT1_OUT
11	PA.3 / SPI0_SS / UART1_TXD / I2C1_SCL / PWM0_CH2 / CLKO / PWM1_BRAKE1
12	PA.2 / SPI0_CLK / UART1_RXD / I2C1_SDA / PWM0_CH3
13	PA.1 / SPI0_MISO / UART0_TXD / UART1_nCTS / PWM0_CH4
14	PA.0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / PWM0_CH5
15	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / TM2 / CLKO / INT4
16	nRESET
17	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / ICE_DAT
18	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / ICE_CLK
19	PC.1 / UART2_TXD / I2C0_SCL / PWM1_CH4 / ACMP0_O
20	PC.0 / UART2_RXD / I2C0_SDA / PWM1_CH5 / ACMP1_O
21	PA.12 / I2C1_SCL
22	PA.13 / I2C1_SDA
23	PA.14 / UART0_TXD
24	PA.15 / UART0_RXD
25	VSS
26	LDO_CAP
27	V_{DD}
28	PB.15 / ADC0_CH15 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / PWM1_CH0 / TM0_EXT / PWM0_BRAKE1
29	PB.14 / ADC0_CH14 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / PWM1_CH1 / TM1_EXT / CLKO
30	PB.13 / ADC0_CH13 / ACMP0_P3 / ACMP1_P3 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / PWM1_CH2 / TM2_EXT
31	PB.12 / ADC0_CH12 / ACMP0_P2 / ACMP1_P2 / SPI0_MOSI / USCI0_CLK / UART0_RXD / PWM1_CH3 / TM3_EXT
32	AV_{DD}

Table 4.1-7 M031TD2AE Multi-function Pin Table

M031TE3AE

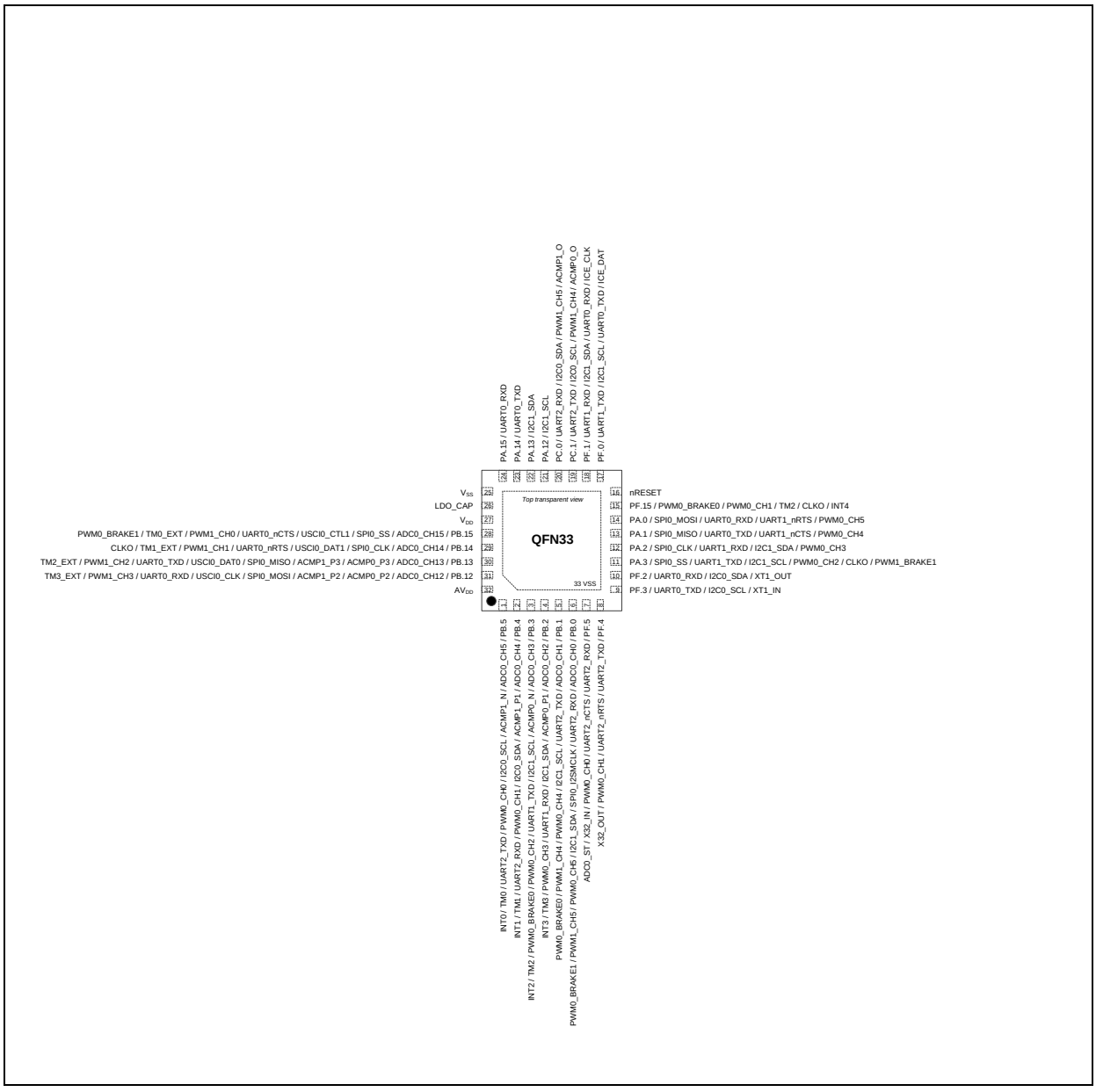


Figure 4.1-14 M031TE3AE Multi-function Pin Diagram

Pin	M031TE3AE Pin Function
1	PB.5 / ADC0_CH5 / ACMP1_N / I2C0_SCL / PWM0_CH0 / UART2_TXD / TM0 / INT0
2	PB.4 / ADC0_CH4 / ACMP1_P1 / I2C0_SDA / PWM0_CH1 / UART2_RXD / TM1 / INT1
3	PB.3 / ADC0_CH3 / ACMP0_N / I2C1_SCL / UART1_TXD / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2
4	PB.2 / ADC0_CH2 / ACMP0_P1 / I2C1_SDA / UART1_RXD / PWM0_CH3 / TM3 / INT3
5	PB.1 / ADC0_CH1 / UART2_TXD / I2C1_SCL / PWM0_CH4 / PWM1_CH4 / PWM0_BRAKE0
6	PB.0 / ADC0_CH0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / PWM0_CH5 / PWM1_CH5 / PWM0_BRAKE1

Pin	M031TE3AE Pin Function
7	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / X32_IN / ADC0_ST
8	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1 / X32_OUT
9	PF.3 / UART0_TXD / I2C0_SCL / XT1_IN
10	PF.2 / UART0_RXD / I2C0_SDA / XT1_OUT
11	PA.3 / SPI0_SS / UART1_TXD / I2C1_SCL / PWM0_CH2 / CLKO / PWM1_BRAKE1
12	PA.2 / SPI0_CLK / UART1_RXD / I2C1_SDA / PWM0_CH3
13	PA.1 / SPI0_MISO / UART0_TXD / UART1_nCTS / PWM0_CH4
14	PA.0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / PWM0_CH5
15	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / TM2 / CLKO / INT4
16	nRESET
17	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / ICE_DAT
18	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / ICE_CLK
19	PC.1 / UART2_TXD / I2C0_SCL / PWM1_CH4 / ACMP0_O
20	PC.0 / UART2_RXD / I2C0_SDA / PWM1_CH5 / ACMP1_O
21	PA.12 / I2C1_SCL
22	PA.13 / I2C1_SDA
23	PA.14 / UART0_TXD
24	PA.15 / UART0_RXD
25	VSS
26	LDO_CAP
27	V_{DD}
28	PB.15 / ADC0_CH15 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / PWM1_CH0 / TM0_EXT / PWM0_BRAKE1
29	PB.14 / ADC0_CH14 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / PWM1_CH1 / TM1_EXT / CLKO
30	PB.13 / ADC0_CH13 / ACMP0_P3 / ACMP1_P3 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / PWM1_CH2 / TM2_EXT
31	PB.12 / ADC0_CH12 / ACMP0_P2 / ACMP1_P2 / SPI0_MOSI / USCI0_CLK / UART0_RXD / PWM1_CH3 / TM3_EXT
32	AV_{DD}

Table 4.1-8 M031TE3AE Multi-function Pin Table

4.1.2.4 M031 Series LQFP 48-Pin Multi-function Pin Diagram

Corresponding Part Number: M031LC2AE, M031LD2AE, M031LE3AE, M031LG6AE, M031LG8AE

M031LC2AE

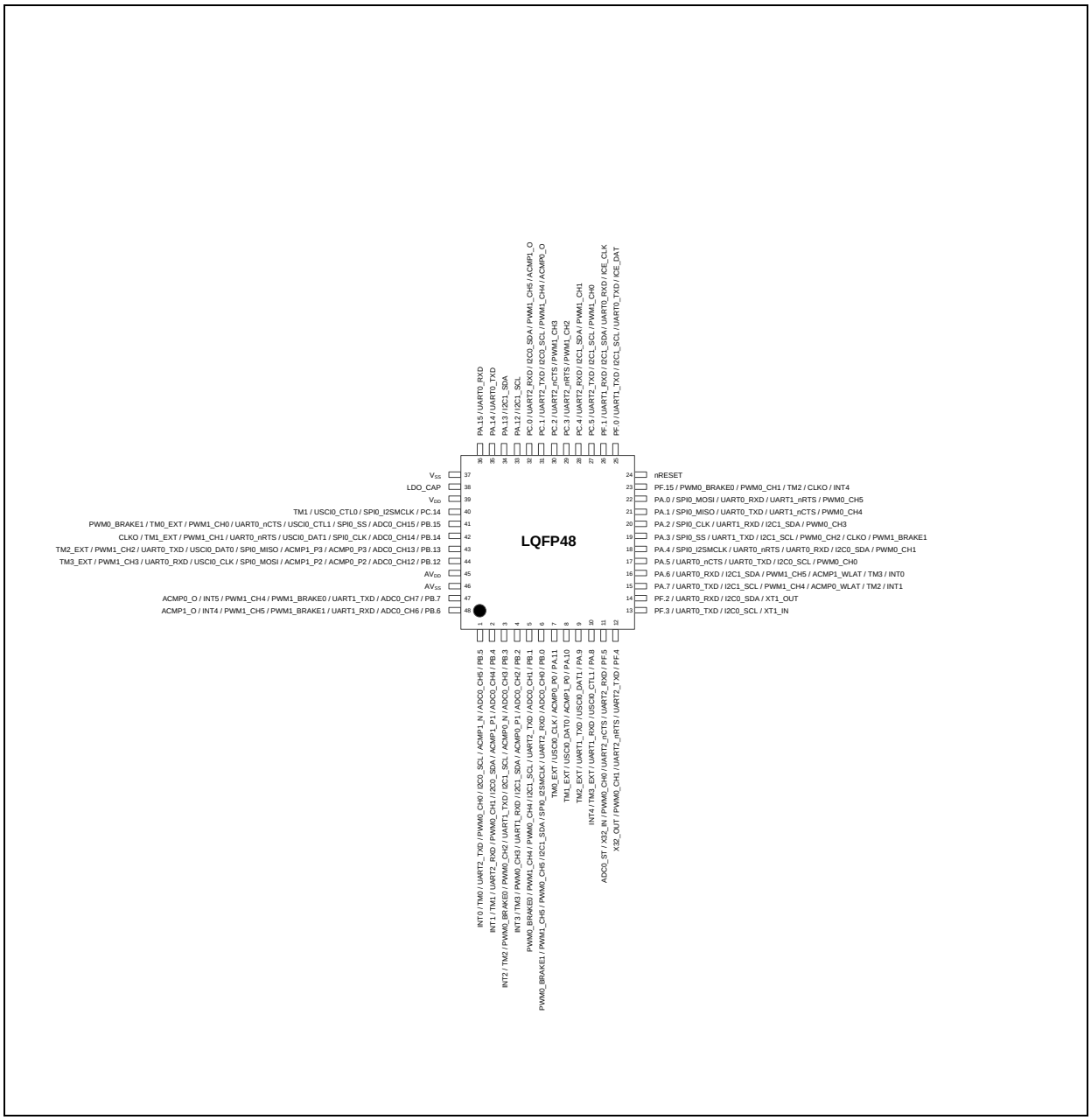


Figure 4.1-15 M031LC2AE Multi-function Pin Diagram

Pin	M031LC2AE Pin Function
1	PB.5 / ADC0_CH5 / ACMP1_N / I2C0_SCL / PWM0_CH0 / UART2_TXD / TM0 / INT0
2	PB.4 / ADC0_CH4 / ACMP1_P1 / I2C0_SDA / PWM0_CH1 / UART2_RXD / TM1 / INT1
3	PB.3 / ADC0_CH3 / ACMP0_N / I2C1_SCL / UART1_TXD / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2

Pin	M031LC2AE Pin Function
4	PB.2 / ADC0_CH2 / ACMP0_P1 / I2C1_SDA / UART1_RXD / PWM0_CH3 / TM3 / INT3
5	PB.1 / ADC0_CH1 / UART2_TXD / I2C1_SCL / PWM0_CH4 / PWM1_CH4 / PWM0_BRAKE0
6	PB.0 / ADC0_CH0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / PWM0_CH5 / PWM1_CH5 / PWM0_BRAKE1
7	PA.11 / ACMP0_P0 / USCI0_CLK / TM0_EXT
8	PA.10 / ACMP1_P0 / USCI0_DAT0 / TM1_EXT
9	PA.9 / USCI0_DAT1 / UART1_TXD / TM2_EXT
10	PA.8 / USCI0_CTL1 / UART1_RXD / TM3_EXT / INT4
11	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / X32_IN / ADC0_ST
12	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1 / X32_OUT
13	PF.3 / UART0_TXD / I2C0_SCL / XT1_IN
14	PF.2 / UART0_RXD / I2C0_SDA / XT1_OUT
15	PA.7 / UART0_TXD / I2C1_SCL / PWM1_CH4 / ACMP0_WLAT / TM2 / INT1
16	PA.6 / UART0_RXD / I2C1_SDA / PWM1_CH5 / ACMP1_WLAT / TM3 / INT0
17	PA.5 / UART0_nCTS / UART0_TXD / I2C0_SCL / PWM0_CH0
18	PA.4 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / PWM0_CH1
19	PA.3 / SPI0_SS / UART1_TXD / I2C1_SCL / PWM0_CH2 / CLKO / PWM1_BRAKE1
20	PA.2 / SPI0_CLK / UART1_RXD / I2C1_SDA / PWM0_CH3
21	PA.1 / SPI0_MISO / UART0_TXD / UART1_nCTS / PWM0_CH4
22	PA.0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / PWM0_CH5
23	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / TM2 / CLKO / INT4
24	nRESET
25	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / ICE_DAT
26	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / ICE_CLK
27	PC.5 / UART2_TXD / I2C1_SCL / PWM1_CH0
28	PC.4 / UART2_RXD / I2C1_SDA / PWM1_CH1
29	PC.3 / UART2_nRTS / PWM1_CH2
30	PC.2 / UART2_nCTS / PWM1_CH3
31	PC.1 / UART2_TXD / I2C0_SCL / PWM1_CH4 / ACMP0_O
32	PC.0 / UART2_RXD / I2C0_SDA / PWM1_CH5 / ACMP1_O
33	PA.12 / I2C1_SCL
34	PA.13 / I2C1_SDA
35	PA.14 / UART0_TXD
36	PA.15 / UART0_RXD
37	VSS

Pin	M031LC2AE Pin Function
38	LDO_CAP
39	V_{DD}
40	PC.14 / SPI0_I2SMCLK / USCI0_CTL0 / TM1
41	PB.15 / ADC0_CH15 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / PWM1_CH0 / TM0_EXT / PWM0_BRAKE1
42	PB.14 / ADC0_CH14 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / PWM1_CH1 / TM1_EXT / CLKO
43	PB.13 / ADC0_CH13 / ACMP0_P3 / ACMP1_P3 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / PWM1_CH2 / TM2_EXT
44	PB.12 / ADC0_CH12 / ACMP0_P2 / ACMP1_P2 / SPI0_MOSI / USCI0_CLK / UART0_RXD / PWM1_CH3 / TM3_EXT
45	AV_{DD}
46	AVSS
47	PB.7 / ADC0_CH7 / UART1_TXD / PWM1_BRAKE0 / PWM1_CH4 / INT5 / ACMP0_O
48	PB.6 / ADC0_CH6 / UART1_RXD / PWM1_BRAKE1 / PWM1_CH5 / INT4 / ACMP1_O

Table 4.1-9 M031LC2AE Multi-function Pin Table

M031LD2AE

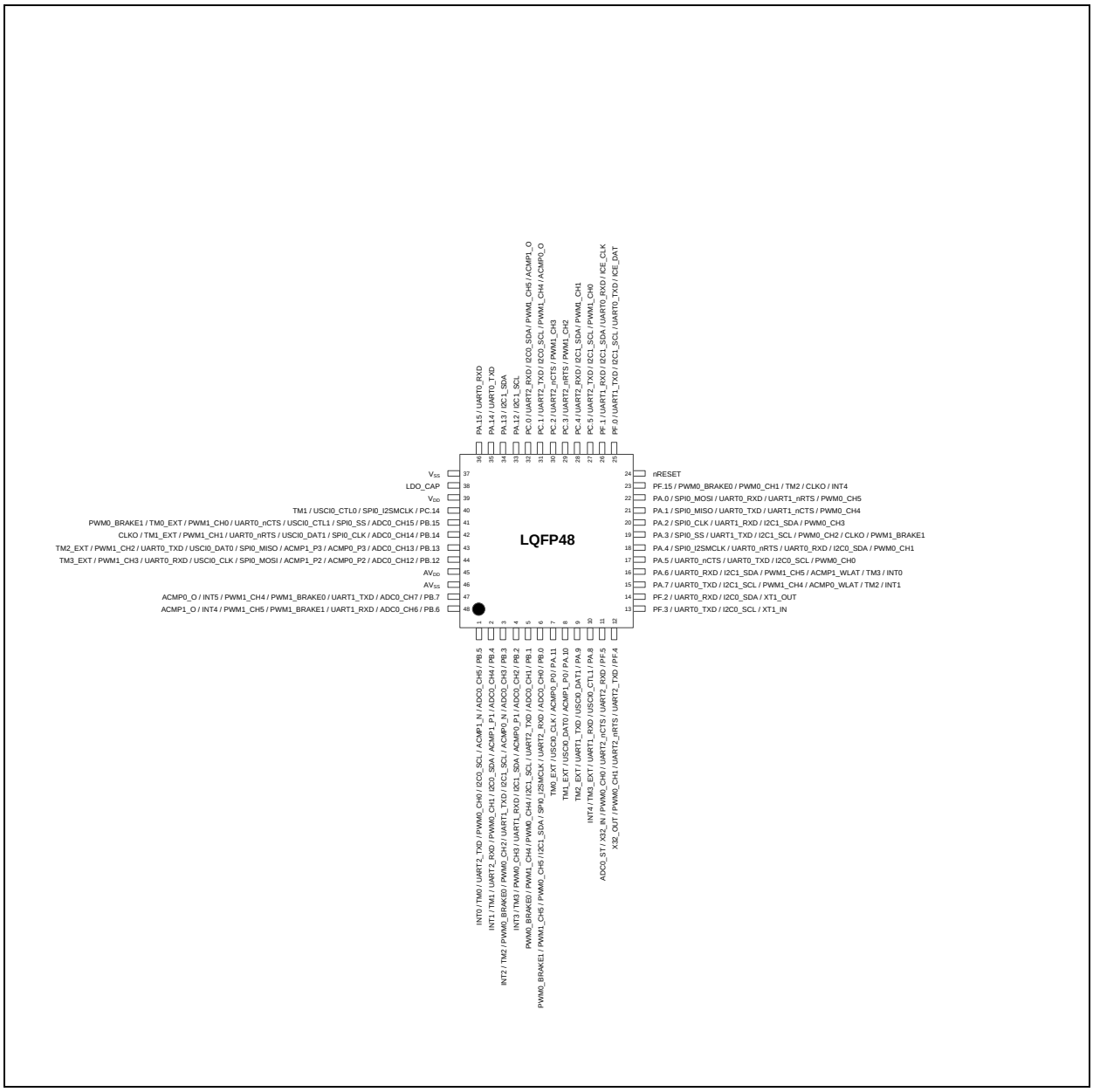


Figure 4.1-16 M031LD2AE Multi-function Pin Diagram

Pin	M031LD2AE Pin Function
1	PB.5 / ADC0_CH5 / ACMP1_N / I2C0_SCL / PWM0_CH0 / UART2_TXD / TM0 / INT0
2	PB.4 / ADC0_CH4 / ACMP1_P1 / I2C0_SDA / PWM0_CH1 / UART2_RXD / TM1 / INT1
3	PB.3 / ADC0_CH3 / ACMP0_N / I2C1_SCL / UART1_TXD / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2
4	PB.2 / ADC0_CH2 / ACMP0_P1 / I2C1_SDA / UART1_RXD / PWM0_CH3 / TM3 / INT3
5	PB.1 / ADC0_CH1 / UART2_TXD / I2C1_SCL / PWM0_CH4 / PWM1_CH4 / PWM0_BRAKE0
6	PB.0 / ADC0_CH0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / PWM0_CH5 / PWM1_CH5 / PWM0_BRAKE1

Pin	M031LD2AE Pin Function
7	PA.11 / ACMP0_P0 / USCI0_CLK / TM0_EXT
8	PA.10 / ACMP1_P0 / USCI0_DAT0 / TM1_EXT
9	PA.9 / USCI0_DAT1 / UART1_TXD / TM2_EXT
10	PA.8 / USCI0_CTL1 / UART1_RXD / TM3_EXT / INT4
11	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / X32_IN / ADC0_ST
12	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1 / X32_OUT
13	PF.3 / UART0_TXD / I2C0_SCL / XT1_IN
14	PF.2 / UART0_RXD / I2C0_SDA / XT1_OUT
15	PA.7 / UART0_TXD / I2C1_SCL / PWM1_CH4 / ACMP0_WLAT / TM2 / INT1
16	PA.6 / UART0_RXD / I2C1_SDA / PWM1_CH5 / ACMP1_WLAT / TM3 / INT0
17	PA.5 / UART0_nCTS / UART0_TXD / I2C0_SCL / PWM0_CH0
18	PA.4 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / PWM0_CH1
19	PA.3 / SPI0_SS / UART1_TXD / I2C1_SCL / PWM0_CH2 / CLKO / PWM1_BRAKE1
20	PA.2 / SPI0_CLK / UART1_RXD / I2C1_SDA / PWM0_CH3
21	PA.1 / SPI0_MISO / UART0_TXD / UART1_nCTS / PWM0_CH4
22	PA.0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / PWM0_CH5
23	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / TM2 / CLKO / INT4
24	nRESET
25	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / ICE_DAT
26	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / ICE_CLK
27	PC.5 / UART2_TXD / I2C1_SCL / PWM1_CH0
28	PC.4 / UART2_RXD / I2C1_SDA / PWM1_CH1
29	PC.3 / UART2_nRTS / PWM1_CH2
30	PC.2 / UART2_nCTS / PWM1_CH3
31	PC.1 / UART2_TXD / I2C0_SCL / PWM1_CH4 / ACMP0_O
32	PC.0 / UART2_RXD / I2C0_SDA / PWM1_CH5 / ACMP1_O
33	PA.12 / I2C1_SCL
34	PA.13 / I2C1_SDA
35	PA.14 / UART0_TXD
36	PA.15 / UART0_RXD
37	VSS
38	LDO_CAP
39	V_{DD}
40	PC.14 / SPI0_I2SMCLK / USCI0_CTL0 / TM1

Pin	M031LD2AE Pin Function
41	PB.15 / ADC0_CH15 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / PWM1_CH0 / TM0_EXT / PWM0_BRAKE1
42	PB.14 / ADC0_CH14 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / PWM1_CH1 / TM1_EXT / CLKO
43	PB.13 / ADC0_CH13 / ACMP0_P3 / ACMP1_P3 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / PWM1_CH2 / TM2_EXT
44	PB.12 / ADC0_CH12 / ACMP0_P2 / ACMP1_P2 / SPI0_MOSI / USCI0_CLK / UART0_RXD / PWM1_CH3 / TM3_EXT
45	AV_{DD}
46	AVSS
47	PB.7 / ADC0_CH7 / UART1_TXD / PWM1_BRAKE0 / PWM1_CH4 / INT5 / ACMP0_O
48	PB.6 / ADC0_CH6 / UART1_RXD / PWM1_BRAKE1 / PWM1_CH5 / INT4 / ACMP1_O

Table 4.1-10 M031LD2AE Multi-function Pin Table

M031LE3AE

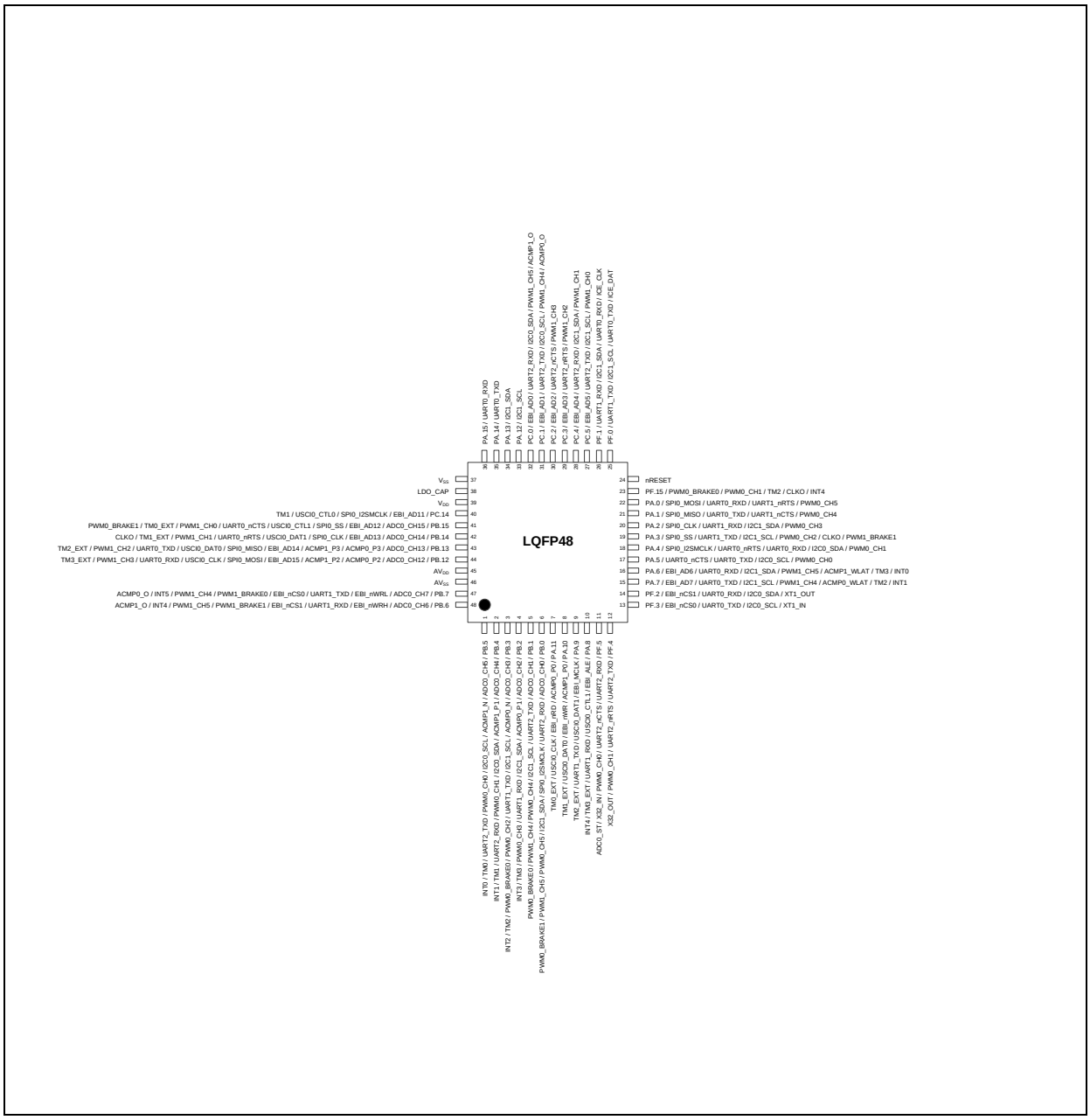


Figure 4.1-17 M031LE3AE Multi-function Pin Diagram

Pin	M031LE3AE Pin Function
1	PB.5 / ADC0_CH5 / ACMP1_N / I2C0_SCL / PWM0_CH0 / UART2_TXD / TM0 / INT0
2	PB.4 / ADC0_CH4 / ACMP1_P1 / I2C0_SDA / PWM0_CH1 / UART2_RXD / TM1 / INT1
3	PB.3 / ADC0_CH3 / ACMP0_N / I2C1_SCL / UART1_TXD / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2
4	PB.2 / ADC0_CH2 / ACMP0_P1 / I2C1_SDA / UART1_RXD / PWM0_CH3 / TM3 / INT3
5	PB.1 / ADC0_CH1 / UART2_TXD / I2C1_SCL / PWM0_CH4 / PWM1_CH4 / PWM0_BRAKE0
6	PB.0 / ADC0_CH0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / PWM0_CH5 / PWM1_CH5 / PWM0_BRAKE1
7	PA.11 / ACMP0_P0 / EBI_nRD / USCI0_CLK / TM0_EXT
8	PA.10 / ACMP1_P0 / EBI_nWR / USCI0_DAT0 / TM1_EXT
9	PA.9 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / TM2_EXT
10	PA.8 / EBI_ALE / USCI0_CTL1 / UART1_RXD / TM3_EXT / INT4
11	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / X32_IN / ADC0_ST
12	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1 / X32_OUT
13	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN
14	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / XT1_OUT
15	PA.7 / EBI_AD7 / UART0_TXD / I2C1_SCL / PWM1_CH4 / ACMP0_WLAT / TM2 / INT1
16	PA.6 / EBI_AD6 / UART0_RXD / I2C1_SDA / PWM1_CH5 / ACMP1_WLAT / TM3 / INT0
17	PA.5 / UART0_nCTS / UART0_TXD / I2C0_SCL / PWM0_CH0
18	PA.4 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / PWM0_CH1
19	PA.3 / SPI0_SS / UART1_TXD / I2C1_SCL / PWM0_CH2 / CLKO / PWM1_BRAKE1
20	PA.2 / SPI0_CLK / UART1_RXD / I2C1_SDA / PWM0_CH3
21	PA.1 / SPI0_MISO / UART0_TXD / UART1_nCTS / PWM0_CH4
22	PA.0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / PWM0_CH5
23	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / TM2 / CLKO / INT4
24	nRESET
25	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / ICE_DAT
26	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / ICE_CLK
27	PC.5 / EBI_AD5 / UART2_TXD / I2C1_SCL / PWM1_CH0
28	PC.4 / EBI_AD4 / UART2_RXD / I2C1_SDA / PWM1_CH1
29	PC.3 / EBI_AD3 / UART2_nRTS / PWM1_CH2
30	PC.2 / EBI_AD2 / UART2_nCTS / PWM1_CH3
31	PC.1 / EBI_AD1 / UART2_TXD / I2C0_SCL / PWM1_CH4 / ACMP0_O
32	PC.0 / EBI_AD0 / UART2_RXD / I2C0_SDA / PWM1_CH5 / ACMP1_O
33	PA.12 / I2C1_SCL

Pin	M031LE3AE Pin Function
34	PA.13 / I2C1_SDA
35	PA.14 / UART0_TXD
36	PA.15 / UART0_RXD
37	VSS
38	LDO_CAP
39	V_{DD}
40	PC.14 / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / TM1
41	PB.15 / ADC0_CH15 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / PWM1_CH0 / TM0_EXT / PWM0_BRAKE1
42	PB.14 / ADC0_CH14 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / PWM1_CH1 / TM1_EXT / CLKO
43	PB.13 / ADC0_CH13 / ACMP0_P3 / ACMP1_P3 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / PWM1_CH2 / TM2_EXT
44	PB.12 / ADC0_CH12 / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / PWM1_CH3 / TM3_EXT
45	AV_{DD}
46	AVSS
47	PB.7 / ADC0_CH7 / EBI_nWRL / UART1_TXD / EBI_nCS0 / PWM1_BRAKE0 / PWM1_CH4 / INT5 / ACMP0_O
48	PB.6 / ADC0_CH6 / EBI_nWRH / UART1_RXD / EBI_nCS1 / PWM1_BRAKE1 / PWM1_CH5 / INT4 / ACMP1_O

Table 4.1-11 M031LE3AE Multi-function Pin Table

M031LG6AE

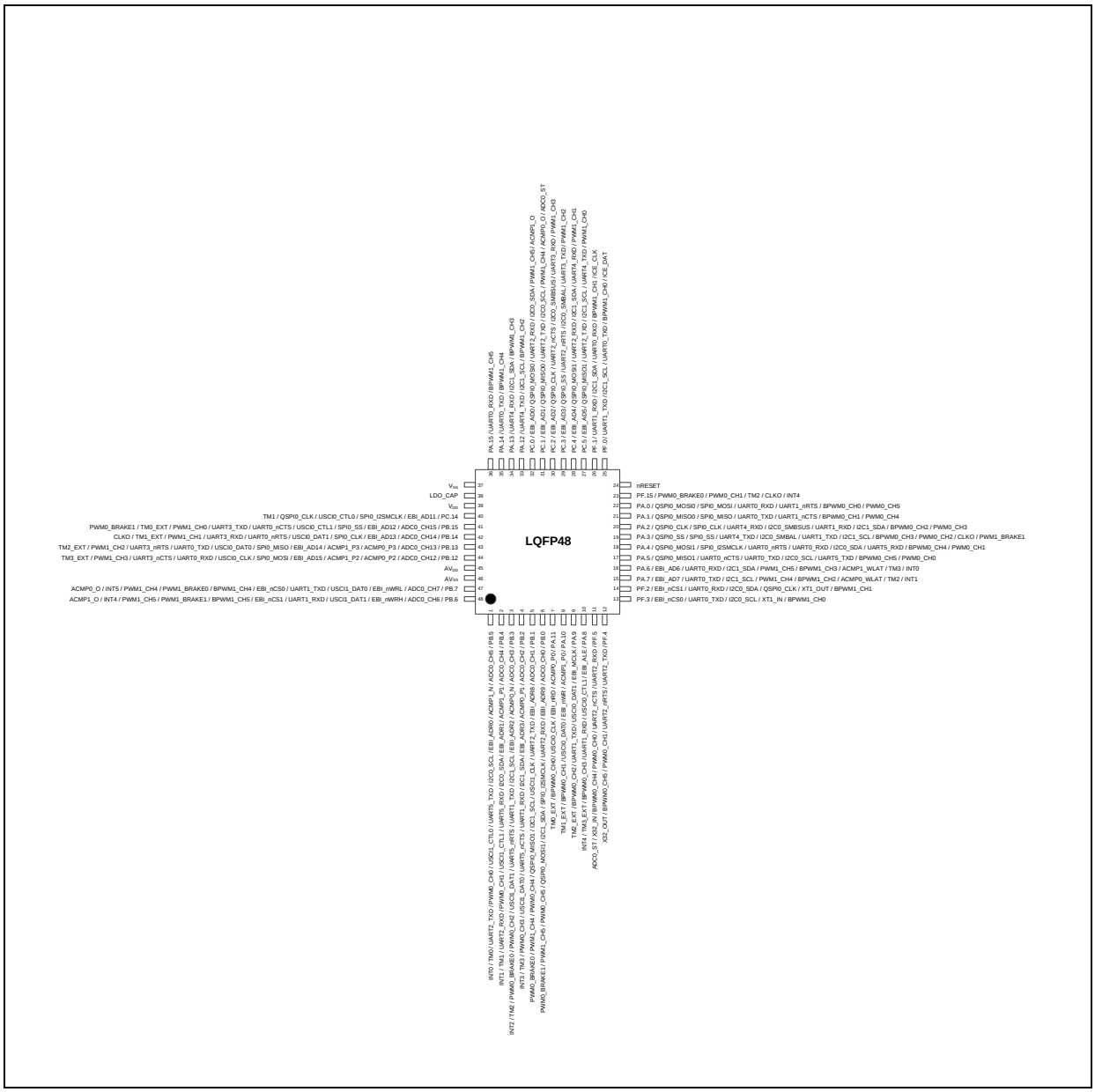


Figure 4.1-18 M031LG6AE Multi-function Pin Diagram

Pin	M031LG6AE Pin Function
1	PB.5 / ADC0_CH5 / ACMP1_N / EBI_ADR0 / I2C0_SCL / UART5_TXD / USCI1_CTL0 / PWM0_CH0 / UART2_TXD / TM0 / INT0
2	PB.4 / ADC0_CH4 / ACMP1_P1 / EBI_ADR1 / I2C0_SDA / UART5_RXD / USCI1_CTL1 / PWM0_CH1 / UART2_RXD / TM1 / INT1
3	PB.3 / ADC0_CH3 / ACMP0_N / EBI_ADR2 / I2C1_SCL / UART1_TXD / UART5_nRTS / USCI1_DAT1 / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2
4	PB.2 / ADC0_CH2 / ACMP0_P1 / EBI_ADR3 / I2C1_SDA / UART1_RXD / UART5_nCTS / USCI1_DAT0 /

Pin	M031LG6AE Pin Function
	PWM0_CH3 / TM3 / INT3
5	PB.1 / ADC0_CH1 / EBI_ADR8 / UART2_TXD / USCI1_CLK / I2C1_SCL / QSPI0_MISO1 / PWM0_CH4 / PWM1_CH4 / PWM0_BRAKE0
6	PB.0 / ADC0_CH0 / EBI_ADR9 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / PWM0_CH5 / PWM1_CH5 / PWM0_BRAKE1
7	PA.11 / ACMP0_P0 / EBI_nRD / USCI0_CLK / BPWM0_CH0 / TM0_EXT
8	PA.10 / ACMP1_P0 / EBI_nWR / USCI0_DAT0 / BPWM0_CH1 / TM1_EXT
9	PA.9 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / BPWM0_CH2 / TM2_EXT
10	PA.8 / EBI_ALE / USCI0_CTL1 / UART1_RXD / BPWM0_CH3 / TM3_EXT / INT4
11	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / BPWM0_CH4 / X32_IN / ADC0_ST
12	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1 / BPWM0_CH5 / X32_OUT
13	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / BPWM1_CH0
14	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / BPWM1_CH1
15	PA.7 / EBI_AD7 / UART0_TXD / I2C1_SCL / PWM1_CH4 / BPWM1_CH2 / ACMP0_WLAT / TM2 / INT1
16	PA.6 / EBI_AD6 / UART0_RXD / I2C1_SDA / PWM1_CH5 / BPWM1_CH3 / ACMP1_WLAT / TM3 / INT0
17	PA.5 / QSPI0_MISO1 / UART0_nCTS / UART0_TXD / I2C0_SCL / UART5_TXD / BPWM0_CH5 / PWM0_CH0
18	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / UART5_RXD / BPWM0_CH4 / PWM0_CH1
19	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / I2C0_SMBAL / UART1_TXD / I2C1_SCL / BPWM0_CH3 / PWM0_CH2 / CLKO / PWM1_BRAKE1
20	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / BPWM0_CH2 / PWM0_CH3
21	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / BPWM0_CH1 / PWM0_CH4
22	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / BPWM0_CH0 / PWM0_CH5
23	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / TM2 / CLKO / INT4
24	nRESET
25	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / BPWM1_CH0 / ICE_DAT
26	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / BPWM1_CH1 / ICE_CLK
27	PC.5 / EBI_AD5 / QSPI0_MISO1 / UART2_TXD / I2C1_SCL / UART4_TXD / PWM1_CH0
28	PC.4 / EBI_AD4 / QSPI0_MOSI1 / UART2_RXD / I2C1_SDA / UART4_RXD / PWM1_CH1
29	PC.3 / EBI_AD3 / QSPI0_SS / UART2_nRTS / I2C0_SMBAL / UART3_TXD / PWM1_CH2
30	PC.2 / EBI_AD2 / QSPI0_CLK / UART2_nCTS / I2C0_SMBSUS / UART3_RXD / PWM1_CH3
31	PC.1 / EBI_AD1 / QSPI0_MISO0 / UART2_TXD / I2C0_SCL / PWM1_CH4 / ACMP0_O / ADC0_ST
32	PC.0 / EBI_AD0 / QSPI0_MOSI0 / UART2_RXD / I2C0_SDA / PWM1_CH5 / ACMP1_O
33	PA.12 / UART4_TXD / I2C1_SCL / BPWM1_CH2
34	PA.13 / UART4_RXD / I2C1_SDA / BPWM1_CH3
35	PA.14 / UART0_TXD / BPWM1_CH4

Pin	M031LG6AE Pin Function
36	PA.15 / UART0_RXD / BPWM1_CH5
37	VSS
38	LDO_CAP
39	V_{DD}
40	PC.14 / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / TM1
41	PB.15 / ADC0_CH15 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / PWM1_CH0 / TM0_EXT / PWM0_BRAKE1
42	PB.14 / ADC0_CH14 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / PWM1_CH1 / TM1_EXT / CLKO
43	PB.13 / ADC0_CH13 / ACMP0_P3 / ACMP1_P3 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / PWM1_CH2 / TM2_EXT
44	PB.12 / ADC0_CH12 / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / PWM1_CH3 / TM3_EXT
45	AV_{DD}
46	AVSS
47	PB.7 / ADC0_CH7 / EBI_nWRL / USCI1_DAT0 / UART1_TXD / EBI_nCS0 / BPWM1_CH4 / PWM1_BRAKE0 / PWM1_CH4 / INT5 / ACMP0_O
48	PB.6 / ADC0_CH6 / EBI_nWRH / USCI1_DAT1 / UART1_RXD / EBI_nCS1 / BPWM1_CH5 / PWM1_BRAKE1 / PWM1_CH5 / INT4 / ACMP1_O

Table 4.1-12 M031LG6AE Multi-function Pin Table

M031LG8AE

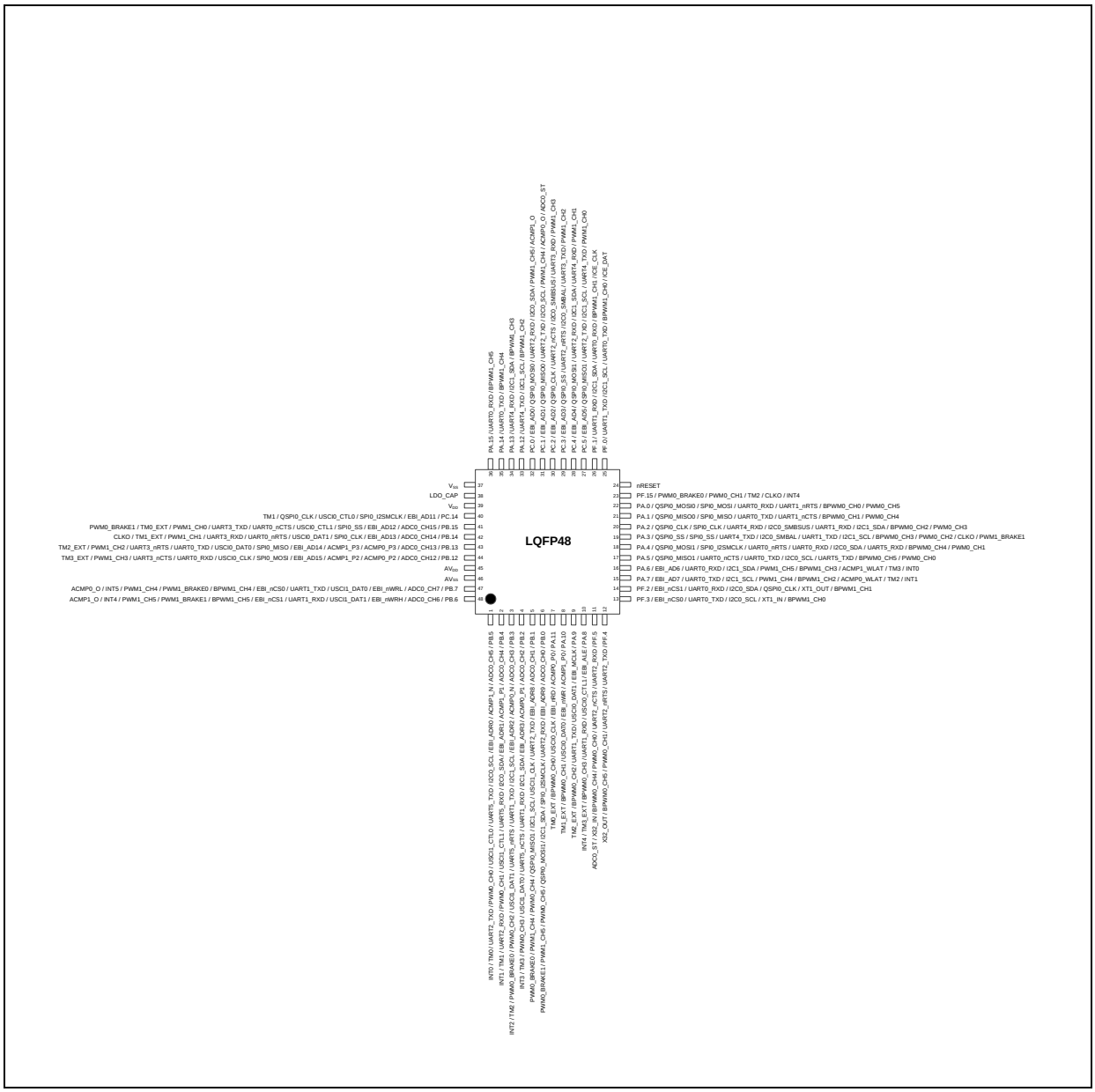


Figure 4.1-19 M031LG8AE Multi-function Pin Diagram

Pin	M031LG8AE Pin Function
1	PB.5 / ADC0_CH5 / ACMP1_N / EBI_ADR0 / I2C0_SCL / UART5_TXD / USCI1_CTL0 / PWM0_CH0 / UART2_TXD / TM0 / INT0
2	PB.4 / ADC0_CH4 / ACMP1_P1 / EBI_ADR1 / I2C0_SDA / UART5_RXD / USCI1_CTL1 / PWM0_CH1 / UART2_RXD / TM1 / INT1
3	PB.3 / ADC0_CH3 / ACMP0_N / EBI_ADR2 / I2C1_SCL / UART1_TXD / UART5_nRTS / USCI1_DAT1 / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2
4	PB.2 / ADC0_CH2 / ACMP0_P1 / EBI_ADR3 / I2C1_SDA / UART1_RXD / UART5_nCTS / USCI1_DAT0 /

Pin	M031LG8AE Pin Function
	PWM0_CH3 / TM3 / INT3
5	PB.1 / ADC0_CH1 / EBI_ADR8 / UART2_TXD / USCI1_CLK / I2C1_SCL / QSPI0_MISO1 / PWM0_CH4 / PWM1_CH4 / PWM0_BRAKE0
6	PB.0 / ADC0_CH0 / EBI_ADR9 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / PWM0_CH5 / PWM1_CH5 / PWM0_BRAKE1
7	PA.11 / ACMP0_P0 / EBI_nRD / USCI0_CLK / BPWM0_CH0 / TM0_EXT
8	PA.10 / ACMP1_P0 / EBI_nWR / USCI0_DAT0 / BPWM0_CH1 / TM1_EXT
9	PA.9 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / BPWM0_CH2 / TM2_EXT
10	PA.8 / EBI_ALE / USCI0_CTL1 / UART1_RXD / BPWM0_CH3 / TM3_EXT / INT4
11	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / BPWM0_CH4 / X32_IN / ADC0_ST
12	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1 / BPWM0_CH5 / X32_OUT
13	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / BPWM1_CH0
14	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / BPWM1_CH1
15	PA.7 / EBI_AD7 / UART0_TXD / I2C1_SCL / PWM1_CH4 / BPWM1_CH2 / ACMP0_WLAT / TM2 / INT1
16	PA.6 / EBI_AD6 / UART0_RXD / I2C1_SDA / PWM1_CH5 / BPWM1_CH3 / ACMP1_WLAT / TM3 / INT0
17	PA.5 / QSPI0_MISO1 / UART0_nCTS / UART0_TXD / I2C0_SCL / UART5_TXD / BPWM0_CH5 / PWM0_CH0
18	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / UART5_RXD / BPWM0_CH4 / PWM0_CH1
19	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / I2C0_SMBAL / UART1_TXD / I2C1_SCL / BPWM0_CH3 / PWM0_CH2 / CLKO / PWM1_BRAKE1
20	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / BPWM0_CH2 / PWM0_CH3
21	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / BPWM0_CH1 / PWM0_CH4
22	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / BPWM0_CH0 / PWM0_CH5
23	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / TM2 / CLKO / INT4
24	nRESET
25	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / BPWM1_CH0 / ICE_DAT
26	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / BPWM1_CH1 / ICE_CLK
27	PC.5 / EBI_AD5 / QSPI0_MISO1 / UART2_TXD / I2C1_SCL / UART4_TXD / PWM1_CH0
28	PC.4 / EBI_AD4 / QSPI0_MOSI1 / UART2_RXD / I2C1_SDA / UART4_RXD / PWM1_CH1
29	PC.3 / EBI_AD3 / QSPI0_SS / UART2_nRTS / I2C0_SMBAL / UART3_TXD / PWM1_CH2
30	PC.2 / EBI_AD2 / QSPI0_CLK / UART2_nCTS / I2C0_SMBSUS / UART3_RXD / PWM1_CH3
31	PC.1 / EBI_AD1 / QSPI0_MISO0 / UART2_TXD / I2C0_SCL / PWM1_CH4 / ACMP0_O / ADC0_ST
32	PC.0 / EBI_AD0 / QSPI0_MOSI0 / UART2_RXD / I2C0_SDA / PWM1_CH5 / ACMP1_O
33	PA.12 / UART4_TXD / I2C1_SCL / BPWM1_CH2
34	PA.13 / UART4_RXD / I2C1_SDA / BPWM1_CH3
35	PA.14 / UART0_TXD / BPWM1_CH4

Pin	M031LG8AE Pin Function
36	PA.15 / UART0_RXD / BPWM1_CH5
37	VSS
38	LDO_CAP
39	V_{DD}
40	PC.14 / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / TM1
41	PB.15 / ADC0_CH15 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / PWM1_CH0 / TM0_EXT / PWM0_BRAKE1
42	PB.14 / ADC0_CH14 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / PWM1_CH1 / TM1_EXT / CLKO
43	PB.13 / ADC0_CH13 / ACMP0_P3 / ACMP1_P3 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / PWM1_CH2 / TM2_EXT
44	PB.12 / ADC0_CH12 / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / PWM1_CH3 / TM3_EXT
45	AV_{DD}
46	AVSS
47	PB.7 / ADC0_CH7 / EBI_nWRL / USCI1_DAT0 / UART1_TXD / EBI_nCS0 / BPWM1_CH4 / PWM1_BRAKE0 / PWM1_CH4 / INT5 / ACMP0_O
48	PB.6 / ADC0_CH6 / EBI_nWRH / USCI1_DAT1 / UART1_RXD / EBI_nCS1 / BPWM1_CH5 / PWM1_BRAKE1 / PWM1_CH5 / INT4 / ACMP1_O

Table 4.1-13 M031LG8AE Multi-function Pin Table

4.1.2.5 M031 Series LQFP 64-Pin Multi-function Pin Diagram

Corresponding Part Number: M031SC2AE, M031SD2AE, M031SE3AE, M031SG6AE, M031SG8AE, M031SIAAE

M031SC2AE

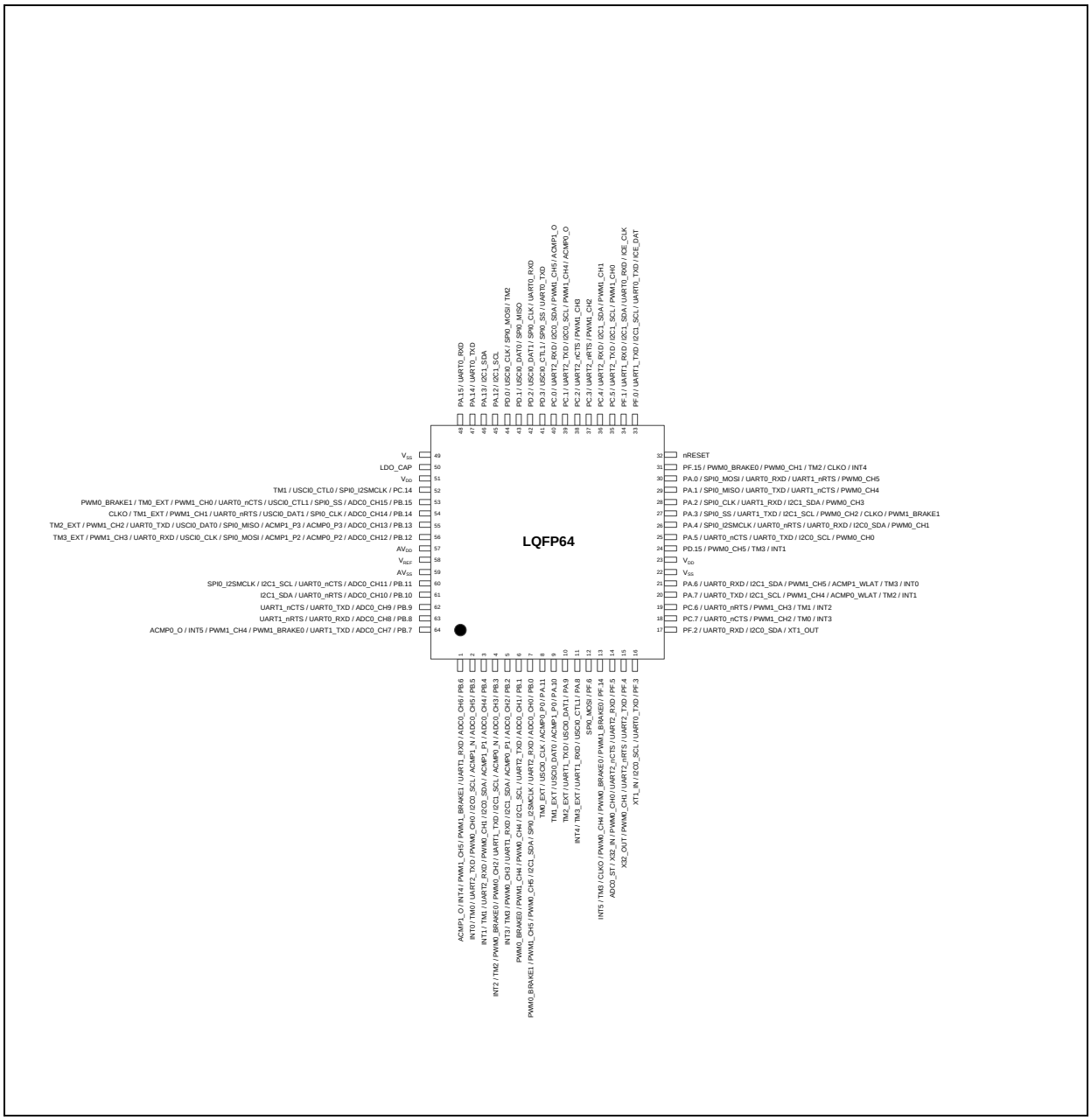


Figure 4.1-20 M031SC2AE Multi-function Pin Diagram

Pin	M031SC2AE Pin Function
1	PB.6 / ADC0_CH6 / UART1_RXD / PWM1_BRAKE1 / PWM1_CH5 / INT4 / ACMP1_O
2	PB.5 / ADC0_CH5 / ACMP1_N / I2C0_SCL / PWM0_CH0 / UART2_TXD / TM0 / INT0
3	PB.4 / ADC0_CH4 / ACMP1_P1 / I2C0_SDA / PWM0_CH1 / UART2_RXD / TM1 / INT1

Pin	M031SC2AE Pin Function
4	PB.3 / ADC0_CH3 / ACMP0_N / I2C1_SCL / UART1_TXD / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2
5	PB.2 / ADC0_CH2 / ACMP0_P1 / I2C1_SDA / UART1_RXD / PWM0_CH3 / TM3 / INT3
6	PB.1 / ADC0_CH1 / UART2_TXD / I2C1_SCL / PWM0_CH4 / PWM1_CH4 / PWM0_BRAKE0
7	PB.0 / ADC0_CH0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / PWM0_CH5 / PWM1_CH5 / PWM0_BRAKE1
8	PA.11 / ACMP0_P0 / USCI0_CLK / TM0_EXT
9	PA.10 / ACMP1_P0 / USCI0_DAT0 / TM1_EXT
10	PA.9 / USCI0_DAT1 / UART1_TXD / TM2_EXT
11	PA.8 / USCI0_CTL1 / UART1_RXD / TM3_EXT / INT4
12	PF.6 / SPI0_MOSI
13	PF.14 / PWM1_BRAKE0 / PWM0_BRAKE0 / PWM0_CH4 / CLKO / TM3 / INT5
14	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / X32_IN / ADC0_ST
15	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1 / X32_OUT
16	PF.3 / UART0_TXD / I2C0_SCL / XT1_IN
17	PF.2 / UART0_RXD / I2C0_SDA / XT1_OUT
18	PC.7 / UART0_nCTS / PWM1_CH2 / TM0 / INT3
19	PC.6 / UART0_nRTS / PWM1_CH3 / TM1 / INT2
20	PA.7 / UART0_TXD / I2C1_SCL / PWM1_CH4 / ACMP0_WLAT / TM2 / INT1
21	PA.6 / UART0_RXD / I2C1_SDA / PWM1_CH5 / ACMP1_WLAT / TM3 / INT0
22	VSS
23	V_{DD}
24	PD.15 / PWM0_CH5 / TM3 / INT1
25	PA.5 / UART0_nCTS / UART0_TXD / I2C0_SCL / PWM0_CH0
26	PA.4 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / PWM0_CH1
27	PA.3 / SPI0_SS / UART1_TXD / I2C1_SCL / PWM0_CH2 / CLKO / PWM1_BRAKE1
28	PA.2 / SPI0_CLK / UART1_RXD / I2C1_SDA / PWM0_CH3
29	PA.1 / SPI0_MISO / UART0_TXD / UART1_nCTS / PWM0_CH4
30	PA.0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / PWM0_CH5
31	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / TM2 / CLKO / INT4
32	nRESET
33	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / ICE_DAT
34	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / ICE_CLK
35	PC.5 / UART2_TXD / I2C1_SCL / PWM1_CH0
36	PC.4 / UART2_RXD / I2C1_SDA / PWM1_CH1
37	PC.3 / UART2_nRTS / PWM1_CH2

Pin	M031SC2AE Pin Function
38	PC.2 / UART2_nCTS / PWM1_CH3
39	PC.1 / UART2_TXD / I2C0_SCL / PWM1_CH4 / ACMP0_O
40	PC.0 / UART2_RXD / I2C0_SDA / PWM1_CH5 / ACMP1_O
41	PD.3 / USCI0_CTL1 / SPI0_SS / UART0_TXD
42	PD.2 / USCI0_DAT1 / SPI0_CLK / UART0_RXD
43	PD.1 / USCI0_DAT0 / SPI0_MISO
44	PD.0 / USCI0_CLK / SPI0_MOSI / TM2
45	PA.12 / I2C1_SCL
46	PA.13 / I2C1_SDA
47	PA.14 / UART0_TXD
48	PA.15 / UART0_RXD
49	VSS
50	LDO_CAP
51	V_{DD}
52	PC.14 / SPI0_I2SMCLK / USCI0_CTL0 / TM1
53	PB.15 / ADC0_CH15 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / PWM1_CH0 / TM0_EXT / PWM0_BRAKE1
54	PB.14 / ADC0_CH14 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / PWM1_CH1 / TM1_EXT / CLKO
55	PB.13 / ADC0_CH13 / ACMP0_P3 / ACMP1_P3 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / PWM1_CH2 / TM2_EXT
56	PB.12 / ADC0_CH12 / ACMP0_P2 / ACMP1_P2 / SPI0_MOSI / USCI0_CLK / UART0_RXD / PWM1_CH3 / TM3_EXT
57	AV_{DD}
58	V_{REF}
59	AVSS
60	PB.11 / ADC0_CH11 / UART0_nCTS / I2C1_SCL / SPI0_I2SMCLK
61	PB.10 / ADC0_CH10 / UART0_nRTS / I2C1_SDA
62	PB.9 / ADC0_CH9 / UART0_TXD / UART1_nCTS
63	PB.8 / ADC0_CH8 / UART0_RXD / UART1_nRTS
64	PB.7 / ADC0_CH7 / UART1_TXD / PWM1_BRAKE0 / PWM1_CH4 / INT5 / ACMP0_O

Table 4.1-14 M031SC2AE Multi-function Pin Table

M031SD2AE

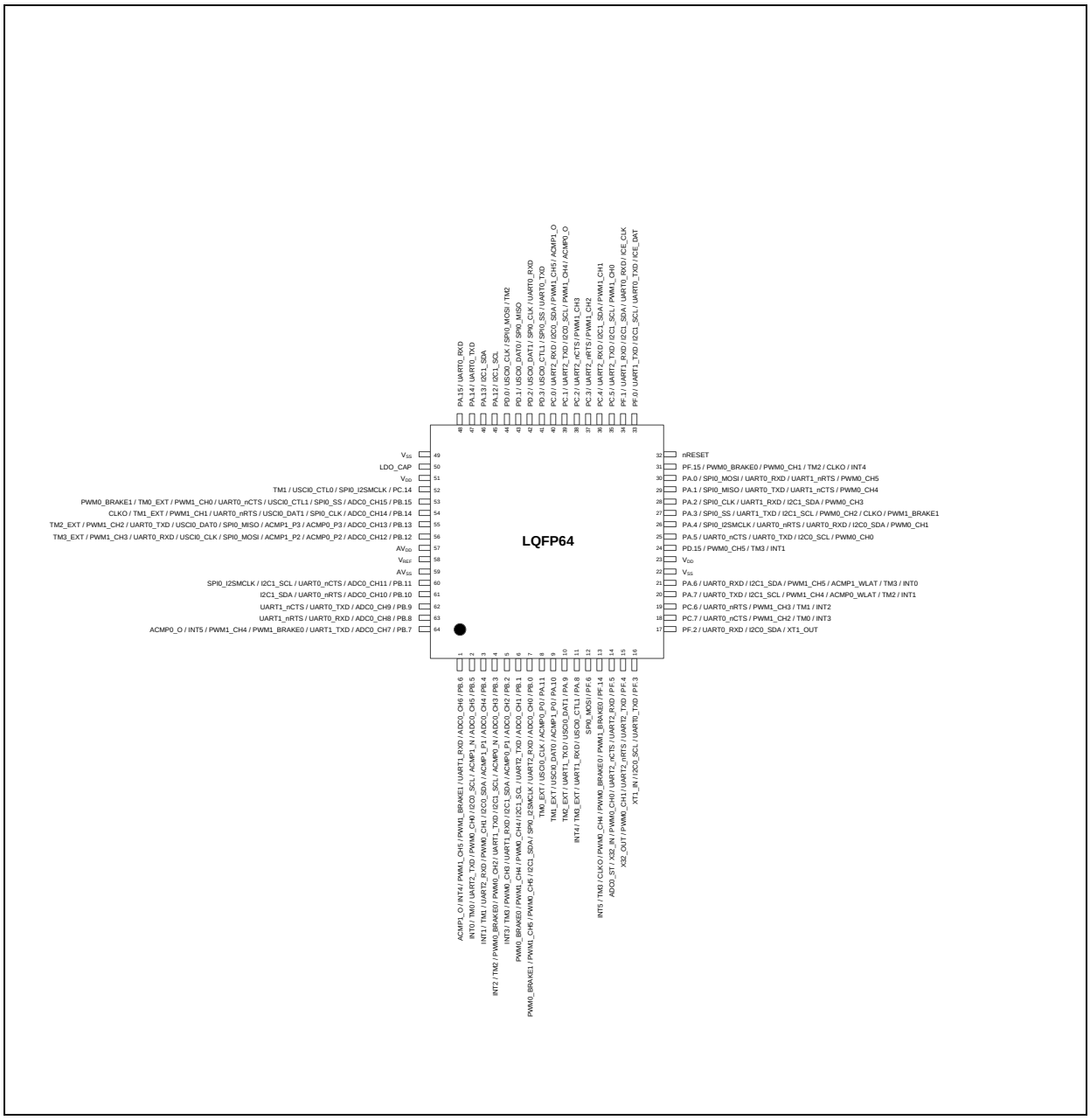


Figure 4.1-21 M031SD2AE Multi-function Pin Diagram

Pin	M031SD2AE Pin Function
1	PB.6 / ADC0_CH6 / UART1_RXD / PWM1_BRAKE1 / PWM1_CH5 / INT4 / ACMP1_O
2	PB.5 / ADC0_CH5 / ACMP1_N / I2C0_SCL / PWM0_CH0 / UART2_TXD / TM0 / INT0
3	PB.4 / ADC0_CH4 / ACMP1_P1 / I2C0_SDA / PWM0_CH1 / UART2_RXD / TM1 / INT1
4	PB.3 / ADC0_CH3 / ACMP0_N / I2C1_SCL / UART1_TXD / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2
5	PB.2 / ADC0_CH2 / ACMP0_P1 / I2C1_SDA / UART1_RXD / PWM0_CH3 / TM3 / INT3

Pin	M031SD2AE Pin Function
6	PB.1 / ADC0_CH1 / UART2_TXD / I2C1_SCL / PWM0_CH4 / PWM1_CH4 / PWM0_BRAKE0
7	PB.0 / ADC0_CH0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / PWM0_CH5 / PWM1_CH5 / PWM0_BRAKE1
8	PA.11 / ACMP0_P0 / USCI0_CLK / TM0_EXT
9	PA.10 / ACMP1_P0 / USCI0_DAT0 / TM1_EXT
10	PA.9 / USCI0_DAT1 / UART1_TXD / TM2_EXT
11	PA.8 / USCI0_CTL1 / UART1_RXD / TM3_EXT / INT4
12	PF.6 / SPI0_MOSI
13	PF.14 / PWM1_BRAKE0 / PWM0_BRAKE0 / PWM0_CH4 / CLKO / TM3 / INT5
14	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / X32_IN / ADC0_ST
15	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1 / X32_OUT
16	PF.3 / UART0_TXD / I2C0_SCL / XT1_IN
17	PF.2 / UART0_RXD / I2C0_SDA / XT1_OUT
18	PC.7 / UART0_nCTS / PWM1_CH2 / TM0 / INT3
19	PC.6 / UART0_nRTS / PWM1_CH3 / TM1 / INT2
20	PA.7 / UART0_TXD / I2C1_SCL / PWM1_CH4 / ACMP0_WLAT / TM2 / INT1
21	PA.6 / UART0_RXD / I2C1_SDA / PWM1_CH5 / ACMP1_WLAT / TM3 / INT0
22	VSS
23	V_{DD}
24	PD.15 / PWM0_CH5 / TM3 / INT1
25	PA.5 / UART0_nCTS / UART0_TXD / I2C0_SCL / PWM0_CH0
26	PA.4 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / PWM0_CH1
27	PA.3 / SPI0_SS / UART1_TXD / I2C1_SCL / PWM0_CH2 / CLKO / PWM1_BRAKE1
28	PA.2 / SPI0_CLK / UART1_RXD / I2C1_SDA / PWM0_CH3
29	PA.1 / SPI0_MISO / UART0_TXD / UART1_nCTS / PWM0_CH4
30	PA.0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / PWM0_CH5
31	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / TM2 / CLKO / INT4
32	nRESET
33	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / ICE_DAT
34	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / ICE_CLK
35	PC.5 / UART2_TXD / I2C1_SCL / PWM1_CH0
36	PC.4 / UART2_RXD / I2C1_SDA / PWM1_CH1
37	PC.3 / UART2_nRTS / PWM1_CH2
38	PC.2 / UART2_nCTS / PWM1_CH3
39	PC.1 / UART2_TXD / I2C0_SCL / PWM1_CH4 / ACMP0_O

Pin	M031SD2AE Pin Function
40	PC.0 / UART2_RXD / I2C0_SDA / PWM1_CH5 / ACMP1_O
41	PD.3 / USCI0_CTL1 / SPI0_SS / UART0_TXD
42	PD.2 / USCI0_DAT1 / SPI0_CLK / UART0_RXD
43	PD.1 / USCI0_DAT0 / SPI0_MISO
44	PD.0 / USCI0_CLK / SPI0_MOSI / TM2
45	PA.12 / I2C1_SCL
46	PA.13 / I2C1_SDA
47	PA.14 / UART0_TXD
48	PA.15 / UART0_RXD
49	VSS
50	LDO_CAP
51	V_{DD}
52	PC.14 / SPI0_I2SMCLK / USCI0_CTL0 / TM1
53	PB.15 / ADC0_CH15 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / PWM1_CH0 / TM0_EXT / PWM0_BRAKE1
54	PB.14 / ADC0_CH14 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / PWM1_CH1 / TM1_EXT / CLKO
55	PB.13 / ADC0_CH13 / ACMP0_P3 / ACMP1_P3 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / PWM1_CH2 / TM2_EXT
56	PB.12 / ADC0_CH12 / ACMP0_P2 / ACMP1_P2 / SPI0_MOSI / USCI0_CLK / UART0_RXD / PWM1_CH3 / TM3_EXT
57	AV_{DD}
58	V_{REF}
59	AVSS
60	PB.11 / ADC0_CH11 / UART0_nCTS / I2C1_SCL / SPI0_I2SMCLK
61	PB.10 / ADC0_CH10 / UART0_nRTS / I2C1_SDA
62	PB.9 / ADC0_CH9 / UART0_TXD / UART1_nCTS
63	PB.8 / ADC0_CH8 / UART0_RXD / UART1_nRTS
64	PB.7 / ADC0_CH7 / UART1_TXD / PWM1_BRAKE0 / PWM1_CH4 / INT5 / ACMP0_O

Table 4.1-15 M031SD2AE Multi-function Pin Table

M031SE3AE

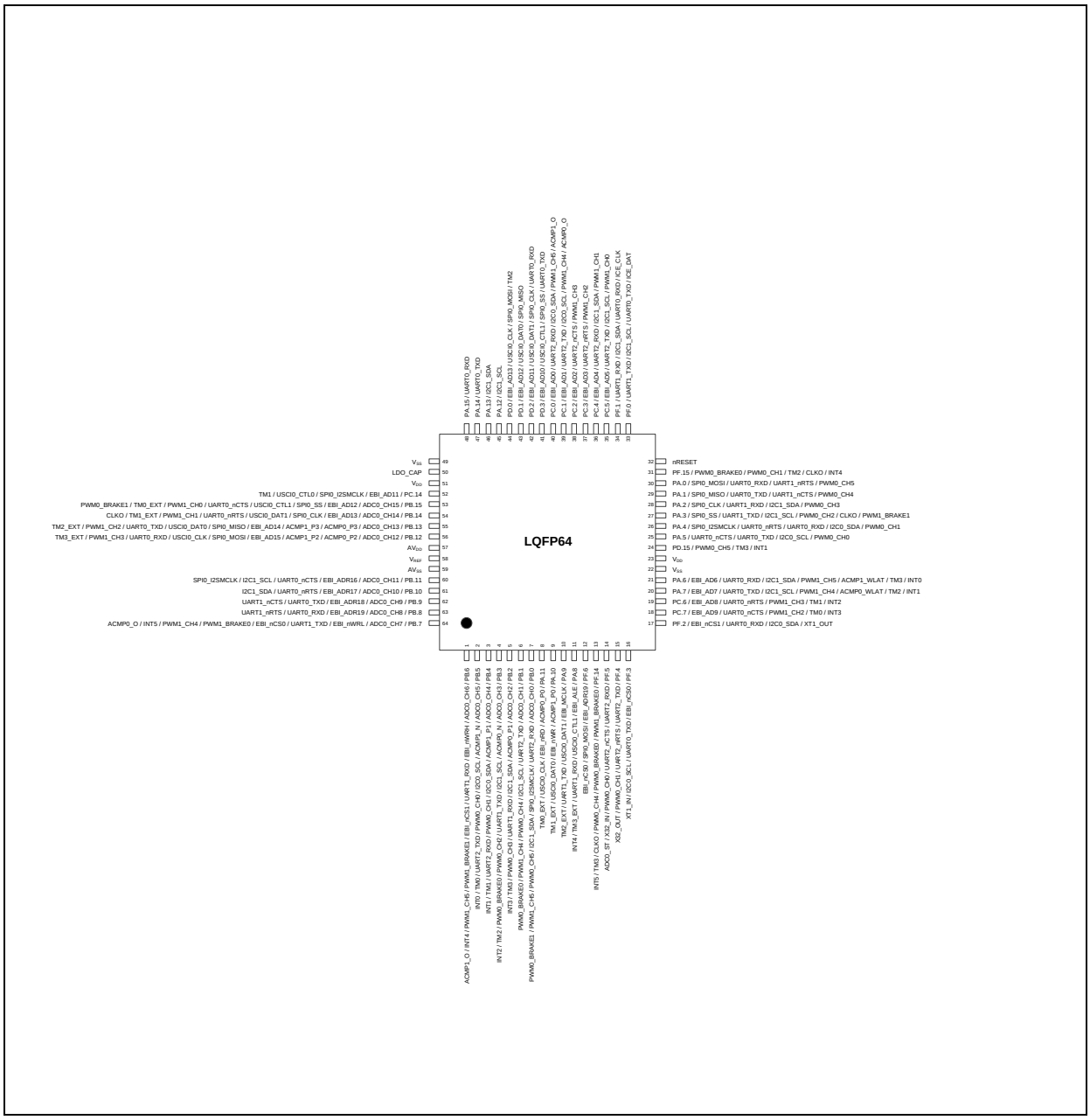


Figure 4.1-22 M031SE3AE Multi-function Pin Diagram

Pin	M031SE3AE Pin Function
1	PB.6 / ADC0_CH6 / EBI_nWRH / UART1_RXD / EBI_nCS1 / PWM1_BRAKE1 / PWM1_CH5 / INT4 / ACMP1_O
2	PB.5 / ADC0_CH5 / ACMP1_N / I2C0_SCL / PWM0_CH0 / UART2_TXD / TM0 / INT0
3	PB.4 / ADC0_CH4 / ACMP1_P1 / I2C0_SDA / PWM0_CH1 / UART2_RXD / TM1 / INT1
4	PB.3 / ADC0_CH3 / ACMP0_N / I2C1_SCL / UART1_TXD / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2
5	PB.2 / ADC0_CH2 / ACMP0_P1 / I2C1_SDA / UART1_RXD / PWM0_CH3 / TM3 / INT3
6	PB.1 / ADC0_CH1 / UART2_TXD / I2C1_SCL / PWM0_CH4 / PWM1_CH4 / PWM0_BRAKE0

Pin	M031SE3AE Pin Function
7	PB.0 / ADC0_CH0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / PWM0_CH5 / PWM1_CH5 / PWM0_BRAKE1
8	PA.11 / ACMP0_P0 / EBI_nRD / USCI0_CLK / TM0_EXT
9	PA.10 / ACMP1_P0 / EBI_nWR / USCI0_DAT0 / TM1_EXT
10	PA.9 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / TM2_EXT
11	PA.8 / EBI_ALE / USCI0_CTL1 / UART1_RXD / TM3_EXT / INT4
12	PF.6 / EBI_ADR19 / SPI0_MOSI / EBI_nCS0
13	PF.14 / PWM1_BRAKE0 / PWM0_BRAKE0 / PWM0_CH4 / CLKO / TM3 / INT5
14	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / X32_IN / ADC0_ST
15	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1 / X32_OUT
16	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN
17	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / XT1_OUT
18	PC.7 / EBI_AD9 / UART0_nCTS / PWM1_CH2 / TM0 / INT3
19	PC.6 / EBI_AD8 / UART0_nRTS / PWM1_CH3 / TM1 / INT2
20	PA.7 / EBI_AD7 / UART0_TXD / I2C1_SCL / PWM1_CH4 / ACMP0_WLAT / TM2 / INT1
21	PA.6 / EBI_AD6 / UART0_RXD / I2C1_SDA / PWM1_CH5 / ACMP1_WLAT / TM3 / INT0
22	VSS
23	V_{DD}
24	PD.15 / PWM0_CH5 / TM3 / INT1
25	PA.5 / UART0_nCTS / UART0_TXD / I2C0_SCL / PWM0_CH0
26	PA.4 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / PWM0_CH1
27	PA.3 / SPI0_SS / UART1_TXD / I2C1_SCL / PWM0_CH2 / CLKO / PWM1_BRAKE1
28	PA.2 / SPI0_CLK / UART1_RXD / I2C1_SDA / PWM0_CH3
29	PA.1 / SPI0_MISO / UART0_TXD / UART1_nCTS / PWM0_CH4
30	PA.0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / PWM0_CH5
31	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / TM2 / CLKO / INT4
32	nRESET
33	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / ICE_DAT
34	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / ICE_CLK
35	PC.5 / EBI_AD5 / UART2_TXD / I2C1_SCL / PWM1_CH0
36	PC.4 / EBI_AD4 / UART2_RXD / I2C1_SDA / PWM1_CH1
37	PC.3 / EBI_AD3 / UART2_nRTS / PWM1_CH2
38	PC.2 / EBI_AD2 / UART2_nCTS / PWM1_CH3
39	PC.1 / EBI_AD1 / UART2_TXD / I2C0_SCL / PWM1_CH4 / ACMP0_O
40	PC.0 / EBI_AD0 / UART2_RXD / I2C0_SDA / PWM1_CH5 / ACMP1_O

Pin	M031SE3AE Pin Function
41	PD.3 / EBI_AD10 / USCI0_CTL1 / SPI0_SS / UART0_TXD
42	PD.2 / EBI_AD11 / USCI0_DAT1 / SPI0_CLK / UART0_RXD
43	PD.1 / EBI_AD12 / USCI0_DAT0 / SPI0_MISO
44	PD.0 / EBI_AD13 / USCI0_CLK / SPI0_MOSI / TM2
45	PA.12 / I2C1_SCL
46	PA.13 / I2C1_SDA
47	PA.14 / UART0_TXD
48	PA.15 / UART0_RXD
49	VSS
50	LDO_CAP
51	V_{DD}
52	PC.14 / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / TM1
53	PB.15 / ADC0_CH15 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / PWM1_CH0 / TM0_EXT / PWM0_BRAKE1
54	PB.14 / ADC0_CH14 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / PWM1_CH1 / TM1_EXT / CLKO
55	PB.13 / ADC0_CH13 / ACMP0_P3 / ACMP1_P3 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / PWM1_CH2 / TM2_EXT
56	PB.12 / ADC0_CH12 / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / PWM1_CH3 / TM3_EXT
57	AV_{DD}
58	V_{REF}
59	AVSS
60	PB.11 / ADC0_CH11 / EBI_ADR16 / UART0_nCTS / I2C1_SCL / SPI0_I2SMCLK
61	PB.10 / ADC0_CH10 / EBI_ADR17 / UART0_nRTS / I2C1_SDA
62	PB.9 / ADC0_CH9 / EBI_ADR18 / UART0_TXD / UART1_nCTS
63	PB.8 / ADC0_CH8 / EBI_ADR19 / UART0_RXD / UART1_nRTS
64	PB.7 / ADC0_CH7 / EBI_nWRL / UART1_TXD / EBI_nCS0 / PWM1_BRAKE0 / PWM1_CH4 / INT5 / ACMP0_O

Table 4.1-16 M031SE3AE Multi-function Pin Table

M031SG6AE

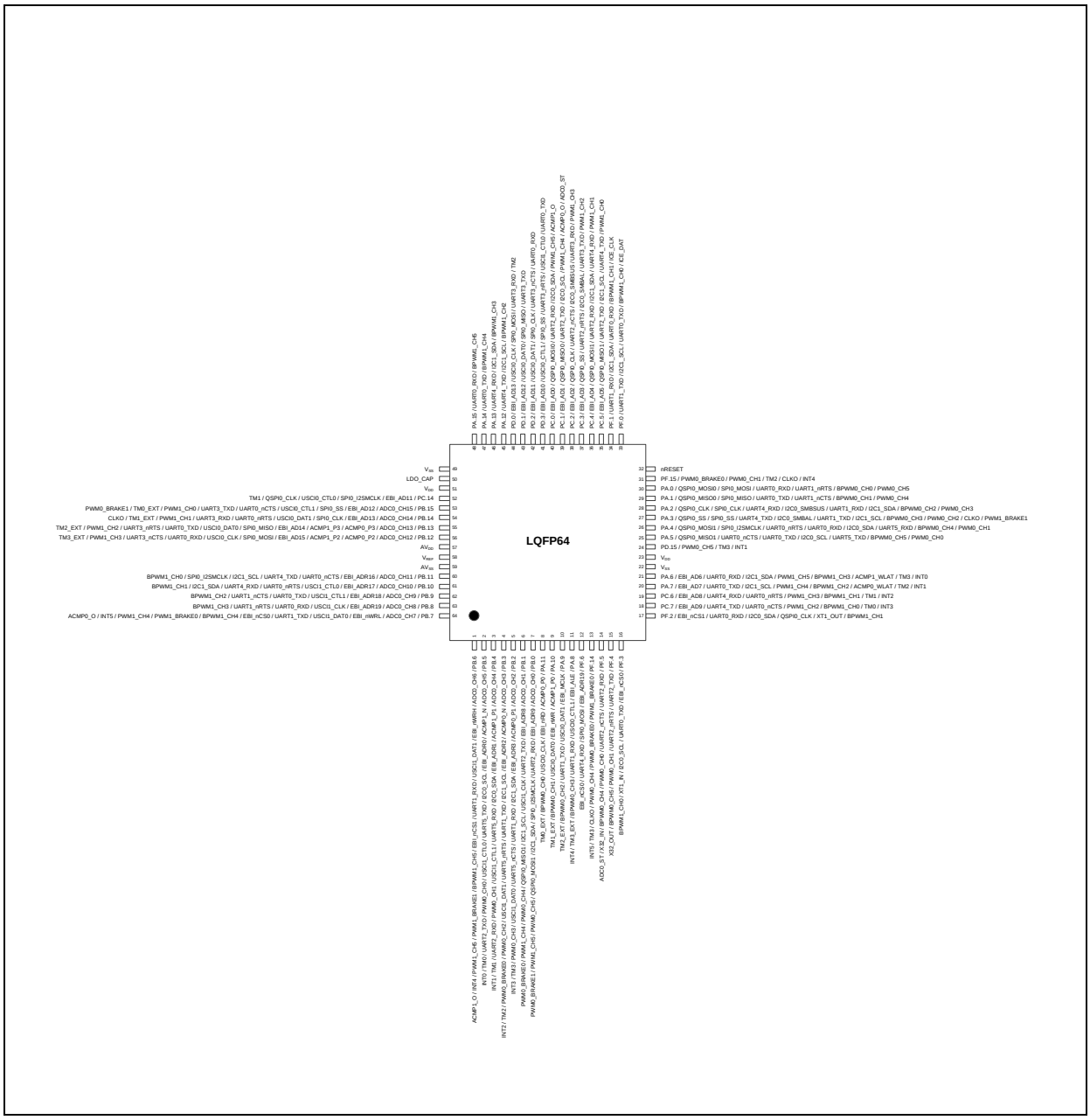


Figure 4.1-23 M031SG6AE Multi-function Pin Diagram

Pin	M031SG6AE Pin Function
1	PB.6 / ADC0_CH6 / EBI_nWRH / USCI1_DAT1 / UART1_RXD / EBI_nCS1 / BPWM1_CH5 / PWM1_BRAKE1 / PWM1_CH5 / INT4 / ACMP1_O
2	PB.5 / ADC0_CH5 / ACMP1_N / EBI_ADR0 / I2C0_SCL / UART5_TXD / USCI1_CTL0 / PWM0_CH0 / UART2_TXD / TM0 / INT0
3	PB.4 / ADC0_CH4 / ACMP1_P1 / EBI_ADR1 / I2C0_SDA / UART5_RXD / USCI1_CTL1 / PWM0_CH1 / UART2_RXD / TM1 / INT1

Pin	M031SG6AE Pin Function
4	PB.3 / ADC0_CH3 / ACMP0_N / EBI_ADR2 / I2C1_SCL / UART1_TXD / UART5_nRTS / USCI1_DAT1 / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2
5	PB.2 / ADC0_CH2 / ACMP0_P1 / EBI_ADR3 / I2C1_SDA / UART1_RXD / UART5_nCTS / USCI1_DAT0 / PWM0_CH3 / TM3 / INT3
6	PB.1 / ADC0_CH1 / EBI_ADR8 / UART2_TXD / USCI1_CLK / I2C1_SCL / QSPI0_MISO1 / PWM0_CH4 / PWM1_CH4 / PWM0_BRAKE0
7	PB.0 / ADC0_CH0 / EBI_ADR9 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / PWM0_CH5 / PWM1_CH5 / PWM0_BRAKE1
8	PA.11 / ACMP0_P0 / EBI_nRD / USCI0_CLK / BPWM0_CH0 / TM0_EXT
9	PA.10 / ACMP1_P0 / EBI_nWR / USCI0_DAT0 / BPWM0_CH1 / TM1_EXT
10	PA.9 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / BPWM0_CH2 / TM2_EXT
11	PA.8 / EBI_ALE / USCI0_CTL1 / UART1_RXD / BPWM0_CH3 / TM3_EXT / INT4
12	PF.6 / EBI_ADR19 / SPI0_MOSI / UART4_RXD / EBI_nCS0
13	PF.14 / PWM1_BRAKE0 / PWM0_BRAKE0 / PWM0_CH4 / CLKO / TM3 / INT5
14	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / BPWM0_CH4 / X32_IN / ADC0_ST
15	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1 / BPWM0_CH5 / X32_OUT
16	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / BPWM1_CH0
17	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / BPWM1_CH1
18	PC.7 / EBI_AD9 / UART4_TXD / UART0_nCTS / PWM1_CH2 / BPWM1_CH0 / TM0 / INT3
19	PC.6 / EBI_AD8 / UART4_RXD / UART0_nRTS / PWM1_CH3 / BPWM1_CH1 / TM1 / INT2
20	PA.7 / EBI_AD7 / UART0_TXD / I2C1_SCL / PWM1_CH4 / BPWM1_CH2 / ACMP0_WLAT / TM2 / INT1
21	PA.6 / EBI_AD6 / UART0_RXD / I2C1_SDA / PWM1_CH5 / BPWM1_CH3 / ACMP1_WLAT / TM3 / INT0
22	VSS
23	V_{DD}
24	PD.15 / PWM0_CH5 / TM3 / INT1
25	PA.5 / QSPI0_MISO1 / UART0_nCTS / UART0_TXD / I2C0_SCL / UART5_TXD / BPWM0_CH5 / PWM0_CH0
26	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / UART5_RXD / BPWM0_CH4 / PWM0_CH1
27	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / I2C0_SMBAL / UART1_TXD / I2C1_SCL / BPWM0_CH3 / PWM0_CH2 / CLKO / PWM1_BRAKE1
28	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / BPWM0_CH2 / PWM0_CH3
29	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / BPWM0_CH1 / PWM0_CH4
30	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / BPWM0_CH0 / PWM0_CH5
31	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / TM2 / CLKO / INT4
32	nRESET
33	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / BPWM1_CH0 / ICE_DAT
34	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / BPWM1_CH1 / ICE_CLK

Pin	M031SG6AE Pin Function
35	PC.5 / EBI_AD5 / QSPI0_MISO1 / UART2_TXD / I2C1_SCL / UART4_TXD / PWM1_CH0
36	PC.4 / EBI_AD4 / QSPI0_MOSI1 / UART2_RXD / I2C1_SDA / UART4_RXD / PWM1_CH1
37	PC.3 / EBI_AD3 / QSPI0_SS / UART2_nRTS / I2C0_SMBAL / UART3_TXD / PWM1_CH2
38	PC.2 / EBI_AD2 / QSPI0_CLK / UART2_nCTS / I2C0_SMBSUS / UART3_RXD / PWM1_CH3
39	PC.1 / EBI_AD1 / QSPI0_MISO0 / UART2_TXD / I2C0_SCL / PWM1_CH4 / ACMP0_O / ADC0_ST
40	PC.0 / EBI_AD0 / QSPI0_MOSI0 / UART2_RXD / I2C0_SDA / PWM1_CH5 / ACMP1_O
41	PD.3 / EBI_AD10 / USCI0_CTL1 / SPI0_SS / UART3_nRTS / USCI1_CTL0 / UART0_TXD
42	PD.2 / EBI_AD11 / USCI0_DAT1 / SPI0_CLK / UART3_nCTS / UART0_RXD
43	PD.1 / EBI_AD12 / USCI0_DAT0 / SPI0_MISO / UART3_TXD
44	PD.0 / EBI_AD13 / USCI0_CLK / SPI0_MOSI / UART3_RXD / TM2
45	PA.12 / UART4_TXD / I2C1_SCL / BPWM1_CH2
46	PA.13 / UART4_RXD / I2C1_SDA / BPWM1_CH3
47	PA.14 / UART0_TXD / BPWM1_CH4
48	PA.15 / UART0_RXD / BPWM1_CH5
49	VSS
50	LDO_CAP
51	V_{DD}
52	PC.14 / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / TM1
53	PB.15 / ADC0_CH15 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / PWM1_CH0 / TM0_EXT / PWM0_BRAKE1
54	PB.14 / ADC0_CH14 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / PWM1_CH1 / TM1_EXT / CLKO
55	PB.13 / ADC0_CH13 / ACMP0_P3 / ACMP1_P3 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / PWM1_CH2 / TM2_EXT
56	PB.12 / ADC0_CH12 / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / PWM1_CH3 / TM3_EXT
57	AV_{DD}
58	V_{REF}
59	AVSS
60	PB.11 / ADC0_CH11 / EBI_ADR16 / UART0_nCTS / UART4_TXD / I2C1_SCL / SPI0_I2SMCLK / BPWM1_CH0
61	PB.10 / ADC0_CH10 / EBI_ADR17 / USCI1_CTL0 / UART0_nRTS / UART4_RXD / I2C1_SDA / BPWM1_CH1
62	PB.9 / ADC0_CH9 / EBI_ADR18 / USCI1_CTL1 / UART0_TXD / UART1_nCTS / BPWM1_CH2
63	PB.8 / ADC0_CH8 / EBI_ADR19 / USCI1_CLK / UART0_RXD / UART1_nRTS / BPWM1_CH3
64	PB.7 / ADC0_CH7 / EBI_nWRL / USCI1_DAT0 / UART1_TXD / EBI_nCS0 / BPWM1_CH4 / PWM1_BRAKE0 / PWM1_CH4 / INT5 / ACMP0_O

Table 4.1-17 M031SG6AE Multi-function Pin Table

M031SG8AE

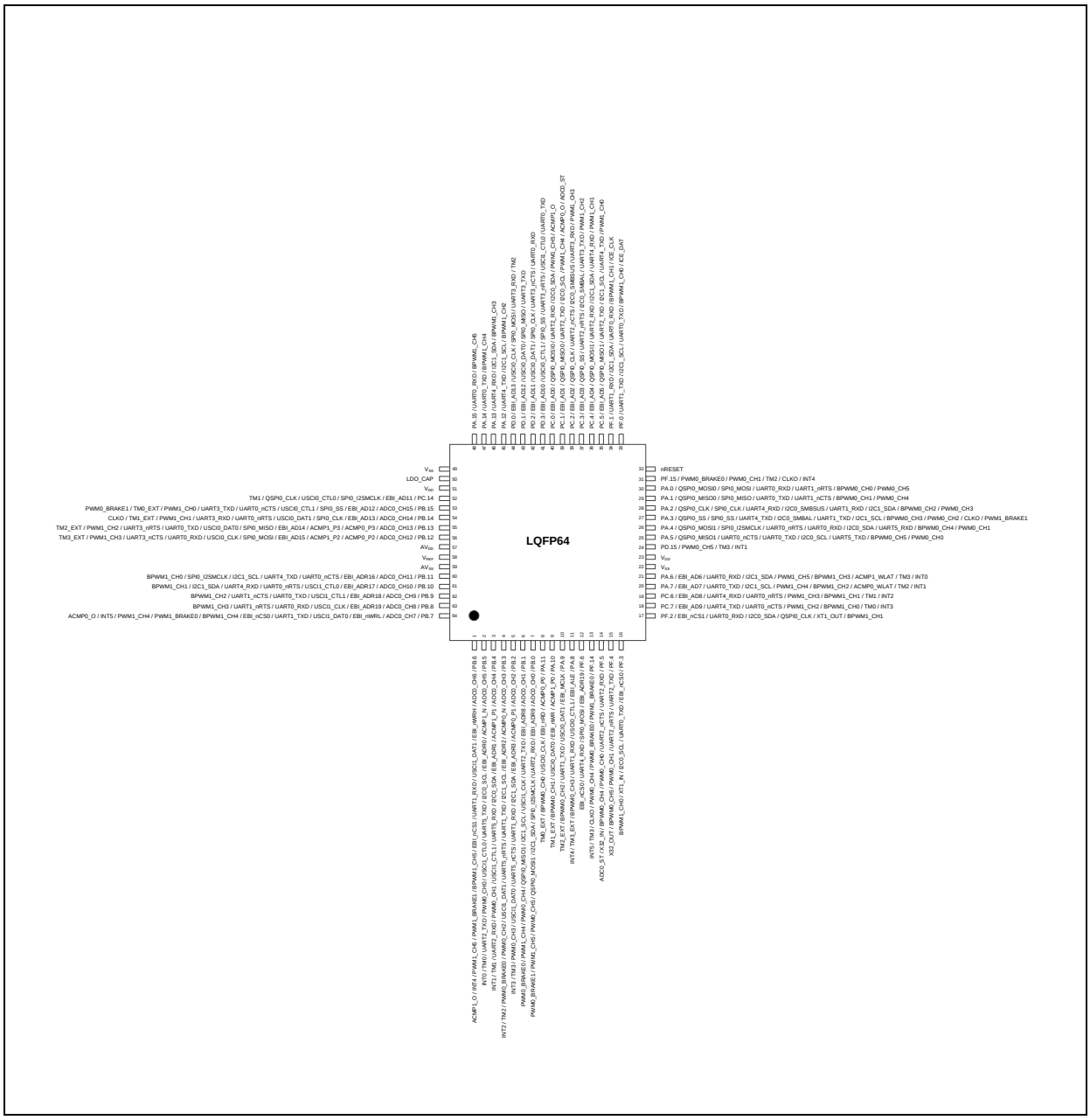


Figure 4.1-24 M031SG8AE Multi-function Pin Diagram

Pin	M031SG8AE Pin Function
1	PB.6 / ADC0_CH6 / EBI_nWRH / USCI1_DAT1 / UART1_RXD / EBI_nCS1 / BPWM1_CH5 / PWM1_BRAKE1 / PWM1_CH5 / INT4 / ACMP1_O
2	PB.5 / ADC0_CH5 / ACMP1_N / EBI_ADR0 / I2C0_SCL / UART5_TXD / USCI1_CTL0 / PWM0_CH0 / UART2_TXD / TM0 / INT0
3	PB.4 / ADC0_CH4 / ACMP1_P1 / EBI_ADR1 / I2C0_SDA / UART5_RXD / USCI1_CTL1 / PWM0_CH1 / UART2_RXD / TM1 / INT1

Pin	M031SG8AE Pin Function
4	PB.3 / ADC0_CH3 / ACMP0_N / EBI_ADR2 / I2C1_SCL / UART1_TXD / UART5_nRTS / USCI1_DAT1 / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2
5	PB.2 / ADC0_CH2 / ACMP0_P1 / EBI_ADR3 / I2C1_SDA / UART1_RXD / UART5_nCTS / USCI1_DAT0 / PWM0_CH3 / TM3 / INT3
6	PB.1 / ADC0_CH1 / EBI_ADR8 / UART2_TXD / USCI1_CLK / I2C1_SCL / QSPI0_MISO1 / PWM0_CH4 / PWM1_CH4 / PWM0_BRAKE0
7	PB.0 / ADC0_CH0 / EBI_ADR9 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / PWM0_CH5 / PWM1_CH5 / PWM0_BRAKE1
8	PA.11 / ACMP0_P0 / EBI_nRD / USCI0_CLK / BPWM0_CH0 / TM0_EXT
9	PA.10 / ACMP1_P0 / EBI_nWR / USCI0_DAT0 / BPWM0_CH1 / TM1_EXT
10	PA.9 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / BPWM0_CH2 / TM2_EXT
11	PA.8 / EBI_ALE / USCI0_CTL1 / UART1_RXD / BPWM0_CH3 / TM3_EXT / INT4
12	PF.6 / EBI_ADR19 / SPI0_MOSI / UART4_RXD / EBI_nCS0
13	PF.14 / PWM1_BRAKE0 / PWM0_BRAKE0 / PWM0_CH4 / CLKO / TM3 / INT5
14	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / BPWM0_CH4 / X32_IN / ADC0_ST
15	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1 / BPWM0_CH5 / X32_OUT
16	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / BPWM1_CH0
17	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / BPWM1_CH1
18	PC.7 / EBI_AD9 / UART4_TXD / UART0_nCTS / PWM1_CH2 / BPWM1_CH0 / TM0 / INT3
19	PC.6 / EBI_AD8 / UART4_RXD / UART0_nRTS / PWM1_CH3 / BPWM1_CH1 / TM1 / INT2
20	PA.7 / EBI_AD7 / UART0_TXD / I2C1_SCL / PWM1_CH4 / BPWM1_CH2 / ACMP0_WLAT / TM2 / INT1
21	PA.6 / EBI_AD6 / UART0_RXD / I2C1_SDA / PWM1_CH5 / BPWM1_CH3 / ACMP1_WLAT / TM3 / INT0
22	VSS
23	V_{DD}
24	PD.15 / PWM0_CH5 / TM3 / INT1
25	PA.5 / QSPI0_MISO1 / UART0_nCTS / UART0_TXD / I2C0_SCL / UART5_TXD / BPWM0_CH5 / PWM0_CH0
26	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / UART5_RXD / BPWM0_CH4 / PWM0_CH1
27	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / I2C0_SMBAL / UART1_TXD / I2C1_SCL / BPWM0_CH3 / PWM0_CH2 / CLKO / PWM1_BRAKE1
28	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / BPWM0_CH2 / PWM0_CH3
29	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / BPWM0_CH1 / PWM0_CH4
30	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / BPWM0_CH0 / PWM0_CH5
31	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / TM2 / CLKO / INT4
32	nRESET
33	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / BPWM1_CH0 / ICE_DAT
34	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / BPWM1_CH1 / ICE_CLK

Pin	M031SG8AE Pin Function
35	PC.5 / EBI_AD5 / QSPI0_MISO1 / UART2_TXD / I2C1_SCL / UART4_TXD / PWM1_CH0
36	PC.4 / EBI_AD4 / QSPI0_MOSI1 / UART2_RXD / I2C1_SDA / UART4_RXD / PWM1_CH1
37	PC.3 / EBI_AD3 / QSPI0_SS / UART2_nRTS / I2C0_SMBAL / UART3_TXD / PWM1_CH2
38	PC.2 / EBI_AD2 / QSPI0_CLK / UART2_nCTS / I2C0_SMBSUS / UART3_RXD / PWM1_CH3
39	PC.1 / EBI_AD1 / QSPI0_MISO0 / UART2_TXD / I2C0_SCL / PWM1_CH4 / ACMP0_O / ADC0_ST
40	PC.0 / EBI_AD0 / QSPI0_MOSI0 / UART2_RXD / I2C0_SDA / PWM1_CH5 / ACMP1_O
41	PD.3 / EBI_AD10 / USCI0_CTL1 / SPI0_SS / UART3_nRTS / USCI1_CTL0 / UART0_TXD
42	PD.2 / EBI_AD11 / USCI0_DAT1 / SPI0_CLK / UART3_nCTS / UART0_RXD
43	PD.1 / EBI_AD12 / USCI0_DAT0 / SPI0_MISO / UART3_TXD
44	PD.0 / EBI_AD13 / USCI0_CLK / SPI0_MOSI / UART3_RXD / TM2
45	PA.12 / UART4_TXD / I2C1_SCL / BPWM1_CH2
46	PA.13 / UART4_RXD / I2C1_SDA / BPWM1_CH3
47	PA.14 / UART0_TXD / BPWM1_CH4
48	PA.15 / UART0_RXD / BPWM1_CH5
49	VSS
50	LDO_CAP
51	V_{DD}
52	PC.14 / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / TM1
53	PB.15 / ADC0_CH15 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / PWM1_CH0 / TM0_EXT / PWM0_BRAKE1
54	PB.14 / ADC0_CH14 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / PWM1_CH1 / TM1_EXT / CLKO
55	PB.13 / ADC0_CH13 / ACMP0_P3 / ACMP1_P3 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / PWM1_CH2 / TM2_EXT
56	PB.12 / ADC0_CH12 / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / PWM1_CH3 / TM3_EXT
57	AV_{DD}
58	V_{REF}
59	AVSS
60	PB.11 / ADC0_CH11 / EBI_ADR16 / UART0_nCTS / UART4_TXD / I2C1_SCL / SPI0_I2SMCLK / BPWM1_CH0
61	PB.10 / ADC0_CH10 / EBI_ADR17 / USCI1_CTL0 / UART0_nRTS / UART4_RXD / I2C1_SDA / BPWM1_CH1
62	PB.9 / ADC0_CH9 / EBI_ADR18 / USCI1_CTL1 / UART0_TXD / UART1_nCTS / BPWM1_CH2
63	PB.8 / ADC0_CH8 / EBI_ADR19 / USCI1_CLK / UART0_RXD / UART1_nRTS / BPWM1_CH3
64	PB.7 / ADC0_CH7 / EBI_nWRL / USCI1_DAT0 / UART1_TXD / EBI_nCS0 / BPWM1_CH4 / PWM1_BRAKE0 / PWM1_CH4 / INT5 / ACMP0_O

Table 4.1-18 M031SG8AE Multi-function Pin Table

M031SIAAE

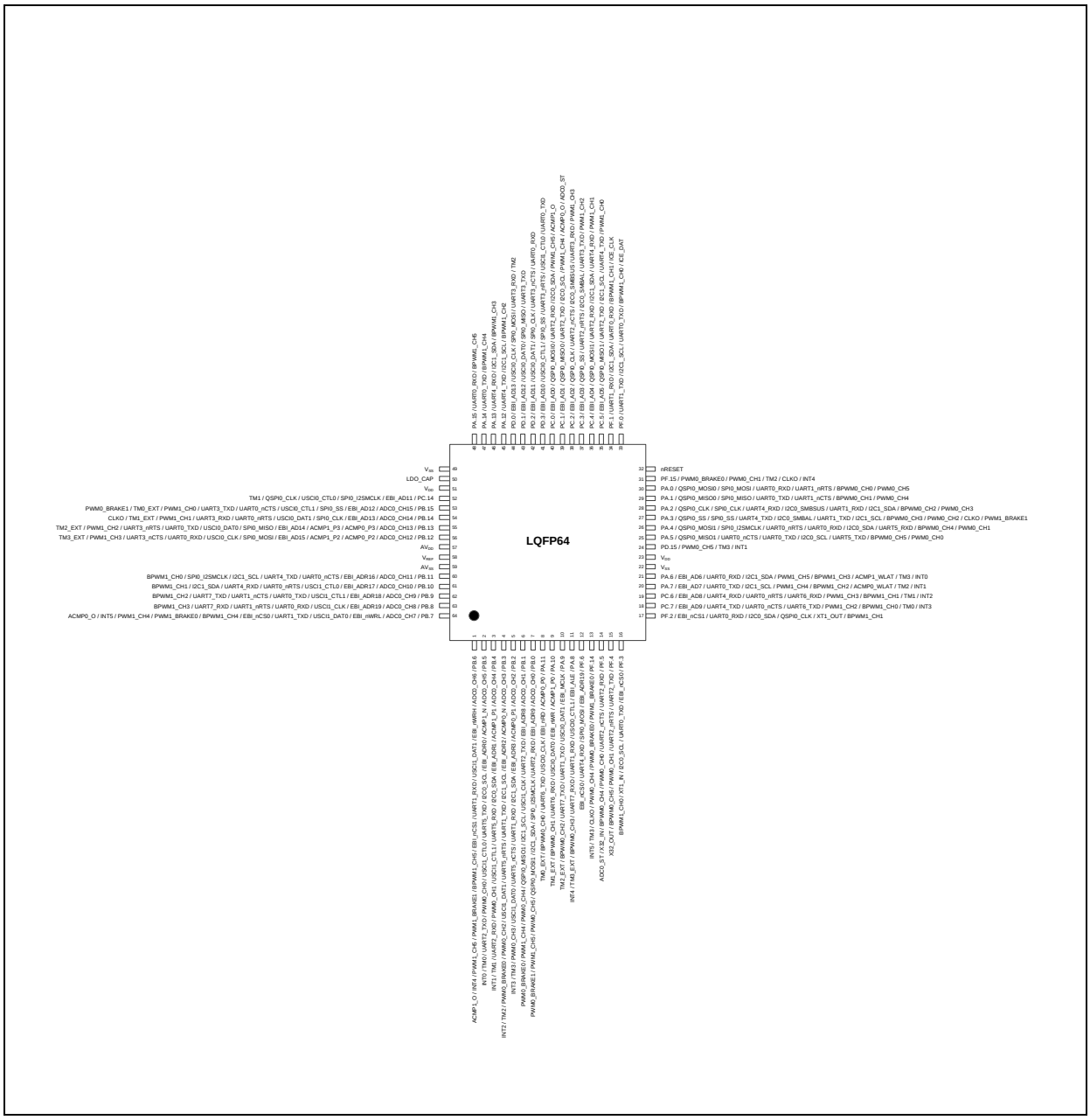


Figure 4.1-25 M031SIAAE Multi-function Pin Diagram

Pin	M031SIAAE Pin Function
1	PB.6 / ADC0_CH6 / EBI_nWRH / USCI1_DAT1 / UART1_RXD / EBI_nCS1 / BPWM1_CH5 / PWM1_BRAKE1 / PWM1_CH5 / INT4 / ACMP1_O
2	PB.5 / ADC0_CH5 / ACMP1_N / EBI_ADR0 / I2C0_SCL / UART5_TXD / USCI1_CTL0 / PWM0_CH0 / UART2_TXD / TM0 / INT0
3	PB.4 / ADC0_CH4 / ACMP1_P1 / EBI_ADR1 / I2C0_SDA / UART5_RXD / USCI1_CTL1 / PWM0_CH1 / UART2_RXD / TM1 / INT1

Pin	M031SIAAE Pin Function
4	PB.3 / ADC0_CH3 / ACMP0_N / EBI_ADR2 / I2C1_SCL / UART1_TXD / UART5_nRTS / USCI1_DAT1 / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2
5	PB.2 / ADC0_CH2 / ACMP0_P1 / EBI_ADR3 / I2C1_SDA / UART1_RXD / UART5_nCTS / USCI1_DAT0 / PWM0_CH3 / TM3 / INT3
6	PB.1 / ADC0_CH1 / EBI_ADR8 / UART2_TXD / USCI1_CLK / I2C1_SCL / QSPI0_MISO1 / PWM0_CH4 / PWM1_CH4 / PWM0_BRAKE0
7	PB.0 / ADC0_CH0 / EBI_ADR9 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / PWM0_CH5 / PWM1_CH5 / PWM0_BRAKE1
8	PA.11 / ACMP0_P0 / EBI_nRD / USCI0_CLK / UART6_TXD / BPWM0_CH0 / TM0_EXT
9	PA.10 / ACMP1_P0 / EBI_nWR / USCI0_DAT0 / UART6_RXD / BPWM0_CH1 / TM1_EXT
10	PA.9 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / UART7_TXD / BPWM0_CH2 / TM2_EXT
11	PA.8 / EBI_ALE / USCI0_CTL1 / UART1_RXD / UART7_RXD / BPWM0_CH3 / TM3_EXT / INT4
12	PF.6 / EBI_ADR19 / SPI0_MOSI / UART4_RXD / EBI_nCS0
13	PF.14 / PWM1_BRAKE0 / PWM0_BRAKE0 / PWM0_CH4 / CLKO / TM3 / INT5
14	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / BPWM0_CH4 / X32_IN / ADC0_ST
15	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1 / BPWM0_CH5 / X32_OUT
16	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / BPWM1_CH0
17	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / BPWM1_CH1
18	PC.7 / EBI_AD9 / UART4_TXD / UART0_nCTS / UART6_TXD / PWM1_CH2 / BPWM1_CH0 / TM0 / INT3
19	PC.6 / EBI_AD8 / UART4_RXD / UART0_nRTS / UART6_RXD / PWM1_CH3 / BPWM1_CH1 / TM1 / INT2
20	PA.7 / EBI_AD7 / UART0_TXD / I2C1_SCL / PWM1_CH4 / BPWM1_CH2 / ACMP0_WLAT / TM2 / INT1
21	PA.6 / EBI_AD6 / UART0_RXD / I2C1_SDA / PWM1_CH5 / BPWM1_CH3 / ACMP1_WLAT / TM3 / INT0
22	VSS
23	V_{DD}
24	PD.15 / PWM0_CH5 / TM3 / INT1
25	PA.5 / QSPI0_MISO1 / UART0_nCTS / UART0_TXD / I2C0_SCL / UART5_TXD / BPWM0_CH5 / PWM0_CH0
26	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / UART5_RXD / BPWM0_CH4 / PWM0_CH1
27	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / I2C0_SMBAL / UART1_TXD / I2C1_SCL / BPWM0_CH3 / PWM0_CH2 / CLKO / PWM1_BRAKE1
28	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / BPWM0_CH2 / PWM0_CH3
29	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / BPWM0_CH1 / PWM0_CH4
30	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / BPWM0_CH0 / PWM0_CH5
31	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / TM2 / CLKO / INT4
32	nRESET
33	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / BPWM1_CH0 / ICE_DAT
34	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / BPWM1_CH1 / ICE_CLK

Pin	M031SIAAE Pin Function
35	PC.5 / EBI_AD5 / QSPI0_MISO1 / UART2_TXD / I2C1_SCL / UART4_TXD / PWM1_CH0
36	PC.4 / EBI_AD4 / QSPI0_MOSI1 / UART2_RXD / I2C1_SDA / UART4_RXD / PWM1_CH1
37	PC.3 / EBI_AD3 / QSPI0_SS / UART2_nRTS / I2C0_SMBAL / UART3_TXD / PWM1_CH2
38	PC.2 / EBI_AD2 / QSPI0_CLK / UART2_nCTS / I2C0_SMBSUS / UART3_RXD / PWM1_CH3
39	PC.1 / EBI_AD1 / QSPI0_MISO0 / UART2_TXD / I2C0_SCL / PWM1_CH4 / ACMP0_O / ADC0_ST
40	PC.0 / EBI_AD0 / QSPI0_MOSI0 / UART2_RXD / I2C0_SDA / PWM1_CH5 / ACMP1_O
41	PD.3 / EBI_AD10 / USCI0_CTL1 / SPI0_SS / UART3_nRTS / USCI1_CTL0 / UART0_TXD
42	PD.2 / EBI_AD11 / USCI0_DAT1 / SPI0_CLK / UART3_nCTS / UART0_RXD
43	PD.1 / EBI_AD12 / USCI0_DAT0 / SPI0_MISO / UART3_TXD
44	PD.0 / EBI_AD13 / USCI0_CLK / SPI0_MOSI / UART3_RXD / TM2
45	PA.12 / UART4_TXD / I2C1_SCL / BPWM1_CH2
46	PA.13 / UART4_RXD / I2C1_SDA / BPWM1_CH3
47	PA.14 / UART0_TXD / BPWM1_CH4
48	PA.15 / UART0_RXD / BPWM1_CH5
49	VSS
50	LDO_CAP
51	V_{DD}
52	PC.14 / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / TM1
53	PB.15 / ADC0_CH15 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / PWM1_CH0 / TM0_EXT / PWM0_BRAKE1
54	PB.14 / ADC0_CH14 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / PWM1_CH1 / TM1_EXT / CLKO
55	PB.13 / ADC0_CH13 / ACMP0_P3 / ACMP1_P3 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / PWM1_CH2 / TM2_EXT
56	PB.12 / ADC0_CH12 / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / PWM1_CH3 / TM3_EXT
57	AV_{DD}
58	V_{REF}
59	AVSS
60	PB.11 / ADC0_CH11 / EBI_ADR16 / UART0_nCTS / UART4_TXD / I2C1_SCL / SPI0_I2SMCLK / BPWM1_CH0
61	PB.10 / ADC0_CH10 / EBI_ADR17 / USCI1_CTL0 / UART0_nRTS / UART4_RXD / I2C1_SDA / BPWM1_CH1
62	PB.9 / ADC0_CH9 / EBI_ADR18 / USCI1_CTL1 / UART0_TXD / UART1_nCTS / UART7_TXD / BPWM1_CH2
63	PB.8 / ADC0_CH8 / EBI_ADR19 / USCI1_CLK / UART0_RXD / UART1_nRTS / UART7_RXD / BPWM1_CH3
64	PB.7 / ADC0_CH7 / EBI_nWRL / USCI1_DAT0 / UART1_TXD / EBI_nCS0 / BPWM1_CH4 / PWM1_BRAKE0 / PWM1_CH4 / INT5 / ACMP0_O

Table 4.1-19 M031SIAAE Multi-function Pin Table

4.1.2.6 M031 Series LQFP 128-Pin Multi-function Pin Diagram

Corresponding Part Number: M031KG6AE, M031KG8AE, M031KIAAE

M031KG6AE

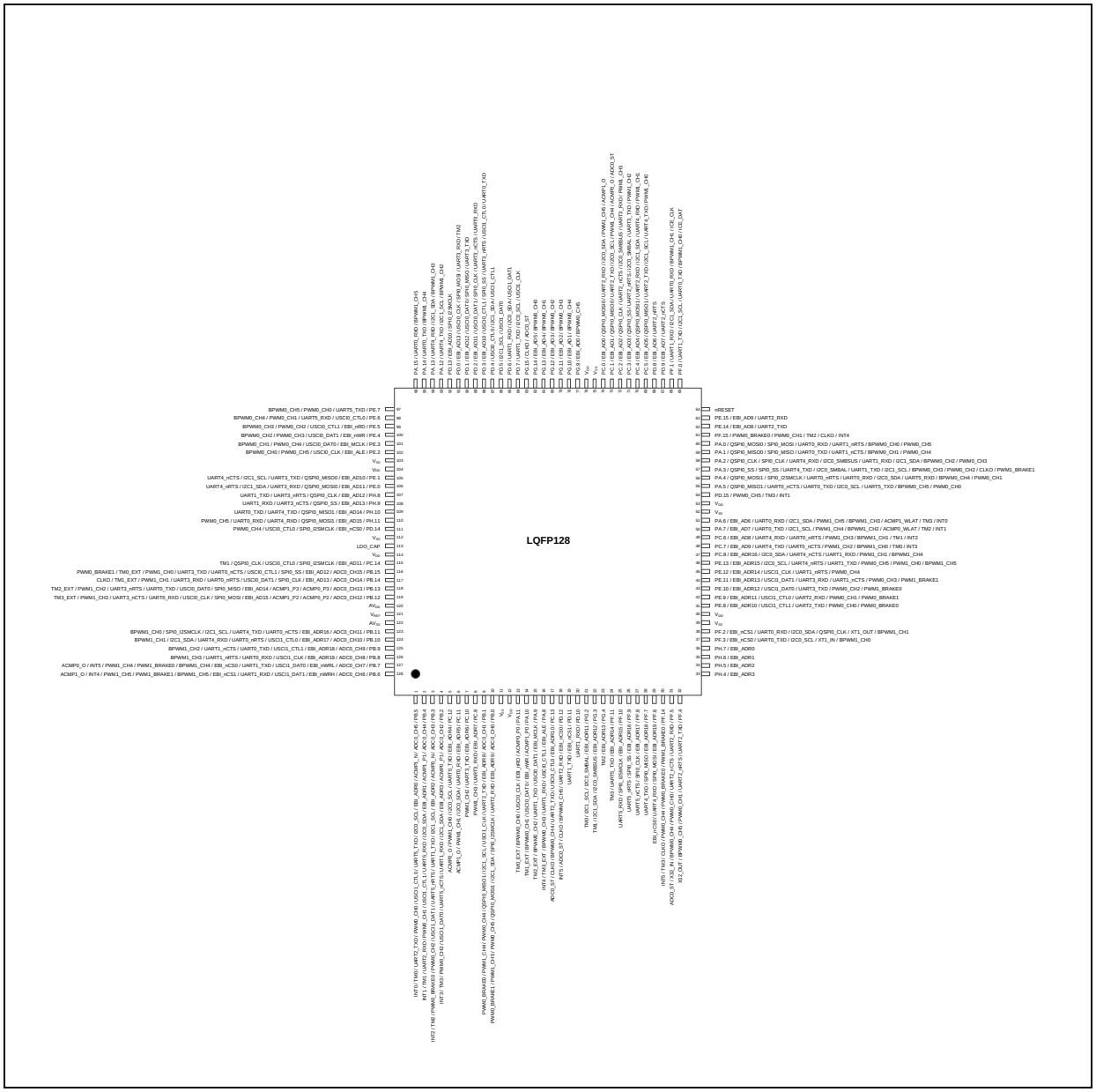


Figure 4.1-26 M031KG6AE Multi-function Pin Diagram

Pin	M031KG6AE Pin Function
1	PB.5 / ADC0_CH5 / ACMP1_N / EBI_ADR0 / I2C0_SCL / UART5_TXD / USCI1_CTL0 / PWM0_CH0 / UART2_TXD / TM0 / INT0
2	PB.4 / ADC0_CH4 / ACMP1_P1 / EBI_ADR1 / I2C0_SDA / UART5_RXD / USCI1_CTL1 / PWM0_CH1 / UART2_RXD / TM1 / INT1

Pin	M031KG6AE Pin Function
3	PB.3 / ADC0_CH3 / ACMP0_N / EBI_ADR2 / I2C1_SCL / UART1_TXD / UART5_nRTS / USCI1_DAT1 / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2
4	PB.2 / ADC0_CH2 / ACMP0_P1 / EBI_ADR3 / I2C1_SDA / UART1_RXD / UART5_nCTS / USCI1_DAT0 / PWM0_CH3 / TM3 / INT3
5	PC.12 / EBI_ADR4 / UART0_TXD / I2C0_SCL / PWM1_CH0 / ACMP0_O
6	PC.11 / EBI_ADR5 / UART0_RXD / I2C0_SDA / PWM1_CH1 / ACMP1_O
7	PC.10 / EBI_ADR6 / UART3_TXD / PWM1_CH2
8	PC.9 / EBI_ADR7 / UART3_RXD / PWM1_CH3
9	PB.1 / ADC0_CH1 / EBI_ADR8 / UART2_TXD / USCI1_CLK / I2C1_SCL / QSPI0_MISO1 / PWM0_CH4 / PWM1_CH4 / PWM0_BRAKE0
10	PB.0 / ADC0_CH0 / EBI_ADR9 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / PWM0_CH5 / PWM1_CH5 / PWM0_BRAKE1
11	VSS
12	V_{DD}
13	PA.11 / ACMP0_P0 / EBI_nRD / USCI0_CLK / BPWM0_CH0 / TM0_EXT
14	PA.10 / ACMP1_P0 / EBI_nWR / USCI0_DAT0 / BPWM0_CH1 / TM1_EXT
15	PA.9 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / BPWM0_CH2 / TM2_EXT
16	PA.8 / EBI_ALE / USCI0_CTL1 / UART1_RXD / BPWM0_CH3 / TM3_EXT / INT4
17	PC.13 / EBI_ADR10 / USCI0_CTL0 / UART2_TXD / BPWM0_CH4 / CLKO / ADC0_ST
18	PD.12 / EBI_nCS0 / UART2_RXD / BPWM0_CH5 / CLKO / ADC0_ST / INT5
19	PD.11 / EBI_nCS1 / UART1_TXD
20	PD.10 / UART1_RXD
21	PG.2 / EBI_ADR11 / I2C0_SMBAL / I2C1_SCL / TM0
22	PG.3 / EBI_ADR12 / I2C0_SMBSUS / I2C1_SDA / TM1
23	PG.4 / EBI_ADR13 / TM2
24	PF.11 / EBI_ADR14 / UART5_TXD / TM3
25	PF.10 / EBI_ADR15 / SPI0_I2SMCLK / UART5_RXD
26	PF.9 / EBI_ADR16 / SPI0_SS / UART5_nRTS
27	PF.8 / EBI_ADR17 / SPI0_CLK / UART5_nCTS
28	PF.7 / EBI_ADR18 / SPI0_MISO / UART4_TXD
29	PF.6 / EBI_ADR19 / SPI0_MOSI / UART4_RXD / EBI_nCS0
30	PF.14 / PWM1_BRAKE0 / PWM0_BRAKE0 / PWM0_CH4 / CLKO / TM3 / INT5
31	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / BPWM0_CH4 / X32_IN / ADC0_ST
32	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1 / BPWM0_CH5 / X32_OUT
33	PH.4 / EBI_ADR3
34	PH.5 / EBI_ADR2

Pin	M031KG6AE Pin Function
35	PH.6 / EBI_ADR1
36	PH.7 / EBI_ADR0
37	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / BPWM1_CH0
38	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / BPWM1_CH1
39	VSS
40	V_{DD}
41	PE.8 / EBI_ADR10 / USCI1_CTL1 / UART2_TXD / PWM0_CH0 / PWM0_BRAKE0
42	PE.9 / EBI_ADR11 / USCI1_CTL0 / UART2_RXD / PWM0_CH1 / PWM0_BRAKE1
43	PE.10 / EBI_ADR12 / USCI1_DAT0 / UART3_TXD / PWM0_CH2 / PWM1_BRAKE0
44	PE.11 / EBI_ADR13 / USCI1_DAT1 / UART3_RXD / UART1_nCTS / PWM0_CH3 / PWM1_BRAKE1
45	PE.12 / EBI_ADR14 / USCI1_CLK / UART1_nRTS / PWM0_CH4
46	PE.13 / EBI_ADR15 / I2C0_SCL / UART4_nRTS / UART1_TXD / PWM0_CH5 / PWM1_CH0 / BPWM1_CH5
47	PC.8 / EBI_ADR16 / I2C0_SDA / UART4_nCTS / UART1_RXD / PWM1_CH1 / BPWM1_CH4
48	PC.7 / EBI_AD9 / UART4_TXD / UART0_nCTS / PWM1_CH2 / BPWM1_CH0 / TM0 / INT3
49	PC.6 / EBI_AD8 / UART4_RXD / UART0_nRTS / PWM1_CH3 / BPWM1_CH1 / TM1 / INT2
50	PA.7 / EBI_AD7 / UART0_TXD / I2C1_SCL / PWM1_CH4 / BPWM1_CH2 / ACMP0_WLAT / TM2 / INT1
51	PA.6 / EBI_AD6 / UART0_RXD / I2C1_SDA / PWM1_CH5 / BPWM1_CH3 / ACMP1_WLAT / TM3 / INT0
52	VSS
53	V_{DD}
54	PD.15 / PWM0_CH5 / TM3 / INT1
55	PA.5 / QSPI0_MISO1 / UART0_nCTS / UART0_TXD / I2C0_SCL / UART5_TXD / BPWM0_CH5 / PWM0_CH0
56	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / UART5_RXD / BPWM0_CH4 / PWM0_CH1
57	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / I2C0_SMBAL / UART1_TXD / I2C1_SCL / BPWM0_CH3 / PWM0_CH2 / CLKO / PWM1_BRAKE1
58	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / BPWM0_CH2 / PWM0_CH3
59	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / BPWM0_CH1 / PWM0_CH4
60	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / BPWM0_CH0 / PWM0_CH5
61	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / TM2 / CLKO / INT4
62	PE.14 / EBI_AD8 / UART2_TXD
63	PE.15 / EBI_AD9 / UART2_RXD
64	nRESET
65	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / BPWM1_CH0 / ICE_DAT
66	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / BPWM1_CH1 / ICE_CLK
67	PD.9 / EBI_AD7 / UART2_nCTS

Pin	M031KG6AE Pin Function
68	PD.8 / EBI_AD6 / UART2_nRTS
69	PC.5 / EBI_AD5 / QSPI0_MISO1 / UART2_TXD / I2C1_SCL / UART4_TXD / PWM1_CH0
70	PC.4 / EBI_AD4 / QSPI0_MOSI1 / UART2_RXD / I2C1_SDA / UART4_RXD / PWM1_CH1
71	PC.3 / EBI_AD3 / QSPI0_SS / UART2_nRTS / I2C0_SMBAL / UART3_TXD / PWM1_CH2
72	PC.2 / EBI_AD2 / QSPI0_CLK / UART2_nCTS / I2C0_SMBSUS / UART3_RXD / PWM1_CH3
73	PC.1 / EBI_AD1 / QSPI0_MISO0 / UART2_TXD / I2C0_SCL / PWM1_CH4 / ACMP0_O / ADC0_ST
74	PC.0 / EBI_AD0 / QSPI0_MOSI0 / UART2_RXD / I2C0_SDA / PWM1_CH5 / ACMP1_O
75	VSS
76	V_{DD}
77	PG.9 / EBI_AD0 / BPWM0_CH5
78	PG.10 / EBI_AD1 / BPWM0_CH4
79	PG.11 / EBI_AD2 / BPWM0_CH3
80	PG.12 / EBI_AD3 / BPWM0_CH2
81	PG.13 / EBI_AD4 / BPWM0_CH1
82	PG.14 / EBI_AD5 / BPWM0_CH0
83	PG.15 / CLKO / ADC0_ST
84	PD.7 / UART1_TXD / I2C0_SCL / USCI1_CLK
85	PD.6 / UART1_RXD / I2C0_SDA / USCI1_DAT1
86	PD.5 / I2C1_SCL / USCI1_DAT0
87	PD.4 / USCI0_CTL0 / I2C1_SDA / USCI1_CTL1
88	PD.3 / EBI_AD10 / USCI0_CTL1 / SPI0_SS / UART3_nRTS / USCI1_CTL0 / UART0_TXD
89	PD.2 / EBI_AD11 / USCI0_DAT1 / SPI0_CLK / UART3_nCTS / UART0_RXD
90	PD.1 / EBI_AD12 / USCI0_DAT0 / SPI0_MISO / UART3_TXD
91	PD.0 / EBI_AD13 / USCI0_CLK / SPI0_MOSI / UART3_RXD / TM2
92	PD.13 / EBI_AD10 / SPI0_I2SMCLK
93	PA.12 / UART4_TXD / I2C1_SCL / BPWM1_CH2
94	PA.13 / UART4_RXD / I2C1_SDA / BPWM1_CH3
95	PA.14 / UART0_TXD / BPWM1_CH4
96	PA.15 / UART0_RXD / BPWM1_CH5
97	PE.7 / UART5_TXD / PWM0_CH0 / BPWM0_CH5
98	PE.6 / USCI0_CTL0 / UART5_RXD / PWM0_CH1 / BPWM0_CH4
99	PE.5 / EBI_nRD / USCI0_CTL1 / PWM0_CH2 / BPWM0_CH3
100	PE.4 / EBI_nWR / USCI0_DAT1 / PWM0_CH3 / BPWM0_CH2
101	PE.3 / EBI_MCLK / USCI0_DAT0 / PWM0_CH4 / BPWM0_CH1

Pin	M031KG6AE Pin Function
102	PE.2 / EBI_ALE / USCI0_CLK / PWM0_CH5 / BPWM0_CH0
103	VSS
104	V_{DD}
105	PE.1 / EBI_AD10 / QSPI0_MISO0 / UART3_TXD / I2C1_SCL / UART4_nCTS
106	PE.0 / EBI_AD11 / QSPI0_MOSI0 / UART3_RXD / I2C1_SDA / UART4_nRTS
107	PH.8 / EBI_AD12 / QSPI0_CLK / UART3_nRTS / UART1_TXD
108	PH.9 / EBI_AD13 / QSPI0_SS / UART3_nCTS / UART1_RXD
109	PH.10 / EBI_AD14 / QSPI0_MISO1 / UART4_TXD / UART0_TXD
110	PH.11 / EBI_AD15 / QSPI0_MOSI1 / UART4_RXD / UART0_RXD / PWM0_CH5
111	PD.14 / EBI_nCS0 / SPI0_I2SMCLK / USCI0_CTL0 / PWM0_CH4
112	VSS
113	LDO_CAP
114	V_{DD}
115	PC.14 / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / TM1
116	PB.15 / ADC0_CH15 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / PWM1_CH0 / TM0_EXT / PWM0_BRAKE1
117	PB.14 / ADC0_CH14 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / PWM1_CH1 / TM1_EXT / CLKO
118	PB.13 / ADC0_CH13 / ACMP0_P3 / ACMP1_P3 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / PWM1_CH2 / TM2_EXT
119	PB.12 / ADC0_CH12 / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / PWM1_CH3 / TM3_EXT
120	AV_{DD}
121	V_{REF}
122	AVSS
123	PB.11 / ADC0_CH11 / EBI_ADR16 / UART0_nCTS / UART4_TXD / I2C1_SCL / SPI0_I2SMCLK / BPWM1_CH0
124	PB.10 / ADC0_CH10 / EBI_ADR17 / USCI1_CTL0 / UART0_nRTS / UART4_RXD / I2C1_SDA / BPWM1_CH1
125	PB.9 / ADC0_CH9 / EBI_ADR18 / USCI1_CTL1 / UART0_TXD / UART1_nCTS / BPWM1_CH2
126	PB.8 / ADC0_CH8 / EBI_ADR19 / USCI1_CLK / UART0_RXD / UART1_nRTS / BPWM1_CH3
127	PB.7 / ADC0_CH7 / EBI_nWRL / USCI1_DAT0 / UART1_TXD / EBI_nCS0 / BPWM1_CH4 / PWM1_BRAKE0 / PWM1_CH4 / INT5 / ACMP0_O
128	PB.6 / ADC0_CH6 / EBI_nWRH / USCI1_DAT1 / UART1_RXD / EBI_nCS1 / BPWM1_CH5 / PWM1_BRAKE1 / PWM1_CH5 / INT4 / ACMP1_O

Table 4.1-20 M031KG6AE Multi-function Pin Table

M031KG8AE

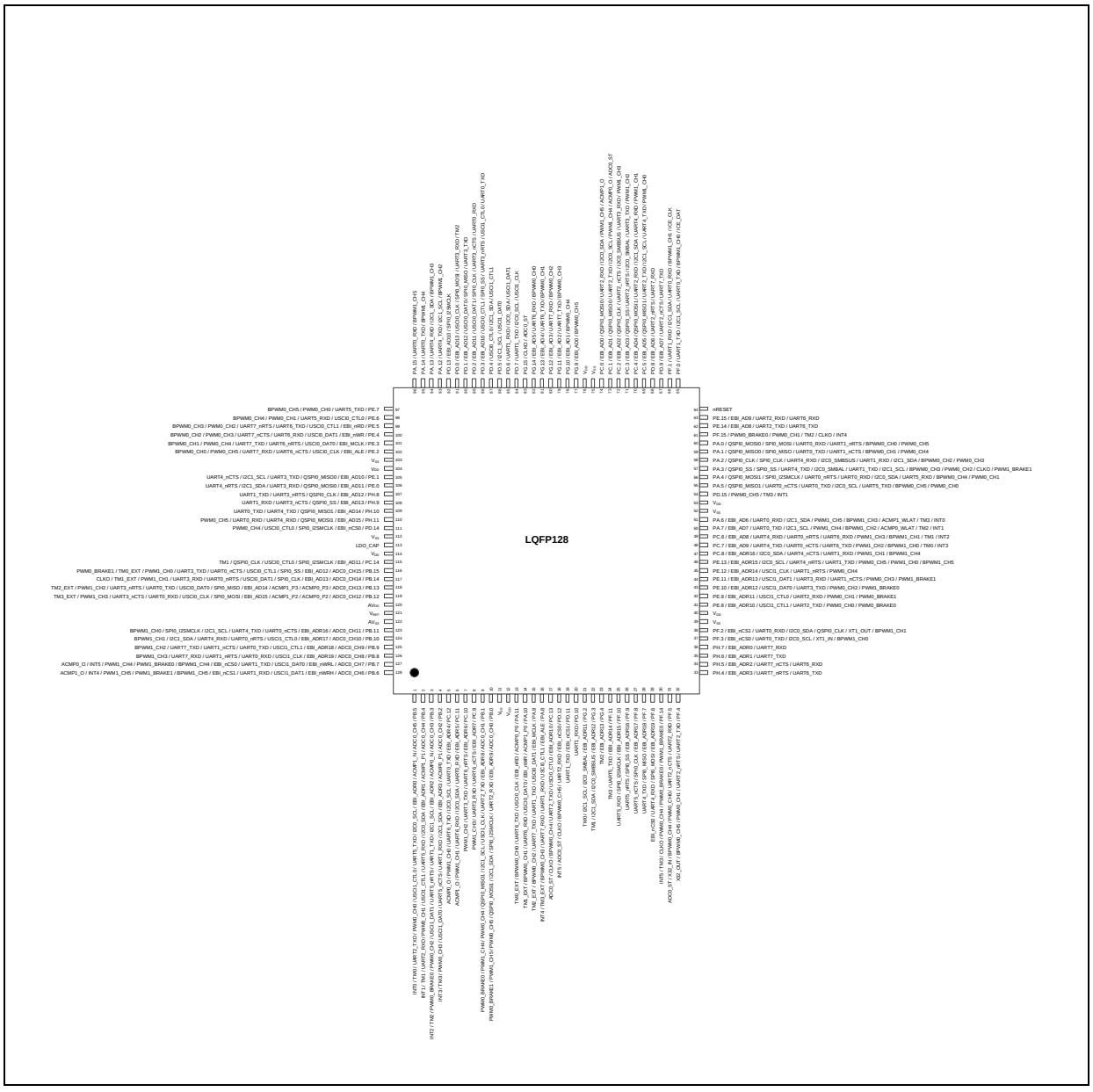


Figure 4.1-27 M031KG8AE Multi-function Pin Diagram

Pin	M031KG8AE Pin Function
1	PB.5 / ADC0_CH5 / ACMP1_N / EBI_ADR0 / I2C0_SCL / UART5_TXD / USCI1_CTL0 / PWM0_CH0 / UART2_TXD / TM0 / INT0
2	PB.4 / ADC0_CH4 / ACMP1_P1 / EBI_ADR1 / I2C0_SDA / UART5_RXD / USCI1_CTL1 / PWM0_CH1 / UART2_RXD / TM1 / INT1
3	PB.3 / ADC0_CH3 / ACMP0_N / EBI_ADR2 / I2C1_SCL / UART1_TXD / UART5_nRTS / USCI1_DAT1 / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2
4	PB.2 / ADC0_CH2 / ACMP0_P1 / EBI_ADR3 / I2C1_SDA / UART1_RXD / UART5_nCTS / USCI1_DAT0 /

Pin	M031KG8AE Pin Function
	PWM0_CH3 / TM3 / INT3
5	PC.12 / EBI_ADR4 / UART0_TXD / I2C0_SCL / PWM1_CH0 / ACMP0_O
6	PC.11 / EBI_ADR5 / UART0_RXD / I2C0_SDA / PWM1_CH1 / ACMP1_O
7	PC.10 / EBI_ADR6 / UART3_TXD / PWM1_CH2
8	PC.9 / EBI_ADR7 / UART3_RXD / PWM1_CH3
9	PB.1 / ADC0_CH1 / EBI_ADR8 / UART2_TXD / USCI1_CLK / I2C1_SCL / QSPI0_MISO1 / PWM0_CH4 / PWM1_CH4 / PWM0_BRAKE0
10	PB.0 / ADC0_CH0 / EBI_ADR9 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / PWM0_CH5 / PWM1_CH5 / PWM0_BRAKE1
11	VSS
12	V_{DD}
13	PA.11 / ACMP0_P0 / EBI_nRD / USCI0_CLK / BPWM0_CH0 / TM0_EXT
14	PA.10 / ACMP1_P0 / EBI_nWR / USCI0_DAT0 / BPWM0_CH1 / TM1_EXT
15	PA.9 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / BPWM0_CH2 / TM2_EXT
16	PA.8 / EBI_ALE / USCI0_CTL1 / UART1_RXD / BPWM0_CH3 / TM3_EXT / INT4
17	PC.13 / EBI_ADR10 / USCI0_CTL0 / UART2_TXD / BPWM0_CH4 / CLKO / ADC0_ST
18	PD.12 / EBI_nCS0 / UART2_RXD / BPWM0_CH5 / CLKO / ADC0_ST / INT5
19	PD.11 / EBI_nCS1 / UART1_TXD
20	PD.10 / UART1_RXD
21	PG.2 / EBI_ADR11 / I2C0_SMBAL / I2C1_SCL / TM0
22	PG.3 / EBI_ADR12 / I2C0_SMBSUS / I2C1_SDA / TM1
23	PG.4 / EBI_ADR13 / TM2
24	PF.11 / EBI_ADR14 / UART5_TXD / TM3
25	PF.10 / EBI_ADR15 / SPI0_I2SMCLK / UART5_RXD
26	PF.9 / EBI_ADR16 / SPI0_SS / UART5_nRTS
27	PF.8 / EBI_ADR17 / SPI0_CLK / UART5_nCTS
28	PF.7 / EBI_ADR18 / SPI0_MISO / UART4_TXD
29	PF.6 / EBI_ADR19 / SPI0_MOSI / UART4_RXD / EBI_nCS0
30	PF.14 / PWM1_BRAKE0 / PWM0_BRAKE0 / PWM0_CH4 / CLKO / TM3 / INT5
31	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / BPWM0_CH4 / X32_IN / ADC0_ST
32	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1 / BPWM0_CH5 / X32_OUT
33	PH.4 / EBI_ADR3
34	PH.5 / EBI_ADR2
35	PH.6 / EBI_ADR1
36	PH.7 / EBI_ADR0
37	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / BPWM1_CH0

Pin	M031KG8AE Pin Function
38	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / BPWM1_CH1
39	VSS
40	V_{DD}
41	PE.8 / EBI_ADR10 / USCI1_CTL1 / UART2_TXD / PWM0_CH0 / PWM0_BRAKE0
42	PE.9 / EBI_ADR11 / USCI1_CTL0 / UART2_RXD / PWM0_CH1 / PWM0_BRAKE1
43	PE.10 / EBI_ADR12 / USCI1_DAT0 / UART3_TXD / PWM0_CH2 / PWM1_BRAKE0
44	PE.11 / EBI_ADR13 / USCI1_DAT1 / UART3_RXD / UART1_nCTS / PWM0_CH3 / PWM1_BRAKE1
45	PE.12 / EBI_ADR14 / USCI1_CLK / UART1_nRTS / PWM0_CH4
46	PE.13 / EBI_ADR15 / I2C0_SCL / UART4_nRTS / UART1_TXD / PWM0_CH5 / PWM1_CH0 / BPWM1_CH5
47	PC.8 / EBI_ADR16 / I2C0_SDA / UART4_nCTS / UART1_RXD / PWM1_CH1 / BPWM1_CH4
48	PC.7 / EBI_AD9 / UART4_TXD / UART0_nCTS / PWM1_CH2 / BPWM1_CH0 / TM0 / INT3
49	PC.6 / EBI_AD8 / UART4_RXD / UART0_nRTS / PWM1_CH3 / BPWM1_CH1 / TM1 / INT2
50	PA.7 / EBI_AD7 / UART0_TXD / I2C1_SCL / PWM1_CH4 / BPWM1_CH2 / ACMP0_WLAT / TM2 / INT1
51	PA.6 / EBI_AD6 / UART0_RXD / I2C1_SDA / PWM1_CH5 / BPWM1_CH3 / ACMP1_WLAT / TM3 / INT0
52	VSS
53	V_{DD}
54	PD.15 / PWM0_CH5 / TM3 / INT1
55	PA.5 / QSPI0_MISO1 / UART0_nCTS / UART0_TXD / I2C0_SCL / UART5_TXD / BPWM0_CH5 / PWM0_CH0
56	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / UART5_RXD / BPWM0_CH4 / PWM0_CH1
57	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / I2C0_SMBAL / UART1_TXD / I2C1_SCL / BPWM0_CH3 / PWM0_CH2 / CLKO / PWM1_BRAKE1
58	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / BPWM0_CH2 / PWM0_CH3
59	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / BPWM0_CH1 / PWM0_CH4
60	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / BPWM0_CH0 / PWM0_CH5
61	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / TM2 / CLKO / INT4
62	PE.14 / EBI_AD8 / UART2_TXD
63	PE.15 / EBI_AD9 / UART2_RXD
64	nRESET
65	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / BPWM1_CH0 / ICE_DAT
66	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / BPWM1_CH1 / ICE_CLK
67	PD.9 / EBI_AD7 / UART2_nCTS
68	PD.8 / EBI_AD6 / UART2_nRTS
69	PC.5 / EBI_AD5 / QSPI0_MISO1 / UART2_TXD / I2C1_SCL / UART4_TXD / PWM1_CH0
70	PC.4 / EBI_AD4 / QSPI0_MOSI1 / UART2_RXD / I2C1_SDA / UART4_RXD / PWM1_CH1

Pin	M031KG8AE Pin Function
71	PC.3 / EBI_AD3 / QSPI0_SS / UART2_nRTS / I2C0_SMBAL / UART3_TXD / PWM1_CH2
72	PC.2 / EBI_AD2 / QSPI0_CLK / UART2_nCTS / I2C0_SMBSUS / UART3_RXD / PWM1_CH3
73	PC.1 / EBI_AD1 / QSPI0_MISO0 / UART2_TXD / I2C0_SCL / PWM1_CH4 / ACMP0_O / ADC0_ST
74	PC.0 / EBI_AD0 / QSPI0_MOSI0 / UART2_RXD / I2C0_SDA / PWM1_CH5 / ACMP1_O
75	VSS
76	V_{DD}
77	PG.9 / EBI_AD0 / BPWM0_CH5
78	PG.10 / EBI_AD1 / BPWM0_CH4
79	PG.11 / EBI_AD2 / BPWM0_CH3
80	PG.12 / EBI_AD3 / BPWM0_CH2
81	PG.13 / EBI_AD4 / BPWM0_CH1
82	PG.14 / EBI_AD5 / BPWM0_CH0
83	PG.15 / CLKO / ADC0_ST
84	PD.7 / UART1_TXD / I2C0_SCL / USCI1_CLK
85	PD.6 / UART1_RXD / I2C0_SDA / USCI1_DAT1
86	PD.5 / I2C1_SCL / USCI1_DAT0
87	PD.4 / USCI0_CTL0 / I2C1_SDA / USCI1_CTL1
88	PD.3 / EBI_AD10 / USCI0_CTL1 / SPI0_SS / UART3_nRTS / USCI1_CTL0 / UART0_TXD
89	PD.2 / EBI_AD11 / USCI0_DAT1 / SPI0_CLK / UART3_nCTS / UART0_RXD
90	PD.1 / EBI_AD12 / USCI0_DAT0 / SPI0_MISO / UART3_TXD
91	PD.0 / EBI_AD13 / USCI0_CLK / SPI0_MOSI / UART3_RXD / TM2
92	PD.13 / EBI_AD10 / SPI0_I2SMCLK
93	PA.12 / UART4_TXD / I2C1_SCL / BPWM1_CH2
94	PA.13 / UART4_RXD / I2C1_SDA / BPWM1_CH3
95	PA.14 / UART0_TXD / BPWM1_CH4
96	PA.15 / UART0_RXD / BPWM1_CH5
97	PE.7 / UART5_TXD / PWM0_CH0 / BPWM0_CH5
98	PE.6 / USCI0_CTL0 / UART5_RXD / PWM0_CH1 / BPWM0_CH4
99	PE.5 / EBI_nRD / USCI0_CTL1 / PWM0_CH2 / BPWM0_CH3
100	PE.4 / EBI_nWR / USCI0_DAT1 / PWM0_CH3 / BPWM0_CH2
101	PE.3 / EBI_MCLK / USCI0_DAT0 / PWM0_CH4 / BPWM0_CH1
102	PE.2 / EBI_ALE / USCI0_CLK / PWM0_CH5 / BPWM0_CH0
103	VSS
104	V_{DD}

Pin	M031KG8AE Pin Function
105	PE.1 / EBI_AD10 / QSPI0_MISO0 / UART3_TXD / I2C1_SCL / UART4_nCTS
106	PE.0 / EBI_AD11 / QSPI0_MOSI0 / UART3_RXD / I2C1_SDA / UART4_nRTS
107	PH.8 / EBI_AD12 / QSPI0_CLK / UART3_nRTS / UART1_TXD
108	PH.9 / EBI_AD13 / QSPI0_SS / UART3_nCTS / UART1_RXD
109	PH.10 / EBI_AD14 / QSPI0_MISO1 / UART4_TXD / UART0_TXD
110	PH.11 / EBI_AD15 / QSPI0_MOSI1 / UART4_RXD / UART0_RXD / PWM0_CH5
111	PD.14 / EBI_nCS0 / SPI0_I2SMCLK / USCI0_CTL0 / PWM0_CH4
112	VSS
113	LDO_CAP
114	V_{DD}
115	PC.14 / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / TM1
116	PB.15 / ADC0_CH15 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / PWM1_CH0 / TM0_EXT / PWM0_BRAKE1
117	PB.14 / ADC0_CH14 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / PWM1_CH1 / TM1_EXT / CLKO
118	PB.13 / ADC0_CH13 / ACMP0_P3 / ACMP1_P3 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / PWM1_CH2 / TM2_EXT
119	PB.12 / ADC0_CH12 / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / PWM1_CH3 / TM3_EXT
120	AV_{DD}
121	V_{REF}
122	AVSS
123	PB.11 / ADC0_CH11 / EBI_ADR16 / UART0_nCTS / UART4_TXD / I2C1_SCL / SPI0_I2SMCLK / BPWM1_CH0
124	PB.10 / ADC0_CH10 / EBI_ADR17 / USCI1_CTL0 / UART0_nRTS / UART4_RXD / I2C1_SDA / BPWM1_CH1
125	PB.9 / ADC0_CH9 / EBI_ADR18 / USCI1_CTL1 / UART0_TXD / UART1_nCTS / BPWM1_CH2
126	PB.8 / ADC0_CH8 / EBI_ADR19 / USCI1_CLK / UART0_RXD / UART1_nRTS / BPWM1_CH3
127	PB.7 / ADC0_CH7 / EBI_nWRL / USCI1_DAT0 / UART1_TXD / EBI_nCS0 / BPWM1_CH4 / PWM1_BRAKE0 / PWM1_CH4 / INT5 / ACMP0_O
128	PB.6 / ADC0_CH6 / EBI_nWRH / USCI1_DAT1 / UART1_RXD / EBI_nCS1 / BPWM1_CH5 / PWM1_BRAKE1 / PWM1_CH5 / INT4 / ACMP1_O

Table 4.1-21 M031KG8AE Multi-function Pin Table

M031KIAAE

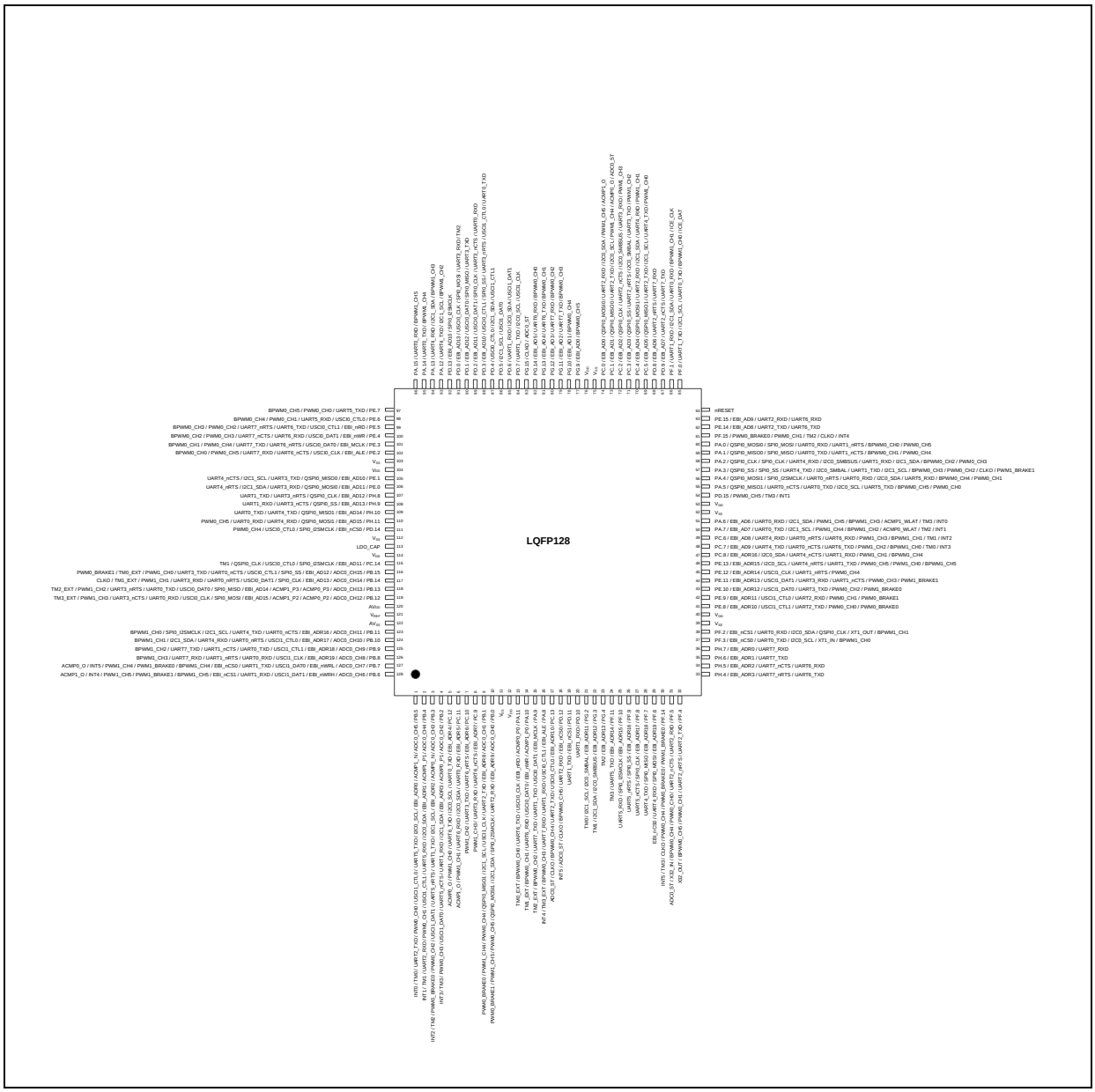


Figure 4.1-28 M031KIAAE Multi-function Pin Diagram

Pin	M031KIAAE Pin Function
1	PB.5 / ADC0_CH5 / ACMP1_N / EBI_ADR0 / I2C0_SCL / UART5_TXD / USCI1_CTL0 / PWM0_CH0 / UART2_TXD / TM0 / INT0
2	PB.4 / ADC0_CH4 / ACMP1_P1 / EBI_ADR1 / I2C0_SDA / UART5_RXD / USCI1_CTL1 / PWM0_CH1 / UART2_RXD / TM1 / INT1
3	PB.3 / ADC0_CH3 / ACMP0_N / EBI_ADR2 / I2C1_SCL / UART1_TXD / UART5_nRTS / USCI1_DAT1 / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2
4	PB.2 / ADC0_CH2 / ACMP0_P1 / EBI_ADR3 / I2C1_SDA / UART1_RXD / UART5_nCTS / USCI1_DAT0 /

Pin	M031KIAAE Pin Function
	PWM0_CH3 / TM3 / INT3
5	PC.12 / EBI_ADR4 / UART0_TXD / I2C0_SCL / UART6_TXD / PWM1_CH0 / ACMP0_O
6	PC.11 / EBI_ADR5 / UART0_RXD / I2C0_SDA / UART6_RXD / PWM1_CH1 / ACMP1_O
7	PC.10 / EBI_ADR6 / UART6_nRTS / UART3_TXD / PWM1_CH2
8	PC.9 / EBI_ADR7 / UART6_nCTS / UART3_RXD / PWM1_CH3
9	PB.1 / ADC0_CH1 / EBI_ADR8 / UART2_TXD / USCI1_CLK / I2C1_SCL / QSPI0_MISO1 / PWM0_CH4 / PWM1_CH4 / PWM0_BRAKE0
10	PB.0 / ADC0_CH0 / EBI_ADR9 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / PWM0_CH5 / PWM1_CH5 / PWM0_BRAKE1
11	VSS
12	V_{DD}
13	PA.11 / ACMP0_P0 / EBI_nRD / USCI0_CLK / UART6_TXD / BPWM0_CH0 / TM0_EXT
14	PA.10 / ACMP1_P0 / EBI_nWR / USCI0_DAT0 / UART6_RXD / BPWM0_CH1 / TM1_EXT
15	PA.9 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / UART7_TXD / BPWM0_CH2 / TM2_EXT
16	PA.8 / EBI_ALE / USCI0_CTL1 / UART1_RXD / UART7_RXD / BPWM0_CH3 / TM3_EXT / INT4
17	PC.13 / EBI_ADR10 / USCI0_CTL0 / UART2_TXD / BPWM0_CH4 / CLKO / ADC0_ST
18	PD.12 / EBI_nCS0 / UART2_RXD / BPWM0_CH5 / CLKO / ADC0_ST / INT5
19	PD.11 / EBI_nCS1 / UART1_TXD
20	PD.10 / UART1_RXD
21	PG.2 / EBI_ADR11 / I2C0_SMBAL / I2C1_SCL / TM0
22	PG.3 / EBI_ADR12 / I2C0_SMBSUS / I2C1_SDA / TM1
23	PG.4 / EBI_ADR13 / TM2
24	PF.11 / EBI_ADR14 / UART5_TXD / TM3
25	PF.10 / EBI_ADR15 / SPI0_I2SMCLK / UART5_RXD
26	PF.9 / EBI_ADR16 / SPI0_SS / UART5_nRTS
27	PF.8 / EBI_ADR17 / SPI0_CLK / UART5_nCTS
28	PF.7 / EBI_ADR18 / SPI0_MISO / UART4_TXD
29	PF.6 / EBI_ADR19 / SPI0_MOSI / UART4_RXD / EBI_nCS0
30	PF.14 / PWM1_BRAKE0 / PWM0_BRAKE0 / PWM0_CH4 / CLKO / TM3 / INT5
31	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / BPWM0_CH4 / X32_IN / ADC0_ST
32	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1 / BPWM0_CH5 / X32_OUT
33	PH.4 / EBI_ADR3 / UART7_nRTS / UART6_TXD
34	PH.5 / EBI_ADR2 / UART7_nCTS / UART6_RXD
35	PH.6 / EBI_ADR1 / UART7_TXD
36	PH.7 / EBI_ADR0 / UART7_RXD
37	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / BPWM1_CH0

Pin	M031KIAAE Pin Function
38	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / BPWM1_CH1
39	VSS
40	V_{DD}
41	PE.8 / EBI_ADR10 / USCI1_CTL1 / UART2_TXD / PWM0_CH0 / PWM0_BRAKE0
42	PE.9 / EBI_ADR11 / USCI1_CTL0 / UART2_RXD / PWM0_CH1 / PWM0_BRAKE1
43	PE.10 / EBI_ADR12 / USCI1_DAT0 / UART3_TXD / PWM0_CH2 / PWM1_BRAKE0
44	PE.11 / EBI_ADR13 / USCI1_DAT1 / UART3_RXD / UART1_nCTS / PWM0_CH3 / PWM1_BRAKE1
45	PE.12 / EBI_ADR14 / USCI1_CLK / UART1_nRTS / PWM0_CH4
46	PE.13 / EBI_ADR15 / I2C0_SCL / UART4_nRTS / UART1_TXD / PWM0_CH5 / PWM1_CH0 / BPWM1_CH5
47	PC.8 / EBI_ADR16 / I2C0_SDA / UART4_nCTS / UART1_RXD / PWM1_CH1 / BPWM1_CH4
48	PC.7 / EBI_AD9 / UART4_TXD / UART0_nCTS / UART6_TXD / PWM1_CH2 / BPWM1_CH0 / TM0 / INT3
49	PC.6 / EBI_AD8 / UART4_RXD / UART0_nRTS / UART6_RXD / PWM1_CH3 / BPWM1_CH1 / TM1 / INT2
50	PA.7 / EBI_AD7 / UART0_TXD / I2C1_SCL / PWM1_CH4 / BPWM1_CH2 / ACMP0_WLAT / TM2 / INT1
51	PA.6 / EBI_AD6 / UART0_RXD / I2C1_SDA / PWM1_CH5 / BPWM1_CH3 / ACMP1_WLAT / TM3 / INT0
52	VSS
53	V_{DD}
54	PD.15 / PWM0_CH5 / TM3 / INT1
55	PA.5 / QSPI0_MISO1 / UART0_nCTS / UART0_TXD / I2C0_SCL / UART5_TXD / BPWM0_CH5 / PWM0_CH0
56	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / UART5_RXD / BPWM0_CH4 / PWM0_CH1
57	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / I2C0_SMBAL / UART1_TXD / I2C1_SCL / BPWM0_CH3 / PWM0_CH2 / CLKO / PWM1_BRAKE1
58	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / BPWM0_CH2 / PWM0_CH3
59	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / BPWM0_CH1 / PWM0_CH4
60	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / BPWM0_CH0 / PWM0_CH5
61	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / TM2 / CLKO / INT4
62	PE.14 / EBI_AD8 / UART2_TXD / UART6_TXD
63	PE.15 / EBI_AD9 / UART2_RXD / UART6_RXD
64	nRESET
65	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / BPWM1_CH0 / ICE_DAT
66	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / BPWM1_CH1 / ICE_CLK
67	PD.9 / EBI_AD7 / UART2_nCTS / UART7_TXD
68	PD.8 / EBI_AD6 / UART2_nRTS / UART7_RXD
69	PC.5 / EBI_AD5 / QSPI0_MISO1 / UART2_TXD / I2C1_SCL / UART4_TXD / PWM1_CH0
70	PC.4 / EBI_AD4 / QSPI0_MOSI1 / UART2_RXD / I2C1_SDA / UART4_RXD / PWM1_CH1

Pin	M031KIAAE Pin Function
71	PC.3 / EBI_AD3 / QSPI0_SS / UART2_nRTS / I2C0_SMBAL / UART3_TXD / PWM1_CH2
72	PC.2 / EBI_AD2 / QSPI0_CLK / UART2_nCTS / I2C0_SMBSUS / UART3_RXD / PWM1_CH3
73	PC.1 / EBI_AD1 / QSPI0_MISO0 / UART2_TXD / I2C0_SCL / PWM1_CH4 / ACMP0_O / ADC0_ST
74	PC.0 / EBI_AD0 / QSPI0_MOSI0 / UART2_RXD / I2C0_SDA / PWM1_CH5 / ACMP1_O
75	VSS
76	V_{DD}
77	PG.9 / EBI_AD0 / BPWM0_CH5
78	PG.10 / EBI_AD1 / BPWM0_CH4
79	PG.11 / EBI_AD2 / UART7_TXD / BPWM0_CH3
80	PG.12 / EBI_AD3 / UART7_RXD / BPWM0_CH2
81	PG.13 / EBI_AD4 / UART6_TXD / BPWM0_CH1
82	PG.14 / EBI_AD5 / UART6_RXD / BPWM0_CH0
83	PG.15 / CLKO / ADC0_ST
84	PD.7 / UART1_TXD / I2C0_SCL / USCI1_CLK
85	PD.6 / UART1_RXD / I2C0_SDA / USCI1_DAT1
86	PD.5 / I2C1_SCL / USCI1_DAT0
87	PD.4 / USCI0_CTL0 / I2C1_SDA / USCI1_CTL1
88	PD.3 / EBI_AD10 / USCI0_CTL1 / SPI0_SS / UART3_nRTS / USCI1_CTL0 / UART0_TXD
89	PD.2 / EBI_AD11 / USCI0_DAT1 / SPI0_CLK / UART3_nCTS / UART0_RXD
90	PD.1 / EBI_AD12 / USCI0_DAT0 / SPI0_MISO / UART3_TXD
91	PD.0 / EBI_AD13 / USCI0_CLK / SPI0_MOSI / UART3_RXD / TM2
92	PD.13 / EBI_AD10 / SPI0_I2SMCLK
93	PA.12 / UART4_TXD / I2C1_SCL / BPWM1_CH2
94	PA.13 / UART4_RXD / I2C1_SDA / BPWM1_CH3
95	PA.14 / UART0_TXD / BPWM1_CH4
96	PA.15 / UART0_RXD / BPWM1_CH5
97	PE.7 / UART5_TXD / PWM0_CH0 / BPWM0_CH5
98	PE.6 / USCI0_CTL0 / UART5_RXD / PWM0_CH1 / BPWM0_CH4
99	PE.5 / EBI_nRD / USCI0_CTL1 / UART6_TXD / UART7_nRTS / PWM0_CH2 / BPWM0_CH3
100	PE.4 / EBI_nWR / USCI0_DAT1 / UART6_RXD / UART7_nCTS / PWM0_CH3 / BPWM0_CH2
101	PE.3 / EBI_MCLK / USCI0_DAT0 / UART6_nRTS / UART7_TXD / PWM0_CH4 / BPWM0_CH1
102	PE.2 / EBI_ALE / USCI0_CLK / UART6_nCTS / UART7_RXD / PWM0_CH5 / BPWM0_CH0
103	VSS
104	V_{DD}

Pin	M031KIAAE Pin Function
105	PE.1 / EBI_AD10 / QSPI0_MISO0 / UART3_TXD / I2C1_SCL / UART4_nCTS
106	PE.0 / EBI_AD11 / QSPI0_MOSI0 / UART3_RXD / I2C1_SDA / UART4_nRTS
107	PH.8 / EBI_AD12 / QSPI0_CLK / UART3_nRTS / UART1_TXD
108	PH.9 / EBI_AD13 / QSPI0_SS / UART3_nCTS / UART1_RXD
109	PH.10 / EBI_AD14 / QSPI0_MISO1 / UART4_TXD / UART0_TXD
110	PH.11 / EBI_AD15 / QSPI0_MOSI1 / UART4_RXD / UART0_RXD / PWM0_CH5
111	PD.14 / EBI_nCS0 / SPI0_I2SMCLK / USCI0_CTL0 / PWM0_CH4
112	VSS
113	LDO_CAP
114	V_{DD}
115	PC.14 / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / TM1
116	PB.15 / ADC0_CH15 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / PWM1_CH0 / TM0_EXT / PWM0_BRAKE1
117	PB.14 / ADC0_CH14 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / PWM1_CH1 / TM1_EXT / CLKO
118	PB.13 / ADC0_CH13 / ACMP0_P3 / ACMP1_P3 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / PWM1_CH2 / TM2_EXT
119	PB.12 / ADC0_CH12 / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / PWM1_CH3 / TM3_EXT
120	AV_{DD}
121	V_{REF}
122	AVSS
123	PB.11 / ADC0_CH11 / EBI_ADR16 / UART0_nCTS / UART4_TXD / I2C1_SCL / SPI0_I2SMCLK / BPWM1_CH0
124	PB.10 / ADC0_CH10 / EBI_ADR17 / USCI1_CTL0 / UART0_nRTS / UART4_RXD / I2C1_SDA / BPWM1_CH1
125	PB.9 / ADC0_CH9 / EBI_ADR18 / USCI1_CTL1 / UART0_TXD / UART1_nCTS / UART7_TXD / BPWM1_CH2
126	PB.8 / ADC0_CH8 / EBI_ADR19 / USCI1_CLK / UART0_RXD / UART1_nRTS / UART7_RXD / BPWM1_CH3
127	PB.7 / ADC0_CH7 / EBI_nWRL / USCI1_DAT0 / UART1_TXD / EBI_nCS0 / BPWM1_CH4 / PWM1_BRAKE0 / PWM1_CH4 / INT5 / ACMP0_O
128	PB.6 / ADC0_CH6 / EBI_nWRH / USCI1_DAT1 / UART1_RXD / EBI_nCS1 / BPWM1_CH5 / PWM1_BRAKE1 / PWM1_CH5 / INT4 / ACMP1_O

Table 4.1-22 M031KIAAE Multi-function Pin Table

4.1.3 M032 Series Pin Diagram

4.1.3.1 M032 Series TSSOP 20-Pin Diagram

Corresponding Part Number: M032FC1AE

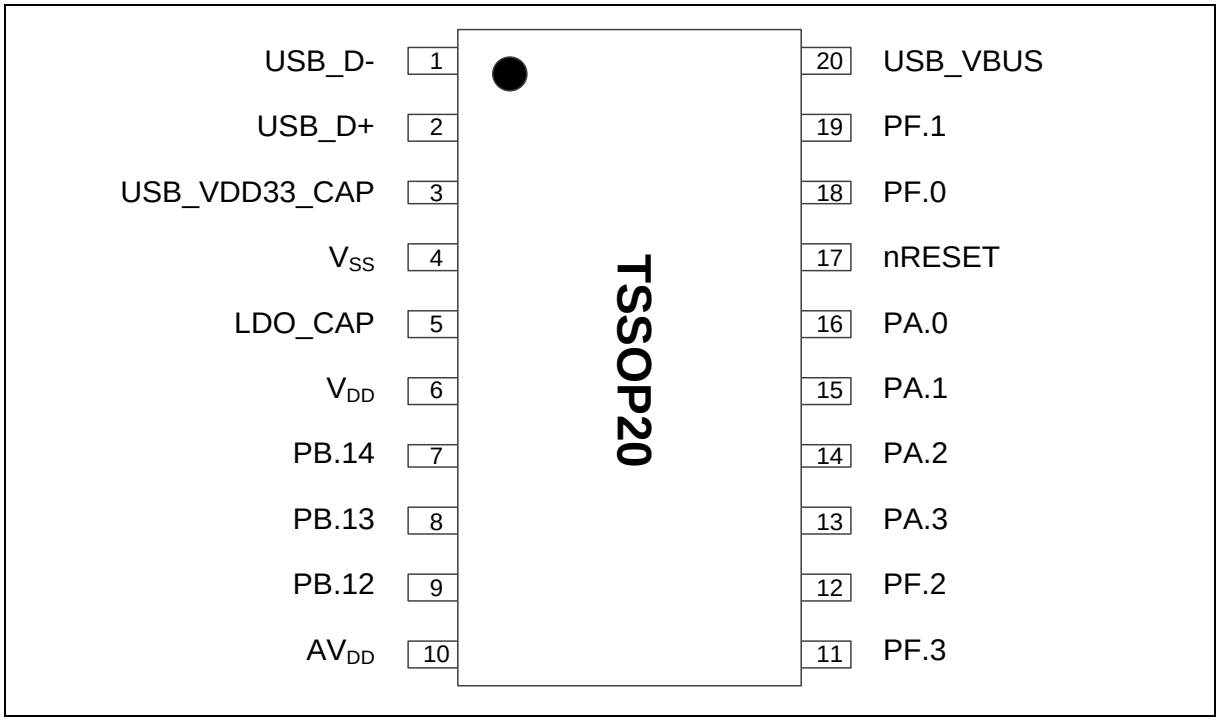


Figure 4.1-29 M032 Series TSSOP 20-pin Diagram

4.1.3.2 M032 Series TSSOP 28-Pin Diagram

Corresponding Part Number: M032EC1AE

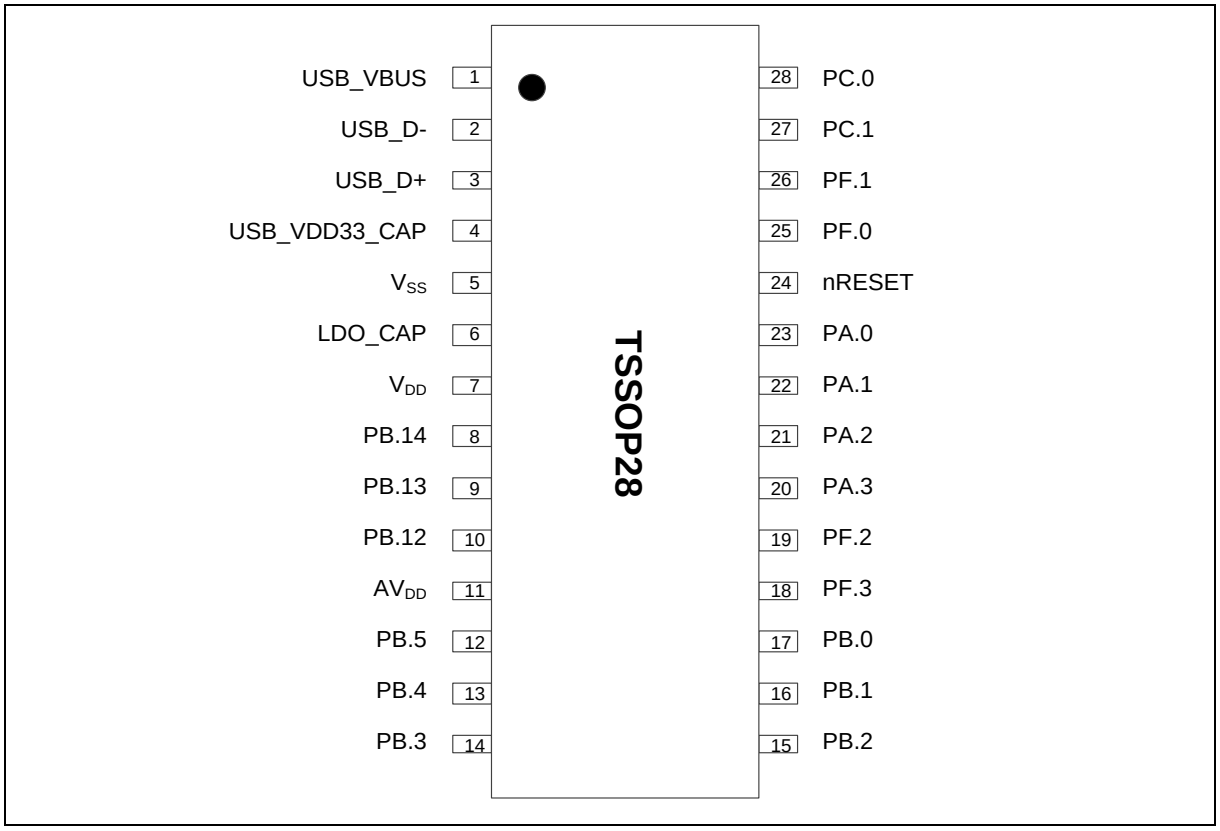


Figure 4.1-30 M032 Series TSSOP 28-pin Diagram

4.1.3.3 M032 Series QFN 33-Pin Diagram

Corresponding Part Number: M032TC1AE, M032TD2AE

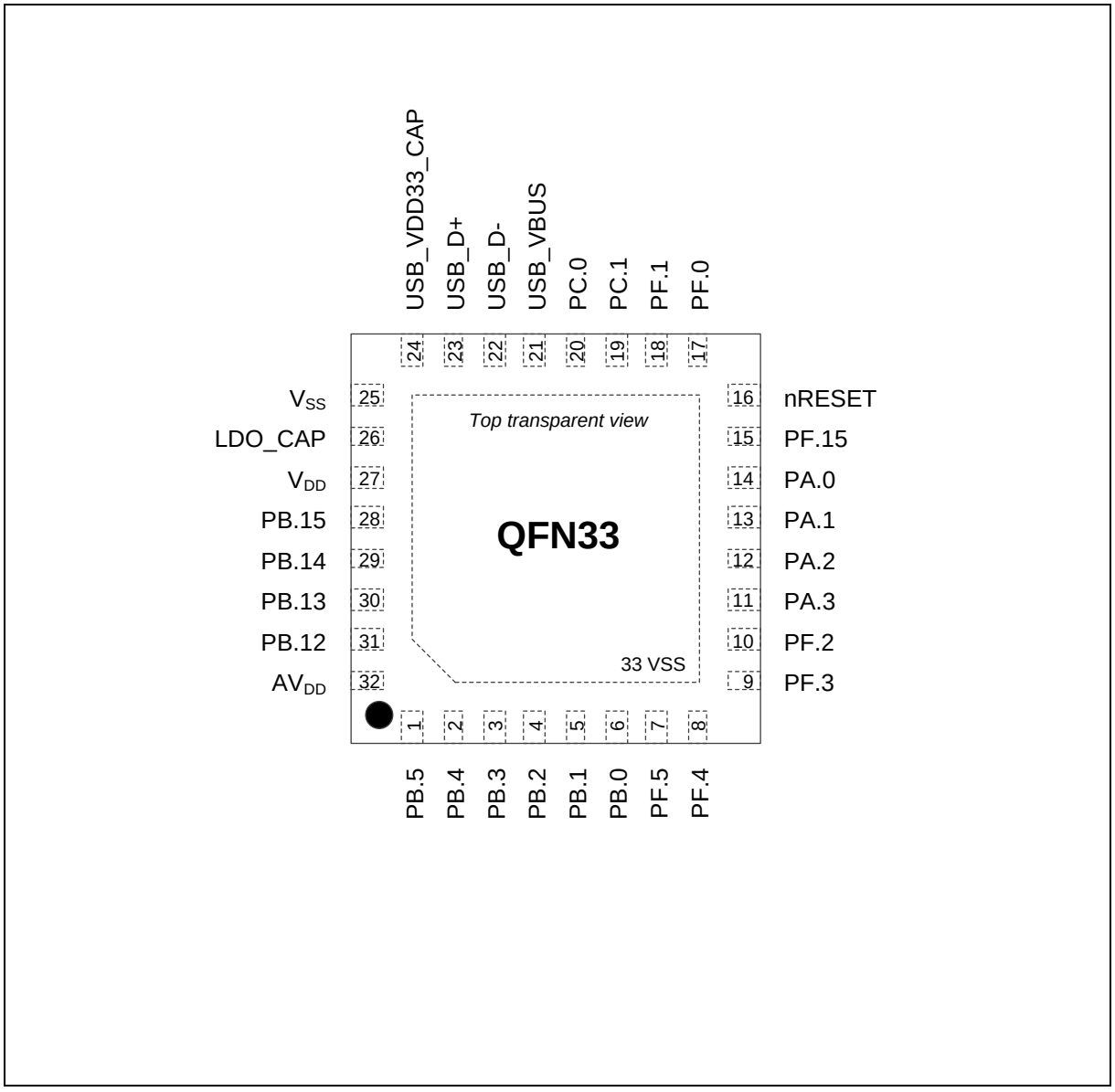


Figure 4.1-31 M032 Series QFN 33-pin Diagram

4.1.3.4 M032 Series LQFP 48-Pin Diagram

Corresponding Part Number: M032LC2AE, M032LD2AE ,M032LE3AE, M032LG6AE, M032LG8AE

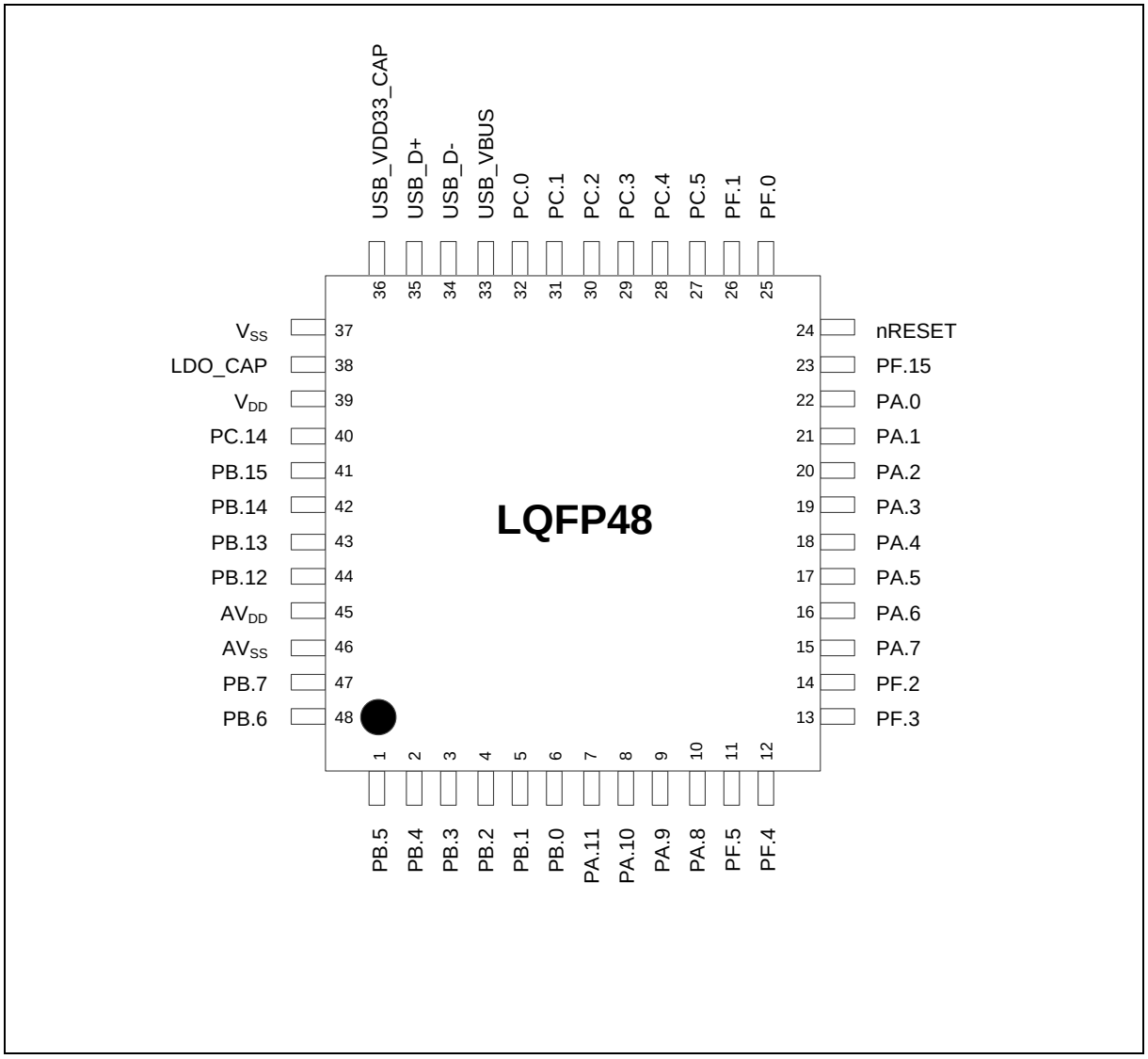


Figure 4.1-32 M032 Series LQFP 48-pin Diagram

4.1.3.5 M032 Series LQFP 64-Pin Diagram

Corresponding Part Number: M032SE3AE, M032SG6AE, M032SG8AE, M032SIAAE

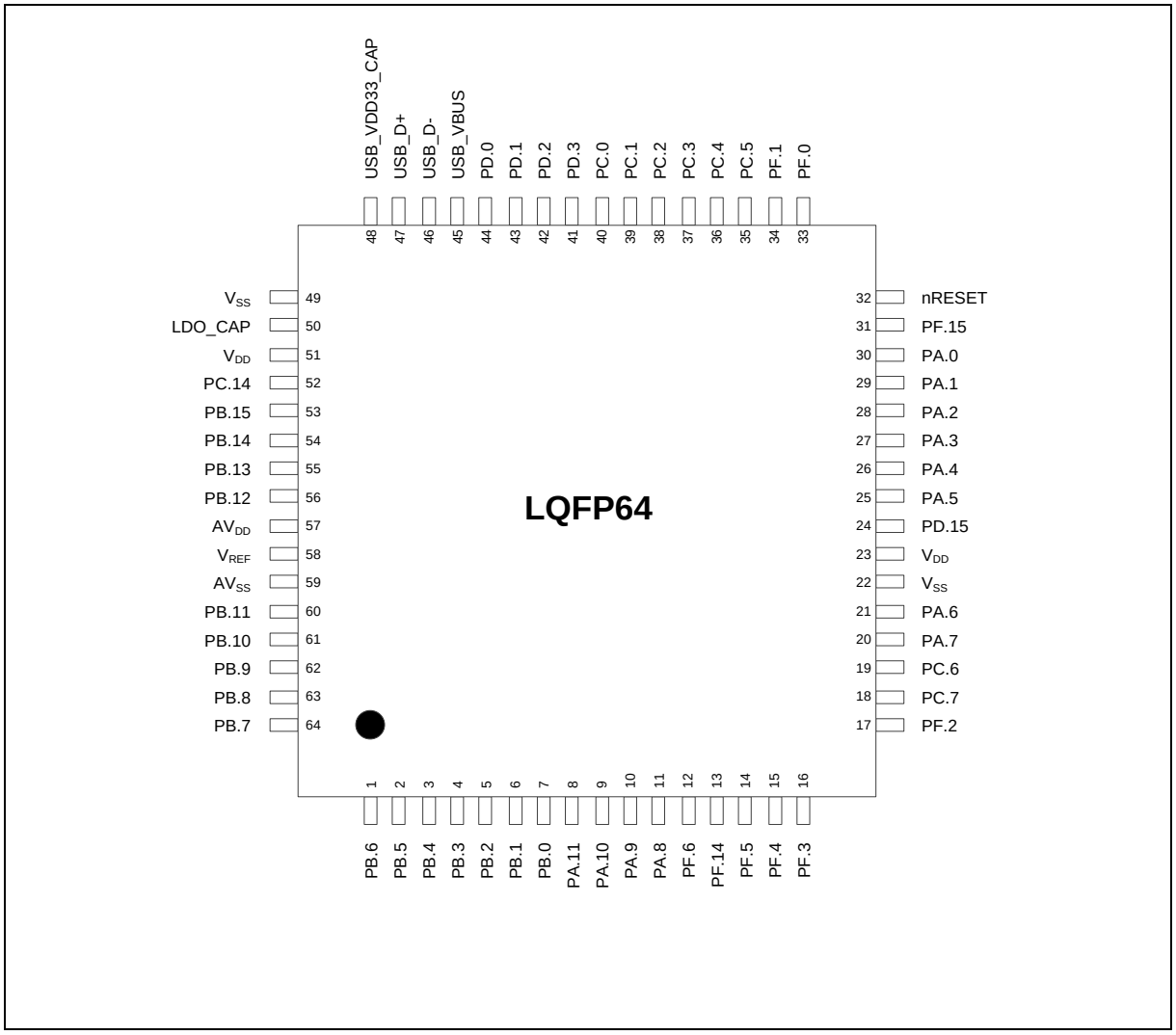


Figure 4.1-33 M032 Series LQFP 64-pin Diagram

4.1.3.6 M032 Series LQFP 128-Pin Diagram

Corresponding Part Number: M032KG6AE, M032KG8AE, M032KIAAE

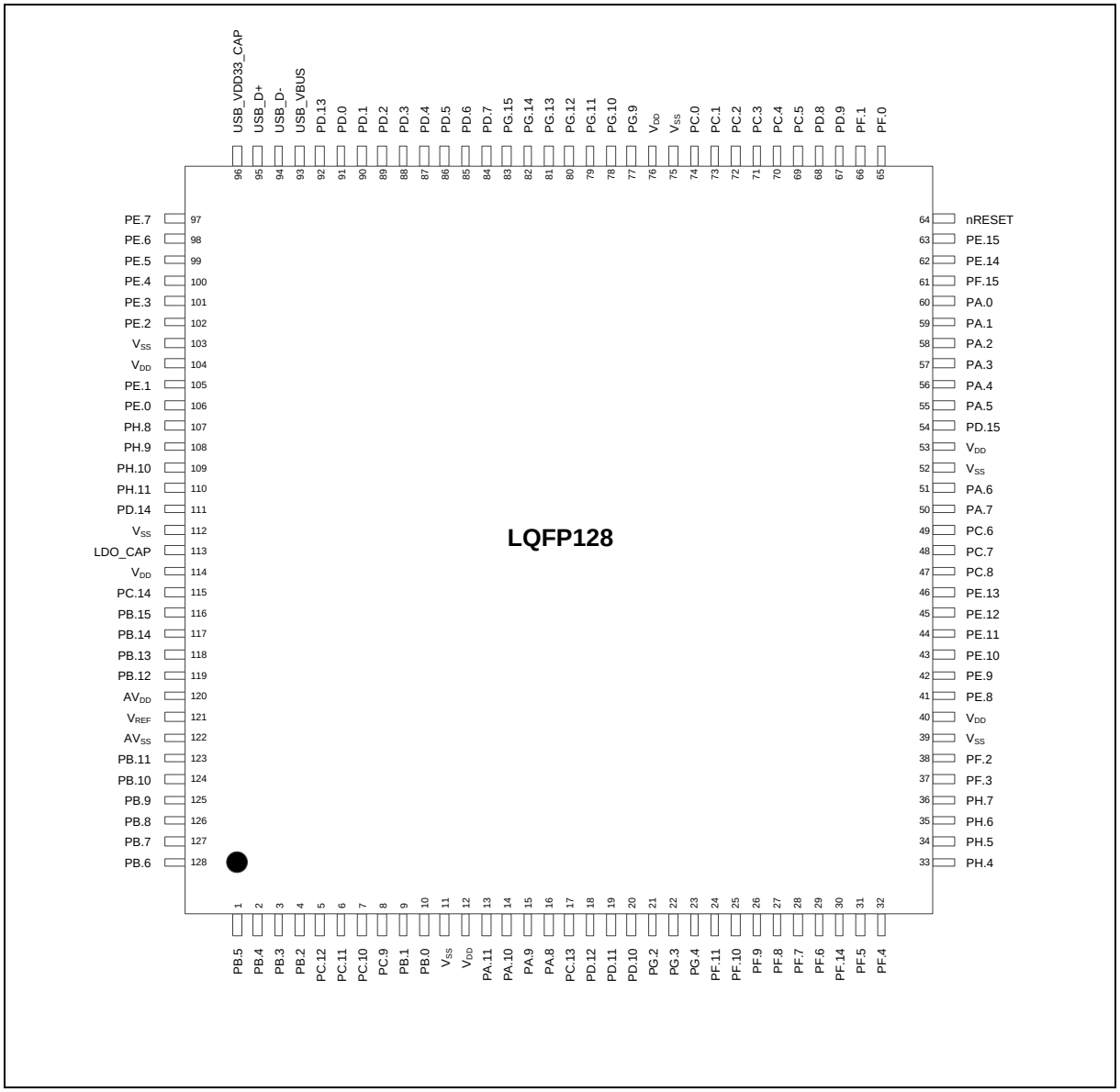


Figure 4.1-34 M032 Series LQFP 128-pin Diagram

4.1.4 M032 Series Multi-function Pin Diagram

4.1.4.1 M032 Series TSSOP 20-Pin Multi-function Pin Diagram

Corresponding Part Number: M032FC1AE

M032FC1AE

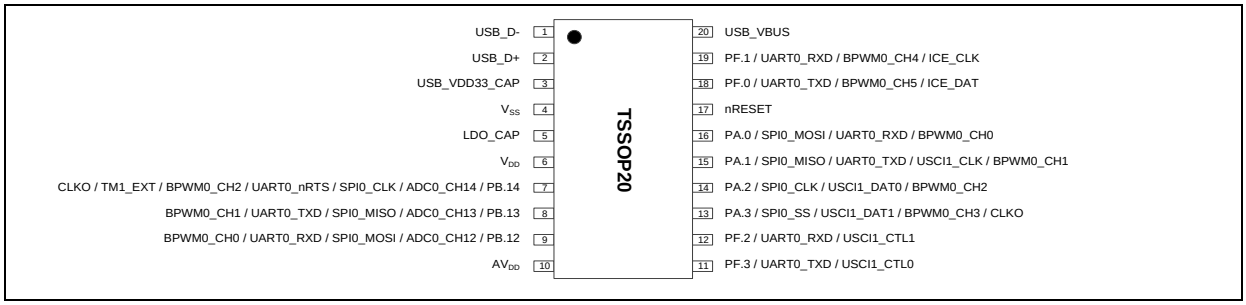


Figure 4.1-35 M032FC1AE Multi-function Pin Diagram

Pin	M032FC1AE Pin Function
1	USB_D-
2	USB_D+
3	USB_V_{DD}33_CAP
4	VSS
5	LDO_CAP
6	V_{DD}
7	PB.14 / ADC0_CH14 / SPI0_CLK / UART0_nRTS / BPWM0_CH2 / TM1_EXT / CLKO
8	PB.13 / ADC0_CH13 / SPI0_MISO / UART0_TXD / BPWM0_CH1
9	PB.12 / ADC0_CH12 / SPI0_MOSI / UART0_RXD / BPWM0_CH0
10	AV_{DD}
11	PF.3 / UART0_TXD / USCI1_CTL0
12	PF.2 / UART0_RXD / USCI1_CTL1
13	PA.3 / SPI0_SS / USCI1_DAT1 / BPWM0_CH3 / CLKO
14	PA.2 / SPI0_CLK / USCI1_DAT0 / BPWM0_CH2
15	PA.1 / SPI0_MISO / UART0_TXD / USCI1_CLK / BPWM0_CH1
16	PA.0 / SPI0_MOSI / UART0_RXD / BPWM0_CH0
17	nRESET
18	PF.0 / UART0_TXD / BPWM0_CH5 / ICE_DAT
19	PF.1 / UART0_RXD / BPWM0_CH4 / ICE_CLK
20	USB_VBUS

Table 4.1-23 M032FC1AE Multi-function Pin Table

4.1.4.2 M032 Series TSSOP 28-Pin Multi-function Pin Diagram

Corresponding Part Number: M032EC1AE

M032EC1AE

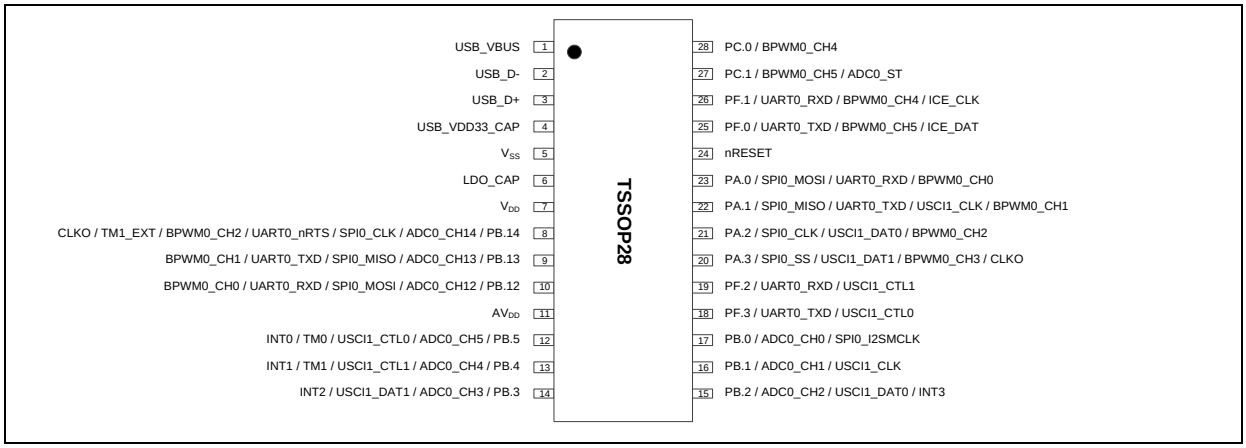


Figure 4.1-36 M032EC1AE Multi-function Pin Diagram

Pin	M032EC1AE Pin Function
1	USB_VBUS
2	USB_D-
3	USB_D+
4	USB_V_{DD}33_CAP
5	VSS
6	LDO_CAP
7	V_{DD}
8	PB.14 / ADC0_CH14 / SPI0_CLK / UART0_nRTS / BPWM0_CH2 / TM1_EXT / CLKO
9	PB.13 / ADC0_CH13 / SPI0_MISO / UART0_TXD / BPWM0_CH1
10	PB.12 / ADC0_CH12 / SPI0_MOSI / UART0_RXD / BPWM0_CH0
11	AV_{DD}
12	PB.5 / ADC0_CH5 / USCI1_CTL0 / TM0 / INT0
13	PB.4 / ADC0_CH4 / USCI1_CTL1 / TM1 / INT1
14	PB.3 / ADC0_CH3 / USCI1_DAT1 / INT2
15	PB.2 / ADC0_CH2 / USCI1_DAT0 / INT3
16	PB.1 / ADC0_CH1 / USCI1_CLK
17	PB.0 / ADC0_CH0 / SPI0_I2SMCLK
18	PF.3 / UART0_TXD / USCI1_CTL0
19	PF.2 / UART0_RXD / USCI1_CTL1
20	PA.3 / SPI0_SS / USCI1_DAT1 / BPWM0_CH3 / CLKO

Pin	M032EC1AE Pin Function
21	PA.2 / SPI0_CLK / USCI1_DAT0 / BPWM0_CH2
22	PA.1 / SPI0_MISO / UART0_TXD / USCI1_CLK / BPWM0_CH1
23	PA.0 / SPI0_MOSI / UART0_RXD / BPWM0_CH0
24	nRESET
25	PF.0 / UART0_TXD / BPWM0_CH5 / ICE_DAT
26	PF.1 / UART0_RXD / BPWM0_CH4 / ICE_CLK
27	PC.1 / BPWM0_CH5 / ADC0_ST
28	PC.0 / BPWM0_CH4

Table 4.1-24 M032EC1AE Multi-function Pin Table

4.1.4.3 M032 Series QFN 33-Pin Multi-function Pin Diagram

Corresponding Part Number: M032TC1AE, M032TD2AE

M032TC1AE

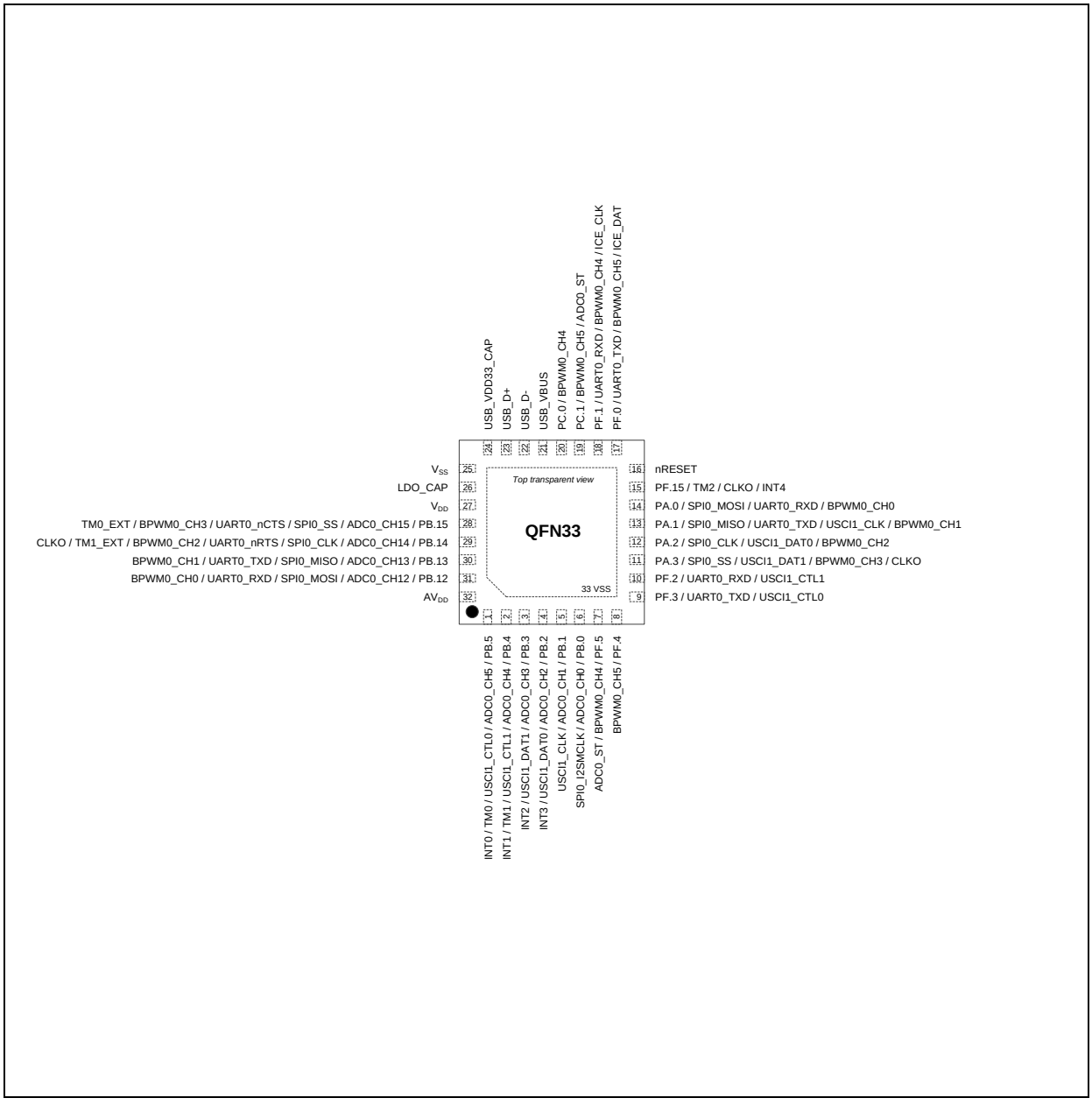


Figure 4.1-37 M032TC1AE Multi-function Pin Diagram

Pin	M032TC1AE Pin Function
1	PB.5 / ADC0_CH5 / USCI1_CTL0 / TM0 / INT0
2	PB.4 / ADC0_CH4 / USCI1_CTL1 / TM1 / INT1
3	PB.3 / ADC0_CH3 / USCI1_DAT1 / INT2

Pin	M032TC1AE Pin Function
4	PB.2 / ADC0_CH2 / USCI1_DAT0 / INT3
5	PB.1 / ADC0_CH1 / USCI1_CLK
6	PB.0 / ADC0_CH0 / SPI0_I2SMCLK
7	PF.5 / BPWM0_CH4 / ADC0_ST
8	PF.4 / BPWM0_CH5
9	PF.3 / UART0_TXD / USCI1_CTL0
10	PF.2 / UART0_RXD / USCI1_CTL1
11	PA.3 / SPI0_SS / USCI1_DAT1 / BPWM0_CH3 / CLKO
12	PA.2 / SPI0_CLK / USCI1_DAT0 / BPWM0_CH2
13	PA.1 / SPI0_MISO / UART0_TXD / USCI1_CLK / BPWM0_CH1
14	PA.0 / SPI0_MOSI / UART0_RXD / BPWM0_CH0
15	PF.15 / CLKO / INT4
16	nRESET
17	PF.0 / UART0_TXD / BPWM0_CH5 / ICE_DAT
18	PF.1 / UART0_RXD / BPWM0_CH4 / ICE_CLK
19	PC.1 / BPWM0_CH5 / ADC0_ST
20	PC.0 / BPWM0_CH4
21	USB_VBUS
22	USB_D-
23	USB_D+
24	USB_V_{DD}33_CAP
25	VSS
26	LDO_CAP
27	V_{DD}
28	PB.15 / ADC0_CH15 / SPI0_SS / UART0_nCTS / BPWM0_CH3 / TM0_EXT
29	PB.14 / ADC0_CH14 / SPI0_CLK / UART0_nRTS / BPWM0_CH2 / TM1_EXT / CLKO
30	PB.13 / ADC0_CH13 / SPI0_MISO / UART0_TXD / BPWM0_CH1
31	PB.12 / ADC0_CH12 / SPI0_MOSI / UART0_RXD / BPWM0_CH0
32	AV_{DD}

Table 4.1-25 M032TC1AE Multi-function Pin Table

M032TD2AE

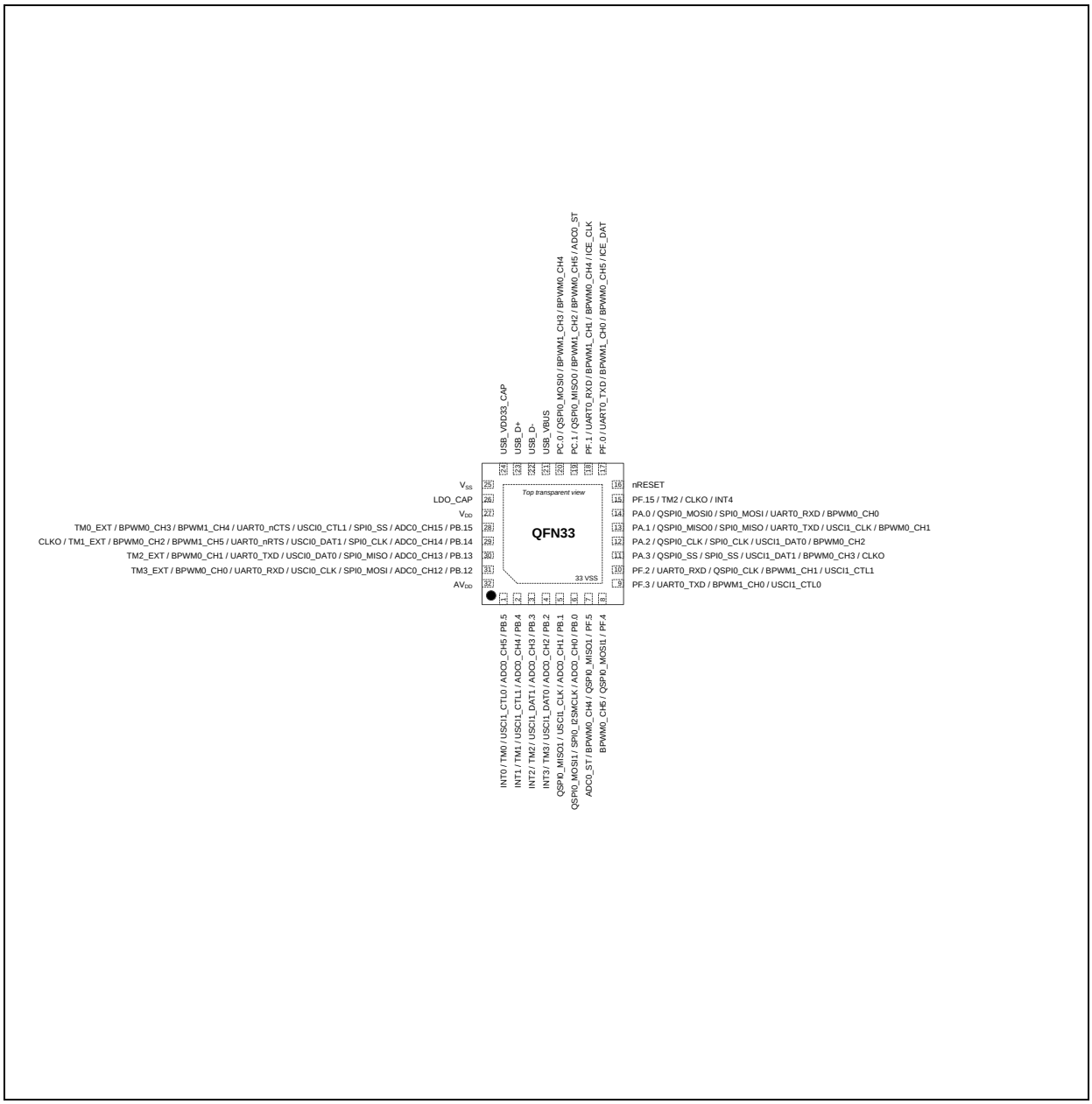


Figure 4.1-38 M032TD2AE Multi-function Pin Diagram

Pin	M032TD2AE Pin Function
1	PB.5 / ADC0_CH5 / USCI1_CTL0 / TM0 / INT0
2	PB.4 / ADC0_CH4 / USCI1_CTL1 / TM1 / INT1
3	PB.3 / ADC0_CH3 / USCI1_DAT1 / TM2 / INT2
4	PB.2 / ADC0_CH2 / USCI1_DAT0 / TM3 / INT3
5	PB.1 / ADC0_CH1 / USCI1_CLK / QSPI0_MISO1

Pin	M032TD2AE Pin Function
6	PB.0 / ADC0_CH0 / SPI0_I2SMCLK / QSPI0_MOSI1
7	PF.5 / QSPI0_MISO1 / BPWM0_CH4 / ADC0_ST
8	PF.4 / QSPI0_MOSI1 / BPWM0_CH5
9	PF.3 / UART0_TXD / BPWM1_CH0 / USCI1_CTL0
10	PF.2 / UART0_RXD / QSPI0_CLK / BPWM1_CH1 / USCI1_CTL1
11	PA.3 / QSPI0_SS / SPI0_SS / USCI1_DAT1 / BPWM0_CH3 / CLKO
12	PA.2 / QSPI0_CLK / SPI0_CLK / USCI1_DAT0 / BPWM0_CH2
13	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / USCI1_CLK / BPWM0_CH1
14	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / BPWM0_CH0
15	PF.15 / TM2 / CLKO / INT4
16	nRESET
17	PF.0 / UART0_TXD / BPWM1_CH0 / BPWM0_CH5 / ICE_DAT
18	PF.1 / UART0_RXD / BPWM1_CH1 / BPWM0_CH4 / ICE_CLK
19	PC.1 / QSPI0_MISO0 / BPWM1_CH2 / BPWM0_CH5 / ADC0_ST
20	PC.0 / QSPI0_MOSI0 / BPWM1_CH3 / BPWM0_CH4
21	USB_VBUS
22	USB_D-
23	USB_D+
24	USB_V_{DD}33_CAP
25	VSS
26	LDO_CAP
27	V_{DD}
28	PB.15 / ADC0_CH15 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / BPWM1_CH4 / BPWM0_CH3 / TM0_EXT
29	PB.14 / ADC0_CH14 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / BPWM1_CH5 / BPWM0_CH2 / TM1_EXT / CLKO
30	PB.13 / ADC0_CH13 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / BPWM0_CH1 / TM2_EXT
31	PB.12 / ADC0_CH12 / SPI0_MOSI / USCI0_CLK / UART0_RXD / BPWM0_CH0 / TM3_EXT
32	AV_{DD}

Table 4.1-26 M032TD2AE Multi-function Pin Table

4.1.4.4 M032 Series LQFP 48-Pin Multi-function Pin Diagram

Corresponding Part Number: M032LC2AE, M032LD2AE, M032LE3AE, M032LG6AE, M032LG8AE

M032LC2AE

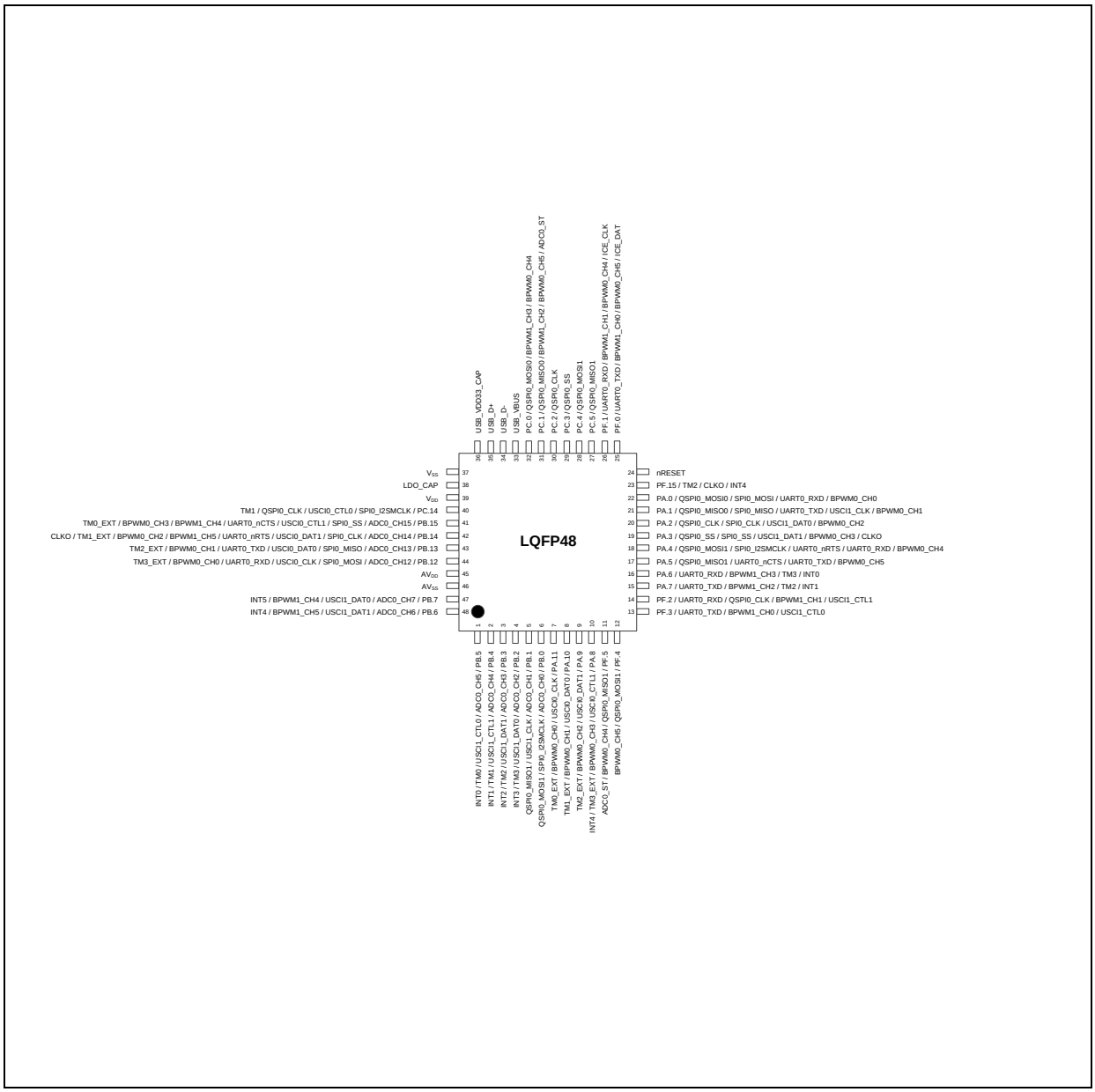


Figure 4.1-39 M032LC2AE Multi-function Pin Diagram

Pin	M032LC2AE Pin Function
1	PB.5 / ADC0_CH5 / USCI1_CTL0 / TM0 / INT0
2	PB.4 / ADC0_CH4 / USCI1_CTL1 / TM1 / INT1
3	PB.3 / ADC0_CH3 / USCI1_DAT1 / TM2 / INT2

Pin	M032LC2AE Pin Function
4	PB.2 / ADC0_CH2 / USCI1_DAT0 / TM3 / INT3
5	PB.1 / ADC0_CH1 / USCI1_CLK / QSPI0_MISO1
6	PB.0 / ADC0_CH0 / SPI0_I2SMCLK / QSPI0_MOSI1
7	PA.11 / USCI0_CLK / BPWM0_CH0 / TM0_EXT
8	PA.10 / USCI0_DAT0 / BPWM0_CH1 / TM1_EXT
9	PA.9 / USCI0_DAT1 / BPWM0_CH2 / TM2_EXT
10	PA.8 / USCI0_CTL1 / BPWM0_CH3 / TM3_EXT / INT4
11	PF.5 / QSPI0_MISO1 / BPWM0_CH4 / ADC0_ST
12	PF.4 / QSPI0_MOSI1 / BPWM0_CH5
13	PF.3 / UART0_TXD / BPWM1_CH0 / USCI1_CTL0
14	PF.2 / UART0_RXD / QSPI0_CLK / BPWM1_CH1 / USCI1_CTL1
15	PA.7 / UART0_TXD / BPWM1_CH2 / TM2 / INT1
16	PA.6 / UART0_RXD / BPWM1_CH3 / TM3 / INT0
17	PA.5 / QSPI0_MISO1 / UART0_nCTS / UART0_TXD / BPWM0_CH5
18	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / BPWM0_CH4
19	PA.3 / QSPI0_SS / SPI0_SS / USCI1_DAT1 / BPWM0_CH3 / CLKO
20	PA.2 / QSPI0_CLK / SPI0_CLK / USCI1_DAT0 / BPWM0_CH2
21	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / USCI1_CLK / BPWM0_CH1
22	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / BPWM0_CH0
23	PF.15 / TM2 / CLKO / INT4
24	nRESET
25	PF.0 / UART0_TXD / BPWM1_CH0 / BPWM0_CH5 / ICE_DAT
26	PF.1 / UART0_RXD / BPWM1_CH1 / BPWM0_CH4 / ICE_CLK
27	PC.5 / QSPI0_MISO1
28	PC.4 / QSPI0_MOSI1
29	PC.3 / QSPI0_SS
30	PC.2 / QSPI0_CLK
31	PC.1 / QSPI0_MISO0 / BPWM1_CH2 / BPWM0_CH5 / ADC0_ST
32	PC.0 / QSPI0_MOSI0 / BPWM1_CH3 / BPWM0_CH4
33	USB_VBUS
34	USB_D-
35	USB_D+
36	USB_V_{DD}33_CAP
37	VSS

Pin	M032LC2AE Pin Function
38	LDO_CAP
39	V_{DD}
40	PC.14 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / TM1
41	PB.15 / ADC0_CH15 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / BPWM1_CH4 / BPWM0_CH3 / TM0_EXT
42	PB.14 / ADC0_CH14 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / BPWM1_CH5 / BPWM0_CH2 / TM1_EXT / CLKO
43	PB.13 / ADC0_CH13 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / BPWM0_CH1 / TM2_EXT
44	PB.12 / ADC0_CH12 / SPI0_MOSI / USCI0_CLK / UART0_RXD / BPWM0_CH0 / TM3_EXT
45	AV_{DD}
46	AVSS
47	PB.7 / ADC0_CH7 / USCI1_DAT0 / BPWM1_CH4 / INT5
48	PB.6 / ADC0_CH6 / USCI1_DAT1 / BPWM1_CH5 / INT4

Table 4.1-27 M032LC2AE Multi-function Pin Table

M032LD2AE

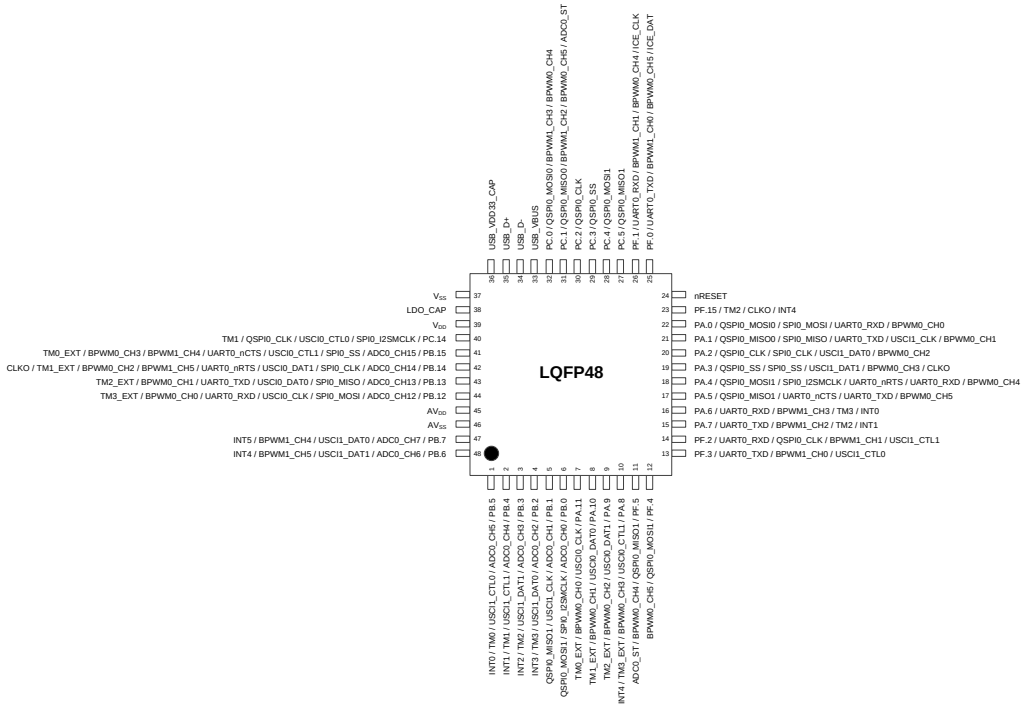


Figure 4.1-40 M032LD2AE Multi-function Pin Diagram

Pin	M032LD2AE Pin Function
1	PB.5 / ADC0_CH5 / USCI1_CTL0 / TM0 / INT0
2	PB.4 / ADC0_CH4 / USCI1_CTL1 / TM1 / INT1
3	PB.3 / ADC0_CH3 / USCI1_DAT1 / TM2 / INT2
4	PB.2 / ADC0_CH2 / USCI1_DAT0 / TM3 / INT3
5	PB.1 / ADC0_CH1 / USCI1_CLK / QSPI0_MISO1
6	PB.0 / ADC0_CH0 / SPI0_I2SMCLK / QSPI0_MOSI1

Pin	M032LD2AE Pin Function
7	PA.11 / USCI0_CLK / BPWM0_CH0 / TM0_EXT
8	PA.10 / USCI0_DAT0 / BPWM0_CH1 / TM1_EXT
9	PA.9 / USCI0_DAT1 / BPWM0_CH2 / TM2_EXT
10	PA.8 / USCI0_CTL1 / BPWM0_CH3 / TM3_EXT / INT4
11	PF.5 / QSPI0_MISO1 / BPWM0_CH4 / ADC0_ST
12	PF.4 / QSPI0_MOSI1 / BPWM0_CH5
13	PF.3 / UART0_TXD / BPWM1_CH0 / USCI1_CTL0
14	PF.2 / UART0_RXD / QSPI0_CLK / BPWM1_CH1 / USCI1_CTL1
15	PA.7 / UART0_TXD / BPWM1_CH2 / TM2 / INT1
16	PA.6 / UART0_RXD / BPWM1_CH3 / TM3 / INT0
17	PA.5 / QSPI0_MISO1 / UART0_nCTS / UART0_TXD / BPWM0_CH5
18	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / BPWM0_CH4
19	PA.3 / QSPI0_SS / SPI0_SS / USCI1_DAT1 / BPWM0_CH3 / CLKO
20	PA.2 / QSPI0_CLK / SPI0_CLK / USCI1_DAT0 / BPWM0_CH2
21	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / USCI1_CLK / BPWM0_CH1
22	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / BPWM0_CH0
23	PF.15 / TM2 / CLKO / INT4
24	nRESET
25	PF.0 / UART0_TXD / BPWM1_CH0 / BPWM0_CH5 / ICE_DAT
26	PF.1 / UART0_RXD / BPWM1_CH1 / BPWM0_CH4 / ICE_CLK
27	PC.5 / QSPI0_MISO1
28	PC.4 / QSPI0_MOSI1
29	PC.3 / QSPI0_SS
30	PC.2 / QSPI0_CLK
31	PC.1 / QSPI0_MISO0 / BPWM1_CH2 / BPWM0_CH5 / ADC0_ST
32	PC.0 / QSPI0_MOSI0 / BPWM1_CH3 / BPWM0_CH4
33	USB_VBUS
34	USB_D-
35	USB_D+
36	USB_V_{DD}33_CAP
37	VSS
38	LDO_CAP
39	V_{DD}
40	PC.14 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / TM1

Pin	M032LD2AE Pin Function
41	PB.15 / ADC0_CH15 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / BPWM1_CH4 / BPWM0_CH3 / TM0_EXT
42	PB.14 / ADC0_CH14 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / BPWM1_CH5 / BPWM0_CH2 / TM1_EXT / CLKO
43	PB.13 / ADC0_CH13 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / BPWM0_CH1 / TM2_EXT
44	PB.12 / ADC0_CH12 / SPI0_MOSI / USCI0_CLK / UART0_RXD / BPWM0_CH0 / TM3_EXT
45	AV_{DD}
46	AVSS
47	PB.7 / ADC0_CH7 / USCI1_DAT0 / BPWM1_CH4 / INT5
48	PB.6 / ADC0_CH6 / USCI1_DAT1 / BPWM1_CH5 / INT4

Table 4.1-28 M032LD2AE Multi-function Pin Table

M032LE3AE

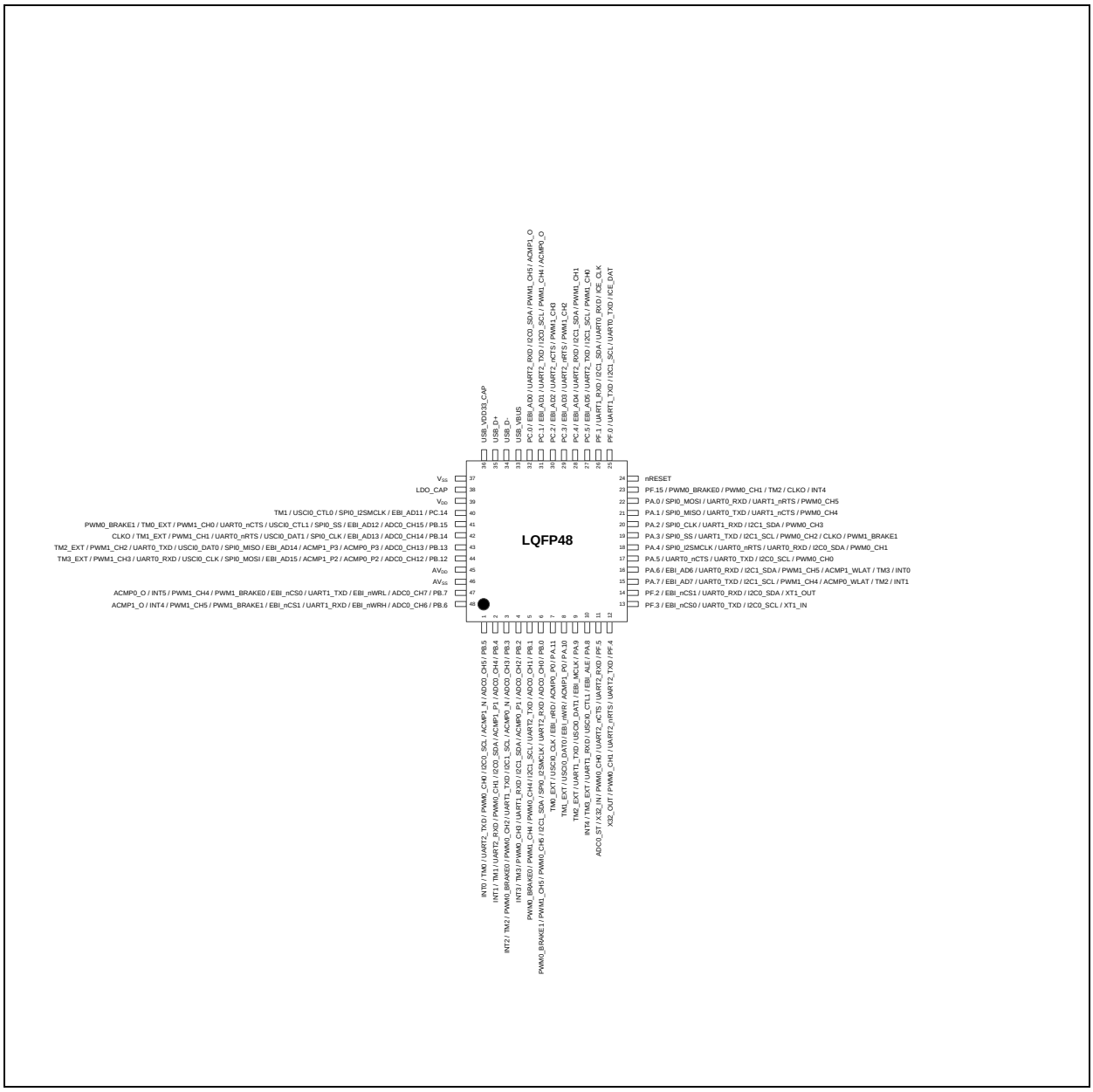


Figure 4.1-41 M032LE3AE Multi-function Pin Diagram

Pin	M032LE3AE Pin Function
1	PB.5 / ADC0_CH5 / ACMP1_N / I2C0_SCL / PWM0_CH0 / UART2_TXD / TM0 / INT0
2	PB.4 / ADC0_CH4 / ACMP1_P1 / I2C0_SDA / PWM0_CH1 / UART2_RXD / TM1 / INT1
3	PB.3 / ADC0_CH3 / ACMP0_N / I2C1_SCL / UART1_TXD / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2
4	PB.2 / ADC0_CH2 / ACMP0_P1 / I2C1_SDA / UART1_RXD / PWM0_CH3 / TM3 / INT3
5	PB.1 / ADC0_CH1 / UART2_TXD / I2C1_SCL / PWM0_CH4 / PWM1_CH4 / PWM0_BRAKE0

Pin	M032LE3AE Pin Function
6	PB.0 / ADC0_CH0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / PWM0_CH5 / PWM1_CH5 / PWM0_BRAKE1
7	PA.11 / ACMP0_P0 / EBI_nRD / USCI0_CLK / TM0_EXT
8	PA.10 / ACMP1_P0 / EBI_nWR / USCI0_DAT0 / TM1_EXT
9	PA.9 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / TM2_EXT
10	PA.8 / EBI_ALE / USCI0_CTL1 / UART1_RXD / TM3_EXT / INT4
11	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / X32_IN / ADC0_ST
12	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1 / X32_OUT
13	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN
14	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / XT1_OUT
15	PA.7 / EBI_AD7 / UART0_TXD / I2C1_SCL / PWM1_CH4 / ACMP0_WLAT / TM2 / INT1
16	PA.6 / EBI_AD6 / UART0_RXD / I2C1_SDA / PWM1_CH5 / ACMP1_WLAT / TM3 / INT0
17	PA.5 / UART0_nCTS / UART0_TXD / I2C0_SCL / PWM0_CH0
18	PA.4 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / PWM0_CH1
19	PA.3 / SPI0_SS / UART1_TXD / I2C1_SCL / PWM0_CH2 / CLKO / PWM1_BRAKE1
20	PA.2 / SPI0_CLK / UART1_RXD / I2C1_SDA / PWM0_CH3
21	PA.1 / SPI0_MISO / UART0_TXD / UART1_nCTS / PWM0_CH4
22	PA.0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / PWM0_CH5
23	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / TM2 / CLKO / INT4
24	nRESET
25	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / ICE_DAT
26	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / ICE_CLK
27	PC.5 / EBI_AD5 / UART2_TXD / I2C1_SCL / PWM1_CH0
28	PC.4 / EBI_AD4 / UART2_RXD / I2C1_SDA / PWM1_CH1
29	PC.3 / EBI_AD3 / UART2_nRTS / PWM1_CH2
30	PC.2 / EBI_AD2 / UART2_nCTS / PWM1_CH3
31	PC.1 / EBI_AD1 / UART2_TXD / I2C0_SCL / PWM1_CH4 / ACMP0_O
32	PC.0 / EBI_AD0 / UART2_RXD / I2C0_SDA / PWM1_CH5 / ACMP1_O
33	USB_VBUS
34	USB_D-
35	USB_D+
36	USB_V_{DD}33_CAP
37	VSS
38	LDO_CAP
39	V_{DD}

Pin	M032LE3AE Pin Function
40	PC.14 / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / TM1
41	PB.15 / ADC0_CH15 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / PWM1_CH0 / TM0_EXT / PWM0_BRAKE1
42	PB.14 / ADC0_CH14 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / PWM1_CH1 / TM1_EXT / CLKO
43	PB.13 / ADC0_CH13 / ACMP0_P3 / ACMP1_P3 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / PWM1_CH2 / TM2_EXT
44	PB.12 / ADC0_CH12 / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / PWM1_CH3 / TM3_EXT
45	AV_{DD}
46	AVSS
47	PB.7 / ADC0_CH7 / EBI_nWRL / UART1_TXD / EBI_nCS0 / PWM1_BRAKE0 / PWM1_CH4 / INT5 / ACMP0_O
48	PB.6 / ADC0_CH6 / EBI_nWRH / UART1_RXD / EBI_nCS1 / PWM1_BRAKE1 / PWM1_CH5 / INT4 / ACMP1_O

Table 4.1-29 M032LE3AE Multi-function Pin Table

M032LG6AE

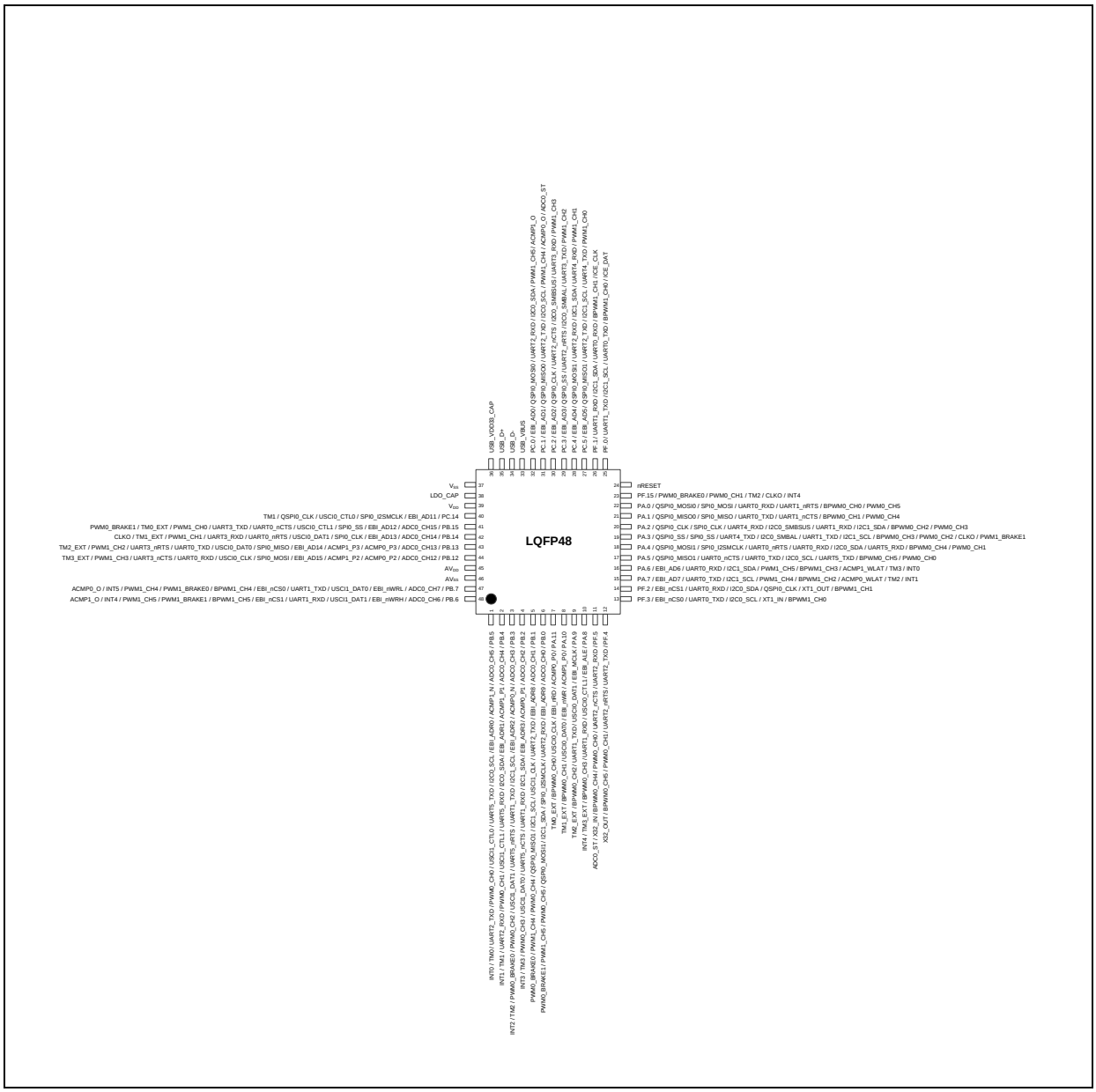


Figure 4.1-42 M032LG6AE Multi-function Pin Diagram

Pin	M032LG6AE Pin Function
1	PB.5 / ADC0_CH5 / ACMP1_N / EBI_ADR0 / I2C0_SCL / UART5_TXD / USCI1_CTL0 / PWM0_CH0 / UART2_TXD / TM0 / INT0
2	PB.4 / ADC0_CH4 / ACMP1_P1 / EBI_ADR1 / I2C0_SDA / UART5_RXD / USCI1_CTL1 / PWM0_CH1 / UART2_RXD / TM1 / INT1
3	PB.3 / ADC0_CH3 / ACMP0_N / EBI_ADR2 / I2C1_SCL / UART1_TXD / UART5_nRTS / USCI1_DAT1 / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2
4	PB.2 / ADC0_CH2 / ACMP0_P1 / EBI_ADR3 / I2C1_SDA / UART1_RXD / UART5_nCTS / USCI1_DAT0 /

Pin	M032LG6AE Pin Function
	PWM0_CH3 / TM3 / INT3
5	PB.1 / ADC0_CH1 / EBI_ADR8 / UART2_TXD / USCI1_CLK / I2C1_SCL / QSPI0_MISO1 / PWM0_CH4 / PWM1_CH4 / PWM0_BRAKE0
6	PB.0 / ADC0_CH0 / EBI_ADR9 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / PWM0_CH5 / PWM1_CH5 / PWM0_BRAKE1
7	PA.11 / ACMP0_P0 / EBI_nRD / USCI0_CLK / BPWM0_CH0 / TM0_EXT
8	PA.10 / ACMP1_P0 / EBI_nWR / USCI0_DAT0 / BPWM0_CH1 / TM1_EXT
9	PA.9 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / BPWM0_CH2 / TM2_EXT
10	PA.8 / EBI_ALE / USCI0_CTL1 / UART1_RXD / BPWM0_CH3 / TM3_EXT / INT4
11	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / BPWM0_CH4 / X32_IN / ADC0_ST
12	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1 / BPWM0_CH5 / X32_OUT
13	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / BPWM1_CH0
14	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / BPWM1_CH1
15	PA.7 / EBI_AD7 / UART0_TXD / I2C1_SCL / PWM1_CH4 / BPWM1_CH2 / ACMP0_WLAT / TM2 / INT1
16	PA.6 / EBI_AD6 / UART0_RXD / I2C1_SDA / PWM1_CH5 / BPWM1_CH3 / ACMP1_WLAT / TM3 / INT0
17	PA.5 / QSPI0_MISO1 / UART0_nCTS / UART0_TXD / I2C0_SCL / UART5_TXD / BPWM0_CH5 / PWM0_CH0
18	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / UART5_RXD / BPWM0_CH4 / PWM0_CH1
19	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / I2C0_SMBAL / UART1_TXD / I2C1_SCL / BPWM0_CH3 / PWM0_CH2 / CLKO / PWM1_BRAKE1
20	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / BPWM0_CH2 / PWM0_CH3
21	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / BPWM0_CH1 / PWM0_CH4
22	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / BPWM0_CH0 / PWM0_CH5
23	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / TM2 / CLKO / INT4
24	nRESET
25	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / BPWM1_CH0 / ICE_DAT
26	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / BPWM1_CH1 / ICE_CLK
27	PC.5 / EBI_AD5 / QSPI0_MISO1 / UART2_TXD / I2C1_SCL / UART4_TXD / PWM1_CH0
28	PC.4 / EBI_AD4 / QSPI0_MOSI1 / UART2_RXD / I2C1_SDA / UART4_RXD / PWM1_CH1
29	PC.3 / EBI_AD3 / QSPI0_SS / UART2_nRTS / I2C0_SMBAL / UART3_TXD / PWM1_CH2
30	PC.2 / EBI_AD2 / QSPI0_CLK / UART2_nCTS / I2C0_SMBSUS / UART3_RXD / PWM1_CH3
31	PC.1 / EBI_AD1 / QSPI0_MISO0 / UART2_TXD / I2C0_SCL / PWM1_CH4 / ACMP0_O / ADC0_ST
32	PC.0 / EBI_AD0 / QSPI0_MOSI0 / UART2_RXD / I2C0_SDA / PWM1_CH5 / ACMP1_O
33	USB_VBUS
34	USB_D-
35	USB_D+

Pin	M032LG6AE Pin Function
36	USB_V_{DD}33_CAP
37	VSS
38	LDO_CAP
39	V_{DD}
40	PC.14 / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / TM1
41	PB.15 / ADC0_CH15 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / PWM1_CH0 / TM0_EXT / PWM0_BRAKE1
42	PB.14 / ADC0_CH14 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / PWM1_CH1 / TM1_EXT / CLKO
43	PB.13 / ADC0_CH13 / ACMP0_P3 / ACMP1_P3 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / PWM1_CH2 / TM2_EXT
44	PB.12 / ADC0_CH12 / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / PWM1_CH3 / TM3_EXT
45	AV_{DD}
46	AVSS
47	PB.7 / ADC0_CH7 / EBI_nWRL / USCI1_DAT0 / UART1_TXD / EBI_nCS0 / BPWM1_CH4 / PWM1_BRAKE0 / PWM1_CH4 / INT5 / ACMP0_O
48	PB.6 / ADC0_CH6 / EBI_nWRH / USCI1_DAT1 / UART1_RXD / EBI_nCS1 / BPWM1_CH5 / PWM1_BRAKE1 / PWM1_CH5 / INT4 / ACMP1_O

Table 4.1-30 M032LG6AE Multi-function Pin Table

M032LG8AE

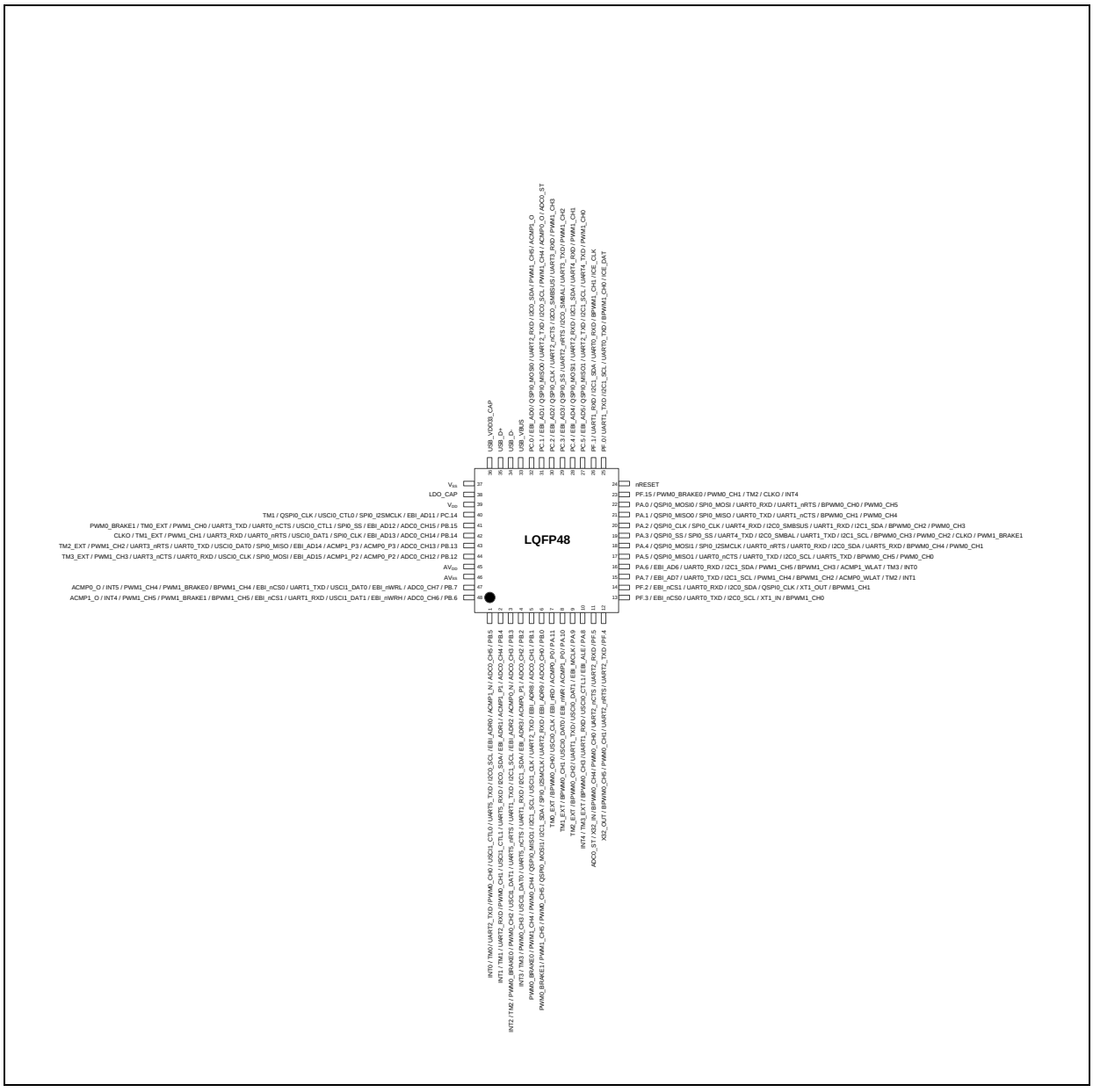


Figure 4.1-43 M032LG8AE Multi-function Pin Diagram

Pin	M032LG8AE Pin Function
1	PB.5 / ADC0_CH5 / ACMP1_N / EBI_ADR0 / I2C0_SCL / UART5_TXD / USCI1_CTL0 / PWM0_CH0 / UART2_TXD / TM0 / INT0
2	PB.4 / ADC0_CH4 / ACMP1_P1 / EBI_ADR1 / I2C0_SDA / UART5_RXD / USCI1_CTL1 / PWM0_CH1 / UART2_RXD / TM1 / INT1
3	PB.3 / ADC0_CH3 / ACMP0_N / EBI_ADR2 / I2C1_SCL / UART1_TXD / UART5_nRTS / USCI1_DAT1 / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2
4	PB.2 / ADC0_CH2 / ACMP0_P1 / EBI_ADR3 / I2C1_SDA / UART1_RXD / UART5_nCTS / USCI1_DAT0 /

Pin	M032LG8AE Pin Function
	PWM0_CH3 / TM3 / INT3
5	PB.1 / ADC0_CH1 / EBI_ADR8 / UART2_TXD / USCI1_CLK / I2C1_SCL / QSPI0_MISO1 / PWM0_CH4 / PWM1_CH4 / PWM0_BRAKE0
6	PB.0 / ADC0_CH0 / EBI_ADR9 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / PWM0_CH5 / PWM1_CH5 / PWM0_BRAKE1
7	PA.11 / ACMP0_P0 / EBI_nRD / USCI0_CLK / BPWM0_CH0 / TM0_EXT
8	PA.10 / ACMP1_P0 / EBI_nWR / USCI0_DAT0 / BPWM0_CH1 / TM1_EXT
9	PA.9 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / BPWM0_CH2 / TM2_EXT
10	PA.8 / EBI_ALE / USCI0_CTL1 / UART1_RXD / BPWM0_CH3 / TM3_EXT / INT4
11	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / BPWM0_CH4 / X32_IN / ADC0_ST
12	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1 / BPWM0_CH5 / X32_OUT
13	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / BPWM1_CH0
14	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / BPWM1_CH1
15	PA.7 / EBI_AD7 / UART0_TXD / I2C1_SCL / PWM1_CH4 / BPWM1_CH2 / ACMP0_WLAT / TM2 / INT1
16	PA.6 / EBI_AD6 / UART0_RXD / I2C1_SDA / PWM1_CH5 / BPWM1_CH3 / ACMP1_WLAT / TM3 / INT0
17	PA.5 / QSPI0_MISO1 / UART0_nCTS / UART0_TXD / I2C0_SCL / UART5_TXD / BPWM0_CH5 / PWM0_CH0
18	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / UART5_RXD / BPWM0_CH4 / PWM0_CH1
19	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / I2C0_SMBAL / UART1_TXD / I2C1_SCL / BPWM0_CH3 / PWM0_CH2 / CLKO / PWM1_BRAKE1
20	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / BPWM0_CH2 / PWM0_CH3
21	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / BPWM0_CH1 / PWM0_CH4
22	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / BPWM0_CH0 / PWM0_CH5
23	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / TM2 / CLKO / INT4
24	nRESET
25	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / BPWM1_CH0 / ICE_DAT
26	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / BPWM1_CH1 / ICE_CLK
27	PC.5 / EBI_AD5 / QSPI0_MISO1 / UART2_TXD / I2C1_SCL / UART4_TXD / PWM1_CH0
28	PC.4 / EBI_AD4 / QSPI0_MOSI1 / UART2_RXD / I2C1_SDA / UART4_RXD / PWM1_CH1
29	PC.3 / EBI_AD3 / QSPI0_SS / UART2_nRTS / I2C0_SMBAL / UART3_TXD / PWM1_CH2
30	PC.2 / EBI_AD2 / QSPI0_CLK / UART2_nCTS / I2C0_SMBSUS / UART3_RXD / PWM1_CH3
31	PC.1 / EBI_AD1 / QSPI0_MISO0 / UART2_TXD / I2C0_SCL / PWM1_CH4 / ACMP0_O / ADC0_ST
32	PC.0 / EBI_AD0 / QSPI0_MOSI0 / UART2_RXD / I2C0_SDA / PWM1_CH5 / ACMP1_O
33	USB_VBUS
34	USB_D-
35	USB_D+

Pin	M032LG8AE Pin Function
36	USB_V_{DD}33_CAP
37	VSS
38	LDO_CAP
39	V_{DD}
40	PC.14 / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / TM1
41	PB.15 / ADC0_CH15 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / PWM1_CH0 / TM0_EXT / PWM0_BRAKE1
42	PB.14 / ADC0_CH14 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / PWM1_CH1 / TM1_EXT / CLKO
43	PB.13 / ADC0_CH13 / ACMP0_P3 / ACMP1_P3 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / PWM1_CH2 / TM2_EXT
44	PB.12 / ADC0_CH12 / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / PWM1_CH3 / TM3_EXT
45	AV_{DD}
46	AVSS
47	PB.7 / ADC0_CH7 / EBI_nWRL / USCI1_DAT0 / UART1_TXD / EBI_nCS0 / BPWM1_CH4 / PWM1_BRAKE0 / PWM1_CH4 / INT5 / ACMP0_O
48	PB.6 / ADC0_CH6 / EBI_nWRH / USCI1_DAT1 / UART1_RXD / EBI_nCS1 / BPWM1_CH5 / PWM1_BRAKE1 / PWM1_CH5 / INT4 / ACMP1_O

Table 4.1-31 M032LG8AE Multi-function Pin Table

4.1.4.5 M032 Series LQFP 64-Pin Multi-function Pin Diagram

Corresponding Part Number: M032SE3AE, M032SG6AE, M032SG8AE, M032SIAAE

M032SE3AE

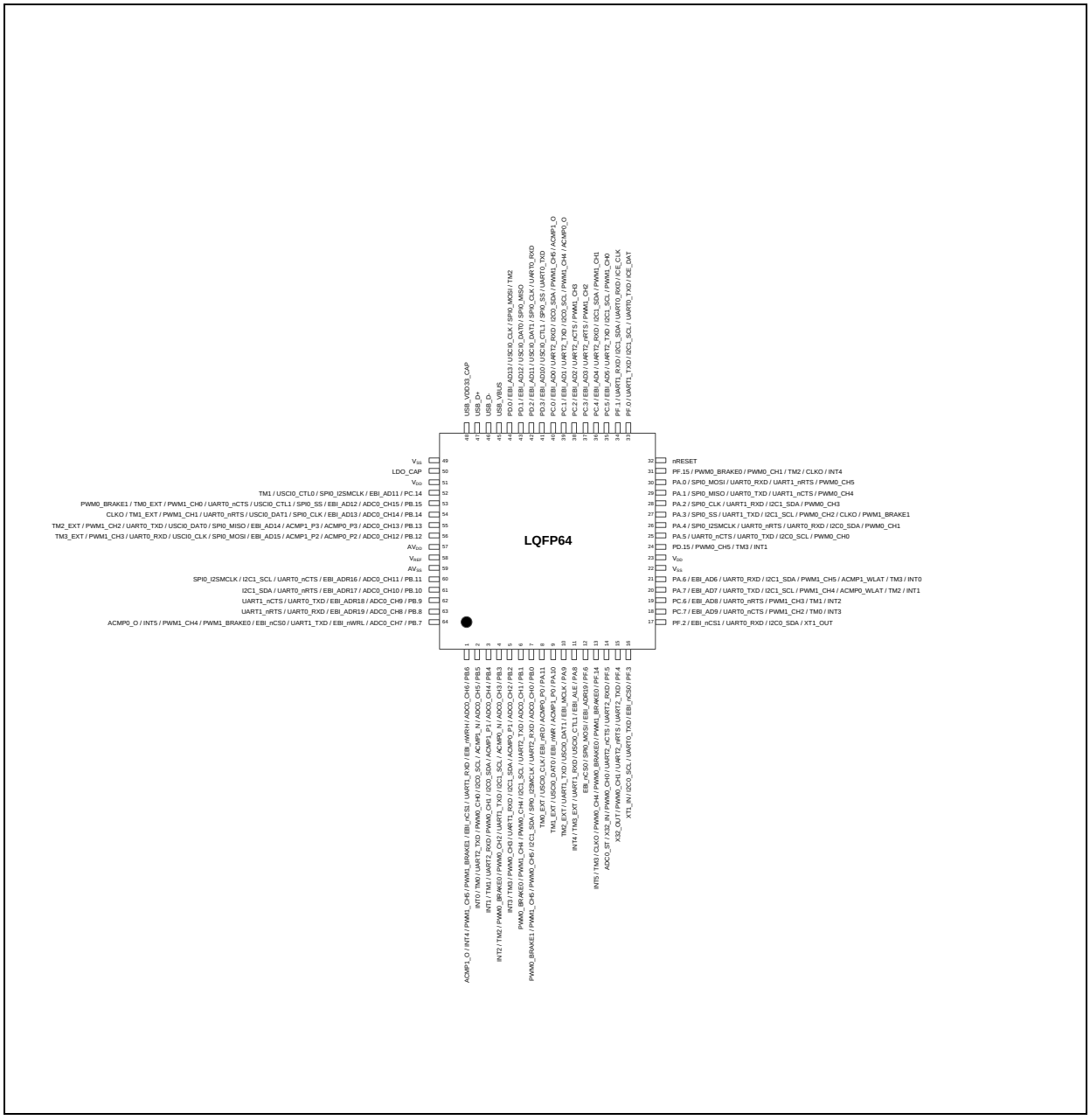


Figure 4.1-44 M032SE3AE Multi-function Pin Diagram

Pin	M032SE3AE Pin Function
1	PB.6 / ADC0_CH6 / EBI_nWRH / UART1_RXD / EBI_nCS1 / PWM1_BRAKE1 / PWM1_CH5 / INT4 / ACMP1_O
2	PB.5 / ADC0_CH5 / ACMP1_N / I2C0_SCL / PWM0_CH0 / UART2_TXD / TM0 / INT0

Pin	M032SE3AE Pin Function
3	PB.4 / ADC0_CH4 / ACMP1_P1 / I2C0_SDA / PWM0_CH1 / UART2_RXD / TM1 / INT1
4	PB.3 / ADC0_CH3 / ACMP0_N / I2C1_SCL / UART1_TXD / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2
5	PB.2 / ADC0_CH2 / ACMP0_P1 / I2C1_SDA / UART1_RXD / PWM0_CH3 / TM3 / INT3
6	PB.1 / ADC0_CH1 / UART2_TXD / I2C1_SCL / PWM0_CH4 / PWM1_CH4 / PWM0_BRAKE0
7	PB.0 / ADC0_CH0 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / PWM0_CH5 / PWM1_CH5 / PWM0_BRAKE1
8	PA.11 / ACMP0_P0 / EBI_nRD / USCI0_CLK / TM0_EXT
9	PA.10 / ACMP1_P0 / EBI_nWR / USCI0_DAT0 / TM1_EXT
10	PA.9 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / TM2_EXT
11	PA.8 / EBI_ALE / USCI0_CTL1 / UART1_RXD / TM3_EXT / INT4
12	PF.6 / EBI_ADR19 / SPI0_MOSI / EBI_nCS0
13	PF.14 / PWM1_BRAKE0 / PWM0_BRAKE0 / PWM0_CH4 / CLKO / TM3 / INT5
14	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / X32_IN / ADC0_ST
15	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1 / X32_OUT
16	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN
17	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / XT1_OUT
18	PC.7 / EBI_AD9 / UART0_nCTS / PWM1_CH2 / TM0 / INT3
19	PC.6 / EBI_AD8 / UART0_nRTS / PWM1_CH3 / TM1 / INT2
20	PA.7 / EBI_AD7 / UART0_TXD / I2C1_SCL / PWM1_CH4 / ACMP0_WLAT / TM2 / INT1
21	PA.6 / EBI_AD6 / UART0_RXD / I2C1_SDA / PWM1_CH5 / ACMP1_WLAT / TM3 / INT0
22	VSS
23	V_{DD}
24	PD.15 / PWM0_CH5 / TM3 / INT1
25	PA.5 / UART0_nCTS / UART0_TXD / I2C0_SCL / PWM0_CH0
26	PA.4 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / PWM0_CH1
27	PA.3 / SPI0_SS / UART1_TXD / I2C1_SCL / PWM0_CH2 / CLKO / PWM1_BRAKE1
28	PA.2 / SPI0_CLK / UART1_RXD / I2C1_SDA / PWM0_CH3
29	PA.1 / SPI0_MISO / UART0_TXD / UART1_nCTS / PWM0_CH4
30	PA.0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / PWM0_CH5
31	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / TM2 / CLKO / INT4
32	nRESET
33	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / ICE_DAT
34	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / ICE_CLK
35	PC.5 / EBI_AD5 / UART2_TXD / I2C1_SCL / PWM1_CH0
36	PC.4 / EBI_AD4 / UART2_RXD / I2C1_SDA / PWM1_CH1

Pin	M032SE3AE Pin Function
37	PC.3 / EBI_AD3 / UART2_nRTS / PWM1_CH2
38	PC.2 / EBI_AD2 / UART2_nCTS / PWM1_CH3
39	PC.1 / EBI_AD1 / UART2_TXD / I2C0_SCL / PWM1_CH4 / ACMP0_O
40	PC.0 / EBI_AD0 / UART2_RXD / I2C0_SDA / PWM1_CH5 / ACMP1_O
41	PD.3 / EBI_AD10 / USCI0_CTL1 / SPI0_SS / UART0_TXD
42	PD.2 / EBI_AD11 / USCI0_DAT1 / SPI0_CLK / UART0_RXD
43	PD.1 / EBI_AD12 / USCI0_DAT0 / SPI0_MISO
44	PD.0 / EBI_AD13 / USCI0_CLK / SPI0_MOSI / TM2
45	USB_VBUS
46	USB_D-
47	USB_D+
48	USB_V_{DD}33_CAP
49	VSS
50	LDO_CAP
51	V_{DD}
52	PC.14 / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / TM1
53	PB.15 / ADC0_CH15 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / PWM1_CH0 / TM0_EXT / PWM0_BRAKE1
54	PB.14 / ADC0_CH14 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / PWM1_CH1 / TM1_EXT / CLKO
55	PB.13 / ADC0_CH13 / ACMP0_P3 / ACMP1_P3 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / PWM1_CH2 / TM2_EXT
56	PB.12 / ADC0_CH12 / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / PWM1_CH3 / TM3_EXT
57	AV_{DD}
58	V_{REF}
59	AVSS
60	PB.11 / ADC0_CH11 / EBI_ADR16 / UART0_nCTS / I2C1_SCL / SPI0_I2SMCLK
61	PB.10 / ADC0_CH10 / EBI_ADR17 / UART0_nRTS / I2C1_SDA
62	PB.9 / ADC0_CH9 / EBI_ADR18 / UART0_TXD / UART1_nCTS
63	PB.8 / ADC0_CH8 / EBI_ADR19 / UART0_RXD / UART1_nRTS
64	PB.7 / ADC0_CH7 / EBI_nWRL / UART1_TXD / EBI_nCS0 / PWM1_BRAKE0 / PWM1_CH4 / INT5 / ACMP0_O

Table 4.1-32 M032SE2AE Multi-function Pin Table

M032SG6AE

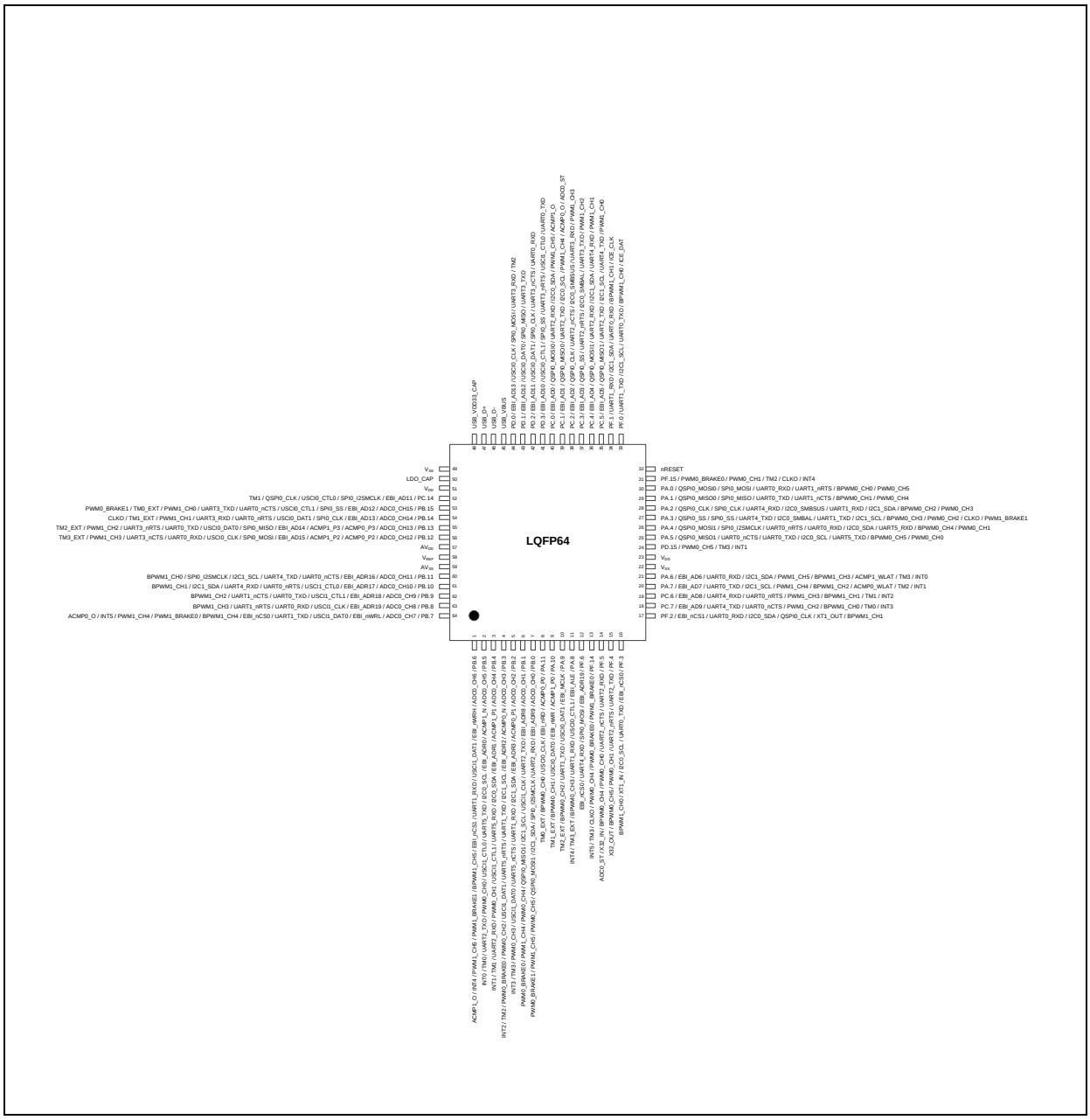


Figure 4.1-45 M032SG6AE Multi-function Pin Diagram

Pin	M032SG6AE Pin Function
1	PB.6 / ADC0_CH6 / EBI_nWRH / USCI1_DAT1 / UART1_RXD / EBI_nCS1 / BPWM1_CH5 / PWM1_BRAKE1 / PWM1_CH5 / INT4 / ACMP1_O
2	PB.5 / ADC0_CH5 / ACMP1_N / EBI_ADR0 / I2C0_SCL / UART5_TXD / USCI1_CTL0 / PWM0_CH0 / UART2_TXD / TM0 / INT0
3	PB.4 / ADC0_CH4 / ACMP1_P1 / EBI_ADR1 / I2C0_SDA / UART5_RXD / USCI1_CTL1 / PWM0_CH1 / UART2_RXD / TM1 / INT1

Pin	M032SG6AE Pin Function
4	PB.3 / ADC0_CH3 / ACMP0_N / EBI_ADR2 / I2C1_SCL / UART1_TXD / UART5_nRTS / USCI1_DAT1 / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2
5	PB.2 / ADC0_CH2 / ACMP0_P1 / EBI_ADR3 / I2C1_SDA / UART1_RXD / UART5_nCTS / USCI1_DAT0 / PWM0_CH3 / TM3 / INT3
6	PB.1 / ADC0_CH1 / EBI_ADR8 / UART2_TXD / USCI1_CLK / I2C1_SCL / QSPI0_MISO1 / PWM0_CH4 / PWM1_CH4 / PWM0_BRAKE0
7	PB.0 / ADC0_CH0 / EBI_ADR9 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / PWM0_CH5 / PWM1_CH5 / PWM0_BRAKE1
8	PA.11 / ACMP0_P0 / EBI_nRD / USCI0_CLK / BPWM0_CH0 / TM0_EXT
9	PA.10 / ACMP1_P0 / EBI_nWR / USCI0_DAT0 / BPWM0_CH1 / TM1_EXT
10	PA.9 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / BPWM0_CH2 / TM2_EXT
11	PA.8 / EBI_ALE / USCI0_CTL1 / UART1_RXD / BPWM0_CH3 / TM3_EXT / INT4
12	PF.6 / EBI_ADR19 / SPI0_MOSI / UART4_RXD / EBI_nCS0
13	PF.14 / PWM1_BRAKE0 / PWM0_BRAKE0 / PWM0_CH4 / CLKO / TM3 / INT5
14	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / BPWM0_CH4 / X32_IN / ADC0_ST
15	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1 / BPWM0_CH5 / X32_OUT
16	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / BPWM1_CH0
17	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / BPWM1_CH1
18	PC.7 / EBI_AD9 / UART4_TXD / UART0_nCTS / PWM1_CH2 / BPWM1_CH0 / TM0 / INT3
19	PC.6 / EBI_AD8 / UART4_RXD / UART0_nRTS / PWM1_CH3 / BPWM1_CH1 / TM1 / INT2
20	PA.7 / EBI_AD7 / UART0_TXD / I2C1_SCL / PWM1_CH4 / BPWM1_CH2 / ACMP0_WLAT / TM2 / INT1
21	PA.6 / EBI_AD6 / UART0_RXD / I2C1_SDA / PWM1_CH5 / BPWM1_CH3 / ACMP1_WLAT / TM3 / INT0
22	VSS
23	V_{DD}
24	PD.15 / PWM0_CH5 / TM3 / INT1
25	PA.5 / QSPI0_MISO1 / UART0_nCTS / UART0_TXD / I2C0_SCL / UART5_TXD / BPWM0_CH5 / PWM0_CH0
26	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / UART5_RXD / BPWM0_CH4 / PWM0_CH1
27	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / I2C0_SMBAL / UART1_TXD / I2C1_SCL / BPWM0_CH3 / PWM0_CH2 / CLKO / PWM1_BRAKE1
28	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / BPWM0_CH2 / PWM0_CH3
29	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / BPWM0_CH1 / PWM0_CH4
30	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / BPWM0_CH0 / PWM0_CH5
31	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / TM2 / CLKO / INT4
32	nRESET
33	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / BPWM1_CH0 / ICE_DAT
34	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / BPWM1_CH1 / ICE_CLK

Pin	M032SG6AE Pin Function
35	PC.5 / EBI_AD5 / QSPI0_MISO1 / UART2_TXD / I2C1_SCL / UART4_TXD / PWM1_CH0
36	PC.4 / EBI_AD4 / QSPI0_MOSI1 / UART2_RXD / I2C1_SDA / UART4_RXD / PWM1_CH1
37	PC.3 / EBI_AD3 / QSPI0_SS / UART2_nRTS / I2C0_SMBAL / UART3_TXD / PWM1_CH2
38	PC.2 / EBI_AD2 / QSPI0_CLK / UART2_nCTS / I2C0_SMBSUS / UART3_RXD / PWM1_CH3
39	PC.1 / EBI_AD1 / QSPI0_MISO0 / UART2_TXD / I2C0_SCL / PWM1_CH4 / ACMP0_O / ADC0_ST
40	PC.0 / EBI_AD0 / QSPI0_MOSI0 / UART2_RXD / I2C0_SDA / PWM1_CH5 / ACMP1_O
41	PD.3 / EBI_AD10 / USCI0_CTL1 / SPI0_SS / UART3_nRTS / USCI1_CTL0 / UART0_TXD
42	PD.2 / EBI_AD11 / USCI0_DAT1 / SPI0_CLK / UART3_nCTS / UART0_RXD
43	PD.1 / EBI_AD12 / USCI0_DAT0 / SPI0_MISO / UART3_TXD
44	PD.0 / EBI_AD13 / USCI0_CLK / SPI0_MOSI / UART3_RXD / TM2
45	USB_VBUS
46	USB_D-
47	USB_D+
48	USB_V_{DD}33_CAP
49	VSS
50	LDO_CAP
51	V_{DD}
52	PC.14 / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / TM1
53	PB.15 / ADC0_CH15 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / PWM1_CH0 / TM0_EXT / PWM0_BRAKE1
54	PB.14 / ADC0_CH14 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / PWM1_CH1 / TM1_EXT / CLKO
55	PB.13 / ADC0_CH13 / ACMP0_P3 / ACMP1_P3 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / PWM1_CH2 / TM2_EXT
56	PB.12 / ADC0_CH12 / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / PWM1_CH3 / TM3_EXT
57	AV_{DD}
58	V_{REF}
59	AVSS
60	PB.11 / ADC0_CH11 / EBI_ADR16 / UART0_nCTS / UART4_TXD / I2C1_SCL / SPI0_I2SMCLK / BPWM1_CH0
61	PB.10 / ADC0_CH10 / EBI_ADR17 / USCI1_CTL0 / UART0_nRTS / UART4_RXD / I2C1_SDA / BPWM1_CH1
62	PB.9 / ADC0_CH9 / EBI_ADR18 / USCI1_CTL1 / UART0_TXD / UART1_nCTS / BPWM1_CH2
63	PB.8 / ADC0_CH8 / EBI_ADR19 / USCI1_CLK / UART0_RXD / UART1_nRTS / BPWM1_CH3
64	PB.7 / ADC0_CH7 / EBI_nWRL / USCI1_DAT0 / UART1_TXD / EBI_nCS0 / BPWM1_CH4 / PWM1_BRAKE0 / PWM1_CH4 / INT5 / ACMP0_O

Table 4.1-33 M032SG6AE Multi-function Pin Table

M032SG8AE

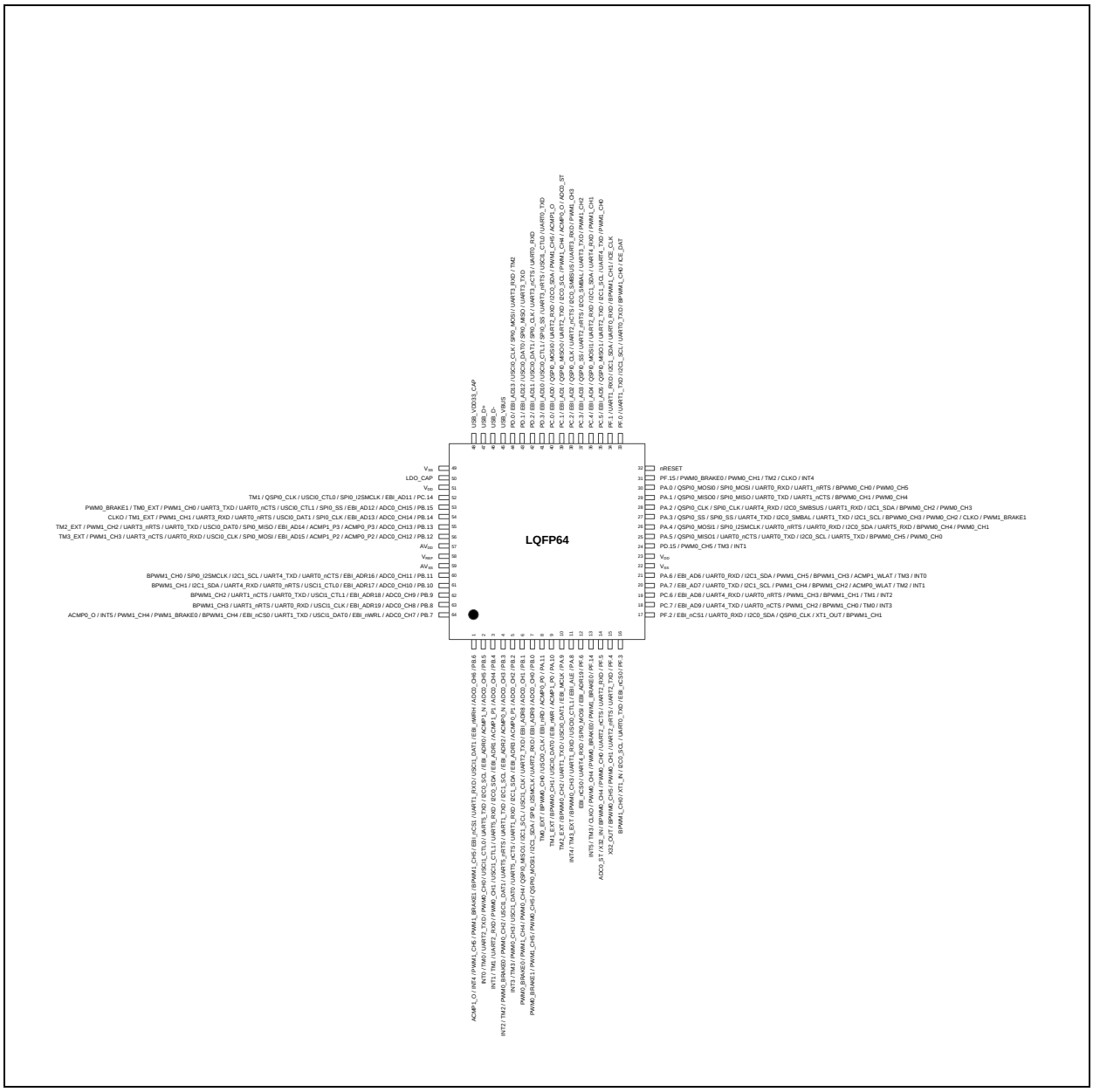


Figure 4.1-46 M032SG8AE Multi-function Pin Diagram

Pin	M032SG8AE Pin Function
1	PB.6 / ADC0_CH6 / EBI_nWRH / USCI1_DAT1 / UART1_RXD / EBI_nCS1 / BPWM1_CH5 / PWM1_BRAKE1 / PWM1_CH5 / INT4 / ACMP1_O
2	PB.5 / ADC0_CH5 / ACMP1_N / EBI_ADR0 / I2C0_SCL / UART5_TXD / USCI1_CTL0 / PWM0_CH0 / UART2_TXD / TM0 / INT0
3	PB.4 / ADC0_CH4 / ACMP1_P1 / EBI_ADR1 / I2C0_SDA / UART5_RXD / USCI1_CTL1 / PWM0_CH1 / UART2_RXD / TM1 / INT1
4	PB.3 / ADC0_CH3 / ACMP0_N / EBI_ADR2 / I2C1_SCL / UART1_TXD / UART5_nRTS / USCI1_DAT1 / PWM0_CH2 /

Pin	M032SG8AE Pin Function
	PWM0_BRAKE0 / TM2 / INT2
5	PB.2 / ADC0_CH2 / ACMP0_P1 / EBI_ADR3 / I2C1_SDA / UART1_RXD / UART5_nCTS / USCI1_DAT0 / PWM0_CH3 / TM3 / INT3
6	PB.1 / ADC0_CH1 / EBI_ADR8 / UART2_TXD / USCI1_CLK / I2C1_SCL / QSPI0_MISO1 / PWM0_CH4 / PWM1_CH4 / PWM0_BRAKE0
7	PB.0 / ADC0_CH0 / EBI_ADR9 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / PWM0_CH5 / PWM1_CH5 / PWM0_BRAKE1
8	PA.11 / ACMP0_P0 / EBI_nRD / USCI0_CLK / BPWM0_CH0 / TM0_EXT
9	PA.10 / ACMP1_P0 / EBI_nWR / USCI0_DAT0 / BPWM0_CH1 / TM1_EXT
10	PA.9 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / BPWM0_CH2 / TM2_EXT
11	PA.8 / EBI_ALE / USCI0_CTL1 / UART1_RXD / BPWM0_CH3 / TM3_EXT / INT4
12	PF.6 / EBI_ADR19 / SPI0_MOSI / UART4_RXD / EBI_nCS0
13	PF.14 / PWM1_BRAKE0 / PWM0_BRAKE0 / PWM0_CH4 / CLKO / TM3 / INT5
14	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / BPWM0_CH4 / X32_IN / ADC0_ST
15	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1 / BPWM0_CH5 / X32_OUT
16	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / BPWM1_CH0
17	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / BPWM1_CH1
18	PC.7 / EBI_AD9 / UART4_TXD / UART0_nCTS / PWM1_CH2 / BPWM1_CH0 / TM0 / INT3
19	PC.6 / EBI_AD8 / UART4_RXD / UART0_nRTS / PWM1_CH3 / BPWM1_CH1 / TM1 / INT2
20	PA.7 / EBI_AD7 / UART0_TXD / I2C1_SCL / PWM1_CH4 / BPWM1_CH2 / ACMP0_WLAT / TM2 / INT1
21	PA.6 / EBI_AD6 / UART0_RXD / I2C1_SDA / PWM1_CH5 / BPWM1_CH3 / ACMP1_WLAT / TM3 / INT0
22	VSS
23	V_{DD}
24	PD.15 / PWM0_CH5 / TM3 / INT1
25	PA.5 / QSPI0_MISO1 / UART0_nCTS / UART0_TXD / I2C0_SCL / UART5_TXD / BPWM0_CH5 / PWM0_CH0
26	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / UART5_RXD / BPWM0_CH4 / PWM0_CH1
27	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / I2C0_SMBAL / UART1_TXD / I2C1_SCL / BPWM0_CH3 / PWM0_CH2 / CLKO / PWM1_BRAKE1
28	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / BPWM0_CH2 / PWM0_CH3
29	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / BPWM0_CH1 / PWM0_CH4
30	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / BPWM0_CH0 / PWM0_CH5
31	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / TM2 / CLKO / INT4
32	nRESET
33	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / BPWM1_CH0 / ICE_DAT
34	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / BPWM1_CH1 / ICE_CLK
35	PC.5 / EBI_AD5 / QSPI0_MISO1 / UART2_TXD / I2C1_SCL / UART4_TXD / PWM1_CH0

Pin	M032SG8AE Pin Function
36	PC.4 / EBI_AD4 / QSPI0_MOSI1 / UART2_RXD / I2C1_SDA / UART4_RXD / PWM1_CH1
37	PC.3 / EBI_AD3 / QSPI0_SS / UART2_nRTS / I2C0_SMBAL / UART3_TXD / PWM1_CH2
38	PC.2 / EBI_AD2 / QSPI0_CLK / UART2_nCTS / I2C0_SMBSUS / UART3_RXD / PWM1_CH3
39	PC.1 / EBI_AD1 / QSPI0_MISO0 / UART2_TXD / I2C0_SCL / PWM1_CH4 / ACMP0_O / ADC0_ST
40	PC.0 / EBI_AD0 / QSPI0_MOSI0 / UART2_RXD / I2C0_SDA / PWM1_CH5 / ACMP1_O
41	PD.3 / EBI_AD10 / USCI0_CTL1 / SPI0_SS / UART3_nRTS / USCI1_CTL0 / UART0_TXD
42	PD.2 / EBI_AD11 / USCI0_DAT1 / SPI0_CLK / UART3_nCTS / UART0_RXD
43	PD.1 / EBI_AD12 / USCI0_DAT0 / SPI0_MISO / UART3_TXD
44	PD.0 / EBI_AD13 / USCI0_CLK / SPI0_MOSI / UART3_RXD / TM2
45	USB_VBUS
46	USB_D-
47	USB_D+
48	USB_V_{DD}33_CAP
49	VSS
50	LDO_CAP
51	V_{DD}
52	PC.14 / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / TM1
53	PB.15 / ADC0_CH15 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / PWM1_CH0 / TM0_EXT / PWM0_BRAKE1
54	PB.14 / ADC0_CH14 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / PWM1_CH1 / TM1_EXT / CLKO
55	PB.13 / ADC0_CH13 / ACMP0_P3 / ACMP1_P3 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / PWM1_CH2 / TM2_EXT
56	PB.12 / ADC0_CH12 / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / PWM1_CH3 / TM3_EXT
57	AV_{DD}
58	V_{REF}
59	AVSS
60	PB.11 / ADC0_CH11 / EBI_ADR16 / UART0_nCTS / UART4_TXD / I2C1_SCL / SPI0_I2SMCLK / BPWM1_CH0
61	PB.10 / ADC0_CH10 / EBI_ADR17 / USCI1_CTL0 / UART0_nRTS / UART4_RXD / I2C1_SDA / BPWM1_CH1
62	PB.9 / ADC0_CH9 / EBI_ADR18 / USCI1_CTL1 / UART0_TXD / UART1_nCTS / BPWM1_CH2
63	PB.8 / ADC0_CH8 / EBI_ADR19 / USCI1_CLK / UART0_RXD / UART1_nRTS / BPWM1_CH3
64	PB.7 / ADC0_CH7 / EBI_nWRL / USCI1_DAT0 / UART1_TXD / EBI_nCS0 / BPWM1_CH4 / PWM1_BRAKE0 / PWM1_CH4 / INT5 / ACMP0_O

Table 4.1-34 M032SG8AE Multi-function Pin Table

M032SIAAE

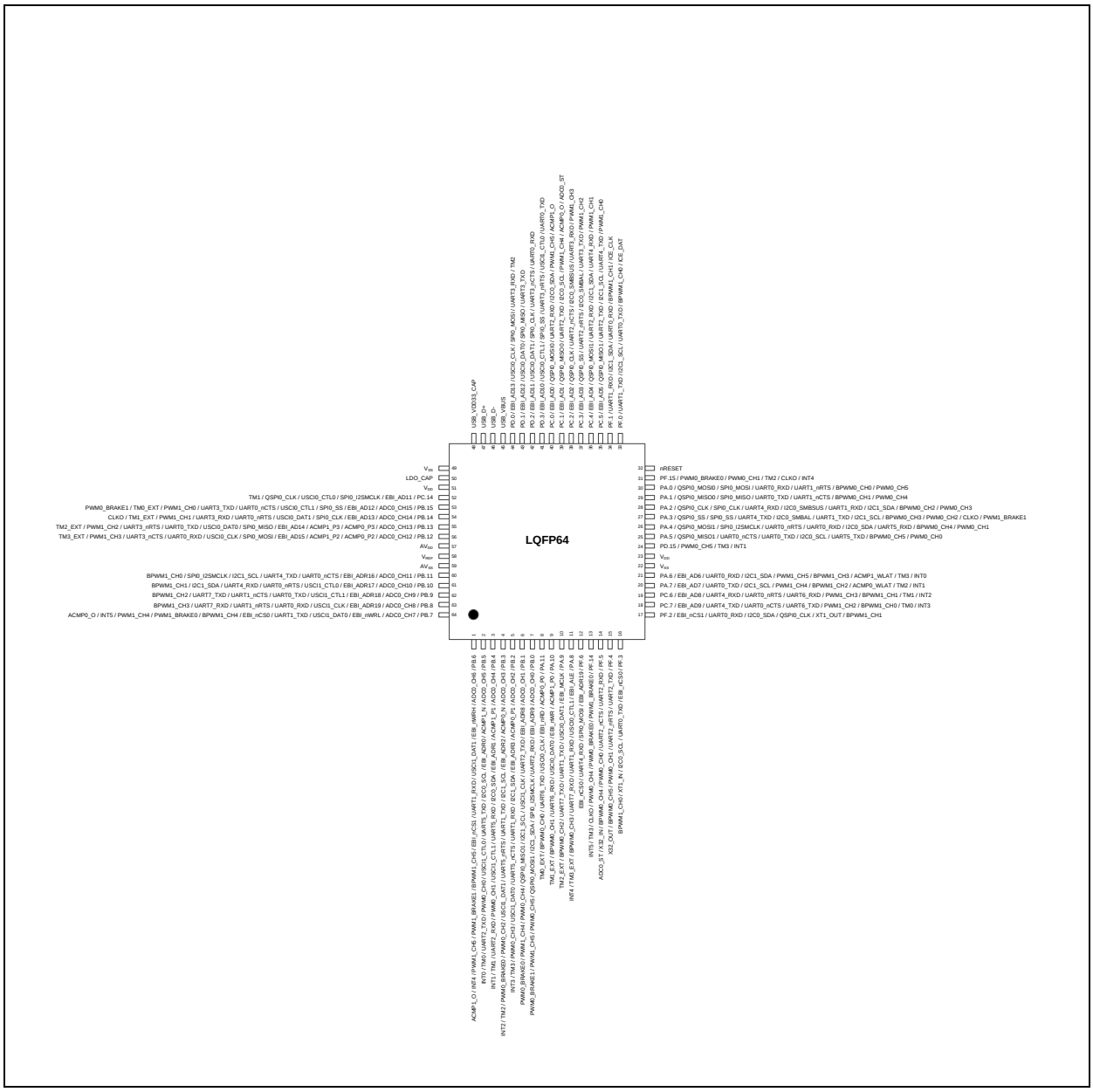


Figure 4.1-47 M032SIAAE Multi-function Pin Diagram

Pin	M032SIAAE Pin Function
1	PB.6 / ADC0_CH6 / EBI_nWRH / USCI1_DAT1 / UART1_RXD / EBI_nCS1 / BPWM1_CH5 / PWM1_BRAKE1 / PWM1_CH5 / INT4 / ACMP1_O
2	PB.5 / ADC0_CH5 / ACMP1_N / EBI_ADR0 / I2C0_SCL / UART5_TXD / USCI1_CTL0 / PWM0_CH0 / UART2_TXD / TM0 / INT0
3	PB.4 / ADC0_CH4 / ACMP1_P1 / EBI_ADR1 / I2C0_SDA / UART5_RXD / USCI1_CTL1 / PWM0_CH1 / UART2_RXD / TM1 / INT1
4	PB.3 / ADC0_CH3 / ACMP0_N / EBI_ADR2 / I2C1_SCL / UART1_TXD / UART5_nRTS / USCI1_DAT1 / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2

Pin	M032SIAAE Pin Function
5	PB.2 / ADC0_CH2 / ACMP0_P1 / EBI_ADR3 / I2C1_SDA / UART1_RXD / UART5_nCTS / USCI1_DAT0 / PWM0_CH3 / TM3 / INT3
6	PB.1 / ADC0_CH1 / EBI_ADR8 / UART2_TXD / USCI1_CLK / I2C1_SCL / QSPI0_MISO1 / PWM0_CH4 / PWM1_CH4 / PWM0_BRAKE0
7	PB.0 / ADC0_CH0 / EBI_ADR9 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / PWM0_CH5 / PWM1_CH5 / PWM0_BRAKE1
8	PA.11 / ACMP0_P0 / EBI_nRD / USCI0_CLK / UART6_TXD / BPWM0_CH0 / TM0_EXT
9	PA.10 / ACMP1_P0 / EBI_nWR / USCI0_DAT0 / UART6_RXD / BPWM0_CH1 / TM1_EXT
10	PA.9 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / UART7_TXD / BPWM0_CH2 / TM2_EXT
11	PA.8 / EBI_ALE / USCI0_CTL1 / UART1_RXD / UART7_RXD / BPWM0_CH3 / TM3_EXT / INT4
12	PF.6 / EBI_ADR19 / SPI0_MOSI / UART4_RXD / EBI_nCS0
13	PF.14 / PWM1_BRAKE0 / PWM0_BRAKE0 / PWM0_CH4 / CLKO / TM3 / INT5
14	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / BPWM0_CH4 / X32_IN / ADC0_ST
15	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1 / BPWM0_CH5 / X32_OUT
16	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / BPWM1_CH0
17	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / BPWM1_CH1
18	PC.7 / EBI_AD9 / UART4_TXD / UART0_nCTS / UART6_TXD / PWM1_CH2 / BPWM1_CH0 / TM0 / INT3
19	PC.6 / EBI_AD8 / UART4_RXD / UART0_nRTS / UART6_RXD / PWM1_CH3 / BPWM1_CH1 / TM1 / INT2
20	PA.7 / EBI_AD7 / UART0_TXD / I2C1_SCL / PWM1_CH4 / BPWM1_CH2 / ACMP0_WLAT / TM2 / INT1
21	PA.6 / EBI_AD6 / UART0_RXD / I2C1_SDA / PWM1_CH5 / BPWM1_CH3 / ACMP1_WLAT / TM3 / INT0
22	VSS
23	V_{DD}
24	PD.15 / PWM0_CH5 / TM3 / INT1
25	PA.5 / QSPI0_MISO1 / UART0_nCTS / UART0_TXD / I2C0_SCL / UART5_TXD / BPWM0_CH5 / PWM0_CH0
26	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / UART5_RXD / BPWM0_CH4 / PWM0_CH1
27	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / I2C0_SMBAL / UART1_TXD / I2C1_SCL / BPWM0_CH3 / PWM0_CH2 / CLKO / PWM1_BRAKE1
28	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / BPWM0_CH2 / PWM0_CH3
29	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / BPWM0_CH1 / PWM0_CH4
30	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / BPWM0_CH0 / PWM0_CH5
31	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / TM2 / CLKO / INT4
32	nRESET
33	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / BPWM1_CH0 / ICE_DAT
34	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / BPWM1_CH1 / ICE_CLK
35	PC.5 / EBI_AD5 / QSPI0_MISO1 / UART2_TXD / I2C1_SCL / UART4_TXD / PWM1_CH0

Pin	M032SIAAE Pin Function
36	PC.4 / EBI_AD4 / QSPI0_MOSI1 / UART2_RXD / I2C1_SDA / UART4_RXD / PWM1_CH1
37	PC.3 / EBI_AD3 / QSPI0_SS / UART2_nRTS / I2C0_SMBAL / UART3_TXD / PWM1_CH2
38	PC.2 / EBI_AD2 / QSPI0_CLK / UART2_nCTS / I2C0_SMBSUS / UART3_RXD / PWM1_CH3
39	PC.1 / EBI_AD1 / QSPI0_MISO0 / UART2_TXD / I2C0_SCL / PWM1_CH4 / ACMP0_O / ADC0_ST
40	PC.0 / EBI_AD0 / QSPI0_MOSI0 / UART2_RXD / I2C0_SDA / PWM1_CH5 / ACMP1_O
41	PD.3 / EBI_AD10 / USCI0_CTL1 / SPI0_SS / UART3_nRTS / USCI1_CTL0 / UART0_TXD
42	PD.2 / EBI_AD11 / USCI0_DAT1 / SPI0_CLK / UART3_nCTS / UART0_RXD
43	PD.1 / EBI_AD12 / USCI0_DAT0 / SPI0_MISO / UART3_TXD
44	PD.0 / EBI_AD13 / USCI0_CLK / SPI0_MOSI / UART3_RXD / TM2
45	USB_VBUS
46	USB_D-
47	USB_D+
48	USB_V_{DD}33_CAP
49	VSS
50	LDO_CAP
51	V_{DD}
52	PC.14 / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / TM1
53	PB.15 / ADC0_CH15 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / PWM1_CH0 / TM0_EXT / PWM0_BRAKE1
54	PB.14 / ADC0_CH14 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / PWM1_CH1 / TM1_EXT / CLKO
55	PB.13 / ADC0_CH13 / ACMP0_P3 / ACMP1_P3 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / PWM1_CH2 / TM2_EXT
56	PB.12 / ADC0_CH12 / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / PWM1_CH3 / TM3_EXT
57	AV_{DD}
58	V_{REF}
59	AVSS
60	PB.11 / ADC0_CH11 / EBI_ADR16 / UART0_nCTS / UART4_TXD / I2C1_SCL / SPI0_I2SMCLK / BPWM1_CH0
61	PB.10 / ADC0_CH10 / EBI_ADR17 / USCI1_CTL0 / UART0_nRTS / UART4_RXD / I2C1_SDA / BPWM1_CH1
62	PB.9 / ADC0_CH9 / EBI_ADR18 / USCI1_CTL1 / UART0_TXD / UART1_nCTS / UART7_TXD / BPWM1_CH2
63	PB.8 / ADC0_CH8 / EBI_ADR19 / USCI1_CLK / UART0_RXD / UART1_nRTS / UART7_RXD / BPWM1_CH3
64	PB.7 / ADC0_CH7 / EBI_nWRL / USCI1_DAT0 / UART1_TXD / EBI_nCS0 / BPWM1_CH4 / PWM1_BRAKE0 / PWM1_CH4 / INT5 / ACMP0_O

Table 4.1-35 M032SIAAE Multi-function Pin Table

4.1.4.6 M032 Series LQFP 128-Pin Multi-function Pin Diagram

Corresponding Part Number: M032KG6AE, M032KG8AE, M032KIAAE

M032KG6AE

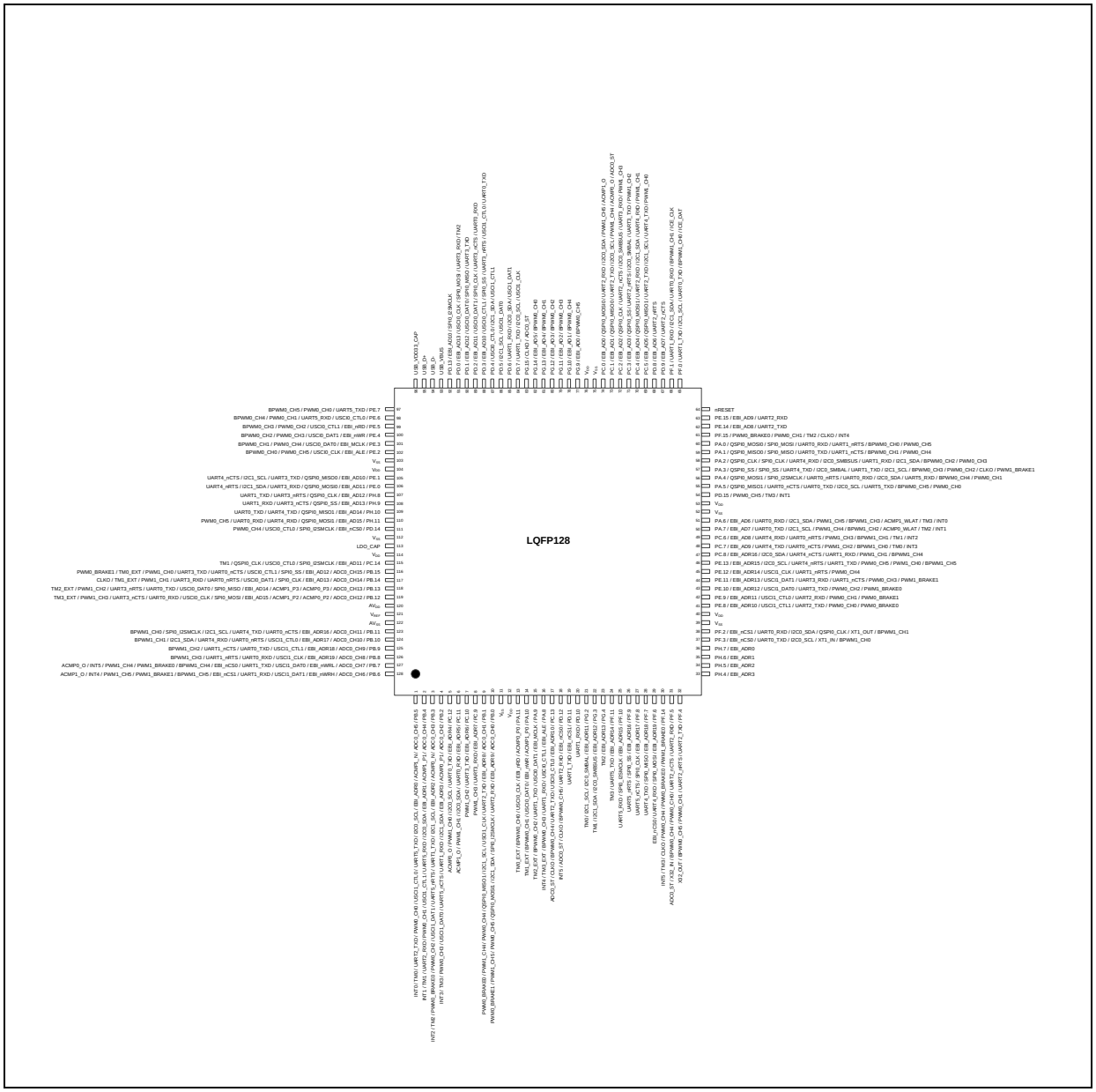


Figure 4.1-48 M032KG6AE Multi-function Pin Diagram

Pin	M032KG6AE Pin Function
1	PB.5 / ADC0_CH5 / ACMP1_N / EBI_ADR0 / I2C0_SCL / UART5_TXD / USCI1_CTL0 / PWM0_CH0 / UART2_TXD / TM0 / INT0
2	PB.4 / ADC0_CH4 / ACMP1_P1 / EBI_ADR1 / I2C0_SDA / UART5_RXD / USCI1_CTL1 / PWM0_CH1 / UART2_RXD / TM1 / INT1

Pin	M032KG6AE Pin Function
3	PB.3 / ADC0_CH3 / ACMP0_N / EBI_ADR2 / I2C1_SCL / UART1_TXD / UART5_nRTS / USCI1_DAT1 / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2
4	PB.2 / ADC0_CH2 / ACMP0_P1 / EBI_ADR3 / I2C1_SDA / UART1_RXD / UART5_nCTS / USCI1_DAT0 / PWM0_CH3 / TM3 / INT3
5	PC.12 / EBI_ADR4 / UART0_TXD / I2C0_SCL / PWM1_CH0 / ACMP0_O
6	PC.11 / EBI_ADR5 / UART0_RXD / I2C0_SDA / PWM1_CH1 / ACMP1_O
7	PC.10 / EBI_ADR6 / UART3_TXD / PWM1_CH2
8	PC.9 / EBI_ADR7 / UART3_RXD / PWM1_CH3
9	PB.1 / ADC0_CH1 / EBI_ADR8 / UART2_TXD / USCI1_CLK / I2C1_SCL / QSPI0_MISO1 / PWM0_CH4 / PWM1_CH4 / PWM0_BRAKE0
10	PB.0 / ADC0_CH0 / EBI_ADR9 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / PWM0_CH5 / PWM1_CH5 / PWM0_BRAKE1
11	VSS
12	V_{DD}
13	PA.11 / ACMP0_P0 / EBI_nRD / USCI0_CLK / BPWM0_CH0 / TM0_EXT
14	PA.10 / ACMP1_P0 / EBI_nWR / USCI0_DAT0 / BPWM0_CH1 / TM1_EXT
15	PA.9 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / BPWM0_CH2 / TM2_EXT
16	PA.8 / EBI_ALE / USCI0_CTL1 / UART1_RXD / BPWM0_CH3 / TM3_EXT / INT4
17	PC.13 / EBI_ADR10 / USCI0_CTL0 / UART2_TXD / BPWM0_CH4 / CLKO / ADC0_ST
18	PD.12 / EBI_nCS0 / UART2_RXD / BPWM0_CH5 / CLKO / ADC0_ST / INT5
19	PD.11 / EBI_nCS1 / UART1_TXD
20	PD.10 / UART1_RXD
21	PG.2 / EBI_ADR11 / I2C0_SMBAL / I2C1_SCL / TM0
22	PG.3 / EBI_ADR12 / I2C0_SMBSUS / I2C1_SDA / TM1
23	PG.4 / EBI_ADR13 / TM2
24	PF.11 / EBI_ADR14 / UART5_TXD / TM3
25	PF.10 / EBI_ADR15 / SPI0_I2SMCLK / UART5_RXD
26	PF.9 / EBI_ADR16 / SPI0_SS / UART5_nRTS
27	PF.8 / EBI_ADR17 / SPI0_CLK / UART5_nCTS
28	PF.7 / EBI_ADR18 / SPI0_MISO / UART4_TXD
29	PF.6 / EBI_ADR19 / SPI0_MOSI / UART4_RXD / EBI_nCS0
30	PF.14 / PWM1_BRAKE0 / PWM0_BRAKE0 / PWM0_CH4 / CLKO / TM3 / INT5
31	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / BPWM0_CH4 / X32_IN / ADC0_ST
32	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1 / BPWM0_CH5 / X32_OUT
33	PH.4 / EBI_ADR3
34	PH.5 / EBI_ADR2

Pin	M032KG6AE Pin Function
35	PH.6 / EBI_ADR1
36	PH.7 / EBI_ADR0
37	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / BPWM1_CH0
38	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / BPWM1_CH1
39	VSS
40	V_{DD}
41	PE.8 / EBI_ADR10 / USCI1_CTL1 / UART2_TXD / PWM0_CH0 / PWM0_BRAKE0
42	PE.9 / EBI_ADR11 / USCI1_CTL0 / UART2_RXD / PWM0_CH1 / PWM0_BRAKE1
43	PE.10 / EBI_ADR12 / USCI1_DAT0 / UART3_TXD / PWM0_CH2 / PWM1_BRAKE0
44	PE.11 / EBI_ADR13 / USCI1_DAT1 / UART3_RXD / UART1_nCTS / PWM0_CH3 / PWM1_BRAKE1
45	PE.12 / EBI_ADR14 / USCI1_CLK / UART1_nRTS / PWM0_CH4
46	PE.13 / EBI_ADR15 / I2C0_SCL / UART4_nRTS / UART1_TXD / PWM0_CH5 / PWM1_CH0 / BPWM1_CH5
47	PC.8 / EBI_ADR16 / I2C0_SDA / UART4_nCTS / UART1_RXD / PWM1_CH1 / BPWM1_CH4
48	PC.7 / EBI_AD9 / UART4_TXD / UART0_nCTS / PWM1_CH2 / BPWM1_CH0 / TM0 / INT3
49	PC.6 / EBI_AD8 / UART4_RXD / UART0_nRTS / PWM1_CH3 / BPWM1_CH1 / TM1 / INT2
50	PA.7 / EBI_AD7 / UART0_TXD / I2C1_SCL / PWM1_CH4 / BPWM1_CH2 / ACMP0_WLAT / TM2 / INT1
51	PA.6 / EBI_AD6 / UART0_RXD / I2C1_SDA / PWM1_CH5 / BPWM1_CH3 / ACMP1_WLAT / TM3 / INT0
52	VSS
53	V_{DD}
54	PD.15 / PWM0_CH5 / TM3 / INT1
55	PA.5 / QSPI0_MISO1 / UART0_nCTS / UART0_TXD / I2C0_SCL / UART5_TXD / BPWM0_CH5 / PWM0_CH0
56	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / UART5_RXD / BPWM0_CH4 / PWM0_CH1
57	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / I2C0_SMBAL / UART1_TXD / I2C1_SCL / BPWM0_CH3 / PWM0_CH2 / CLKO / PWM1_BRAKE1
58	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / BPWM0_CH2 / PWM0_CH3
59	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / BPWM0_CH1 / PWM0_CH4
60	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / BPWM0_CH0 / PWM0_CH5
61	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / TM2 / CLKO / INT4
62	PE.14 / EBI_AD8 / UART2_TXD
63	PE.15 / EBI_AD9 / UART2_RXD
64	nRESET
65	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / BPWM1_CH0 / ICE_DAT
66	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / BPWM1_CH1 / ICE_CLK
67	PD.9 / EBI_AD7 / UART2_nCTS

Pin	M032KG6AE Pin Function
68	PD.8 / EBI_AD6 / UART2_nRTS
69	PC.5 / EBI_AD5 / QSPI0_MISO1 / UART2_TXD / I2C1_SCL / UART4_TXD / PWM1_CH0
70	PC.4 / EBI_AD4 / QSPI0_MOSI1 / UART2_RXD / I2C1_SDA / UART4_RXD / PWM1_CH1
71	PC.3 / EBI_AD3 / QSPI0_SS / UART2_nRTS / I2C0_SMBAL / UART3_TXD / PWM1_CH2
72	PC.2 / EBI_AD2 / QSPI0_CLK / UART2_nCTS / I2C0_SMBSUS / UART3_RXD / PWM1_CH3
73	PC.1 / EBI_AD1 / QSPI0_MISO0 / UART2_TXD / I2C0_SCL / PWM1_CH4 / ACMP0_O / ADC0_ST
74	PC.0 / EBI_AD0 / QSPI0_MOSI0 / UART2_RXD / I2C0_SDA / PWM1_CH5 / ACMP1_O
75	VSS
76	V_{DD}
77	PG.9 / EBI_AD0 / BPWM0_CH5
78	PG.10 / EBI_AD1 / BPWM0_CH4
79	PG.11 / EBI_AD2 / BPWM0_CH3
80	PG.12 / EBI_AD3 / BPWM0_CH2
81	PG.13 / EBI_AD4 / BPWM0_CH1
82	PG.14 / EBI_AD5 / BPWM0_CH0
83	PG.15 / CLKO / ADC0_ST
84	PD.7 / UART1_TXD / I2C0_SCL / USCI1_CLK
85	PD.6 / UART1_RXD / I2C0_SDA / USCI1_DAT1
86	PD.5 / I2C1_SCL / USCI1_DAT0
87	PD.4 / USCI0_CTL0 / I2C1_SDA / USCI1_CTL1
88	PD.3 / EBI_AD10 / USCI0_CTL1 / SPI0_SS / UART3_nRTS / USCI1_CTL0 / UART0_TXD
89	PD.2 / EBI_AD11 / USCI0_DAT1 / SPI0_CLK / UART3_nCTS / UART0_RXD
90	PD.1 / EBI_AD12 / USCI0_DAT0 / SPI0_MISO / UART3_TXD
91	PD.0 / EBI_AD13 / USCI0_CLK / SPI0_MOSI / UART3_RXD / TM2
92	PD.13 / EBI_AD10 / SPI0_I2SMCLK
93	USB_VBUS
94	USB_D-
95	USB_D+
96	USB_V_{DD}33_CAP
97	PE.7 / UART5_TXD / PWM0_CH0 / BPWM0_CH5
98	PE.6 / USCI0_CTL0 / UART5_RXD / PWM0_CH1 / BPWM0_CH4
99	PE.5 / EBI_nRD / USCI0_CTL1 / PWM0_CH2 / BPWM0_CH3
100	PE.4 / EBI_nWR / USCI0_DAT1 / PWM0_CH3 / BPWM0_CH2
101	PE.3 / EBI_MCLK / USCI0_DAT0 / PWM0_CH4 / BPWM0_CH1

Pin	M032KG6AE Pin Function
102	PE.2 / EBI_ALE / USCI0_CLK / PWM0_CH5 / BPWM0_CH0
103	VSS
104	V_{DD}
105	PE.1 / EBI_AD10 / QSPI0_MISO0 / UART3_TXD / I2C1_SCL / UART4_nCTS
106	PE.0 / EBI_AD11 / QSPI0_MOSI0 / UART3_RXD / I2C1_SDA / UART4_nRTS
107	PH.8 / EBI_AD12 / QSPI0_CLK / UART3_nRTS / UART1_TXD
108	PH.9 / EBI_AD13 / QSPI0_SS / UART3_nCTS / UART1_RXD
109	PH.10 / EBI_AD14 / QSPI0_MISO1 / UART4_TXD / UART0_TXD
110	PH.11 / EBI_AD15 / QSPI0_MOSI1 / UART4_RXD / UART0_RXD / PWM0_CH5
111	PD.14 / EBI_nCS0 / SPI0_I2SMCLK / USCI0_CTL0 / PWM0_CH4
112	VSS
113	LDO_CAP
114	V_{DD}
115	PC.14 / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / TM1
116	PB.15 / ADC0_CH15 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / PWM1_CH0 / TM0_EXT / PWM0_BRAKE1
117	PB.14 / ADC0_CH14 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / PWM1_CH1 / TM1_EXT / CLKO
118	PB.13 / ADC0_CH13 / ACMP0_P3 / ACMP1_P3 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / PWM1_CH2 / TM2_EXT
119	PB.12 / ADC0_CH12 / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / PWM1_CH3 / TM3_EXT
120	AV_{DD}
121	V_{REF}
122	AVSS
123	PB.11 / ADC0_CH11 / EBI_ADR16 / UART0_nCTS / UART4_TXD / I2C1_SCL / SPI0_I2SMCLK / BPWM1_CH0
124	PB.10 / ADC0_CH10 / EBI_ADR17 / USCI1_CTL0 / UART0_nRTS / UART4_RXD / I2C1_SDA / BPWM1_CH1
125	PB.9 / ADC0_CH9 / EBI_ADR18 / USCI1_CTL1 / UART0_TXD / UART1_nCTS / BPWM1_CH2
126	PB.8 / ADC0_CH8 / EBI_ADR19 / USCI1_CLK / UART0_RXD / UART1_nRTS / BPWM1_CH3
127	PB.7 / ADC0_CH7 / EBI_nWRL / USCI1_DAT0 / UART1_TXD / EBI_nCS0 / BPWM1_CH4 / PWM1_BRAKE0 / PWM1_CH4 / INT5 / ACMP0_O
128	PB.6 / ADC0_CH6 / EBI_nWRH / USCI1_DAT1 / UART1_RXD / EBI_nCS1 / BPWM1_CH5 / PWM1_BRAKE1 / PWM1_CH5 / INT4 / ACMP1_O

Table 4.1-36 M032KG6AE Multi-function Pin Table

M032KG8AE

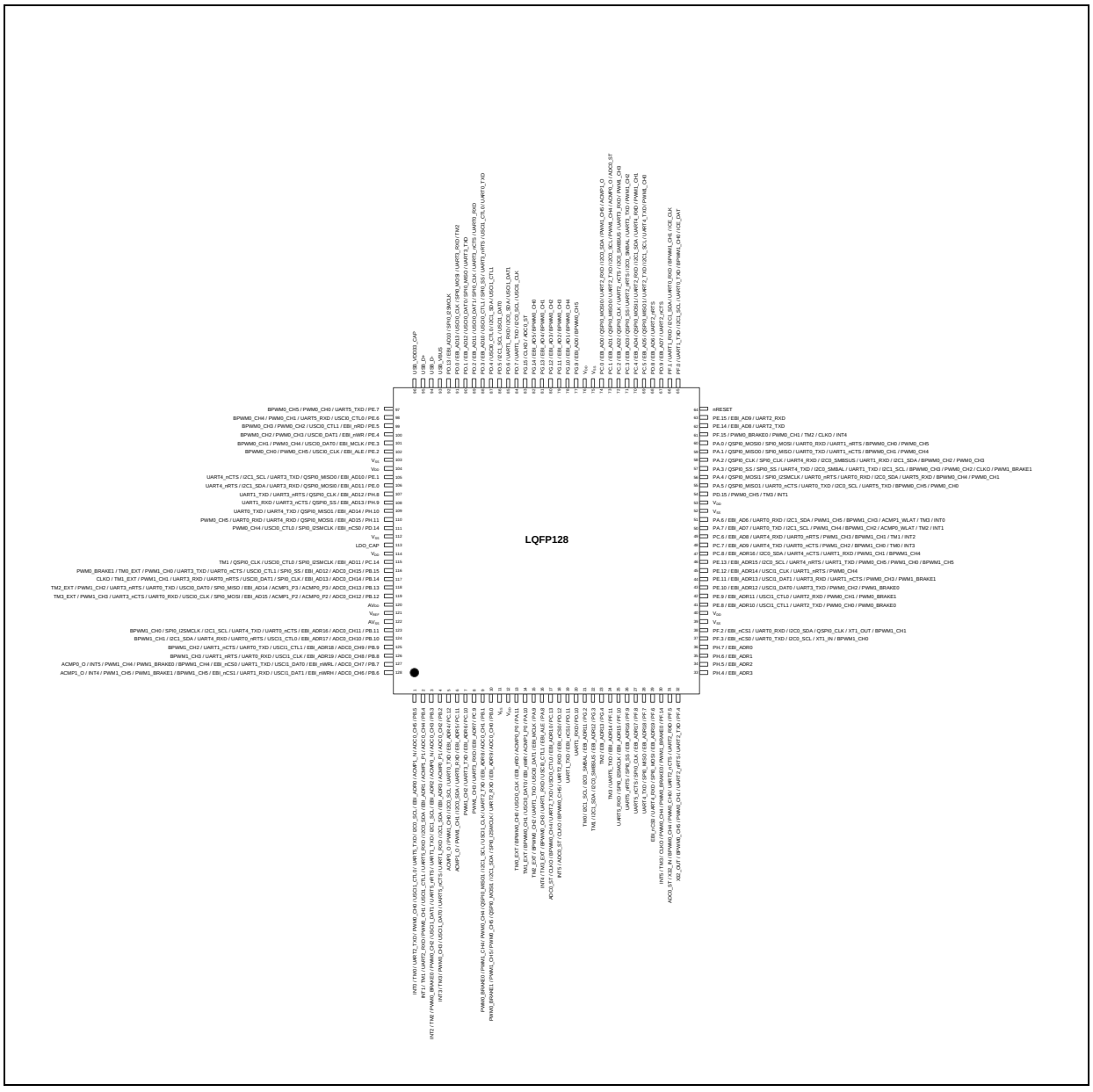


Figure 4.1-49 M032KG8AE Multi-function Pin Diagram

Pin	M032KG8AE Pin Function
1	PB.5 / ADC0_CH5 / ACMP1_N / EBI_ADR0 / I2C0_SCL / UART5_TXD / USCI1_CTL0 / PWM0_CH0 / UART2_TXD / TM0 / INT0
2	PB.4 / ADC0_CH4 / ACMP1_P1 / EBI_ADR1 / I2C0_SDA / UART5_RXD / USCI1_CTL1 / PWM0_CH1 / UART2_RXD / TM1 / INT1
3	PB.3 / ADC0_CH3 / ACMP0_N / EBI_ADR2 / I2C1_SCL / UART1_TXD / UART5_nRTS / USCI1_DAT1 / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2
4	PB.2 / ADC0_CH2 / ACMP0_P1 / EBI_ADR3 / I2C1_SDA / UART1_RXD / UART5_nCTS / USCI1_DAT0 /

Pin	M032KG8AE Pin Function
	PWM0_CH3 / TM3 / INT3
5	PC.12 / EBI_ADR4 / UART0_TXD / I2C0_SCL / PWM1_CH0 / ACMP0_O
6	PC.11 / EBI_ADR5 / UART0_RXD / I2C0_SDA / PWM1_CH1 / ACMP1_O
7	PC.10 / EBI_ADR6 / UART3_TXD / PWM1_CH2
8	PC.9 / EBI_ADR7 / UART3_RXD / PWM1_CH3
9	PB.1 / ADC0_CH1 / EBI_ADR8 / UART2_TXD / USCI1_CLK / I2C1_SCL / QSPI0_MISO1 / PWM0_CH4 / PWM1_CH4 / PWM0_BRAKE0
10	PB.0 / ADC0_CH0 / EBI_ADR9 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / PWM0_CH5 / PWM1_CH5 / PWM0_BRAKE1
11	VSS
12	V_{DD}
13	PA.11 / ACMP0_P0 / EBI_nRD / USCI0_CLK / BPWM0_CH0 / TM0_EXT
14	PA.10 / ACMP1_P0 / EBI_nWR / USCI0_DAT0 / BPWM0_CH1 / TM1_EXT
15	PA.9 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / BPWM0_CH2 / TM2_EXT
16	PA.8 / EBI_ALE / USCI0_CTL1 / UART1_RXD / BPWM0_CH3 / TM3_EXT / INT4
17	PC.13 / EBI_ADR10 / USCI0_CTL0 / UART2_TXD / BPWM0_CH4 / CLKO / ADC0_ST
18	PD.12 / EBI_nCS0 / UART2_RXD / BPWM0_CH5 / CLKO / ADC0_ST / INT5
19	PD.11 / EBI_nCS1 / UART1_TXD
20	PD.10 / UART1_RXD
21	PG.2 / EBI_ADR11 / I2C0_SMBAL / I2C1_SCL / TM0
22	PG.3 / EBI_ADR12 / I2C0_SMBSUS / I2C1_SDA / TM1
23	PG.4 / EBI_ADR13 / TM2
24	PF.11 / EBI_ADR14 / UART5_TXD / TM3
25	PF.10 / EBI_ADR15 / SPI0_I2SMCLK / UART5_RXD
26	PF.9 / EBI_ADR16 / SPI0_SS / UART5_nRTS
27	PF.8 / EBI_ADR17 / SPI0_CLK / UART5_nCTS
28	PF.7 / EBI_ADR18 / SPI0_MISO / UART4_TXD
29	PF.6 / EBI_ADR19 / SPI0_MOSI / UART4_RXD / EBI_nCS0
30	PF.14 / PWM1_BRAKE0 / PWM0_BRAKE0 / PWM0_CH4 / CLKO / TM3 / INT5
31	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / BPWM0_CH4 / X32_IN / ADC0_ST
32	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1 / BPWM0_CH5 / X32_OUT
33	PH.4 / EBI_ADR3
34	PH.5 / EBI_ADR2
35	PH.6 / EBI_ADR1
36	PH.7 / EBI_ADR0
37	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / BPWM1_CH0

Pin	M032KG8AE Pin Function
38	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / BPWM1_CH1
39	VSS
40	V_{DD}
41	PE.8 / EBI_ADR10 / USCI1_CTL1 / UART2_TXD / PWM0_CH0 / PWM0_BRAKE0
42	PE.9 / EBI_ADR11 / USCI1_CTL0 / UART2_RXD / PWM0_CH1 / PWM0_BRAKE1
43	PE.10 / EBI_ADR12 / USCI1_DAT0 / UART3_TXD / PWM0_CH2 / PWM1_BRAKE0
44	PE.11 / EBI_ADR13 / USCI1_DAT1 / UART3_RXD / UART1_nCTS / PWM0_CH3 / PWM1_BRAKE1
45	PE.12 / EBI_ADR14 / USCI1_CLK / UART1_nRTS / PWM0_CH4
46	PE.13 / EBI_ADR15 / I2C0_SCL / UART4_nRTS / UART1_TXD / PWM0_CH5 / PWM1_CH0 / BPWM1_CH5
47	PC.8 / EBI_ADR16 / I2C0_SDA / UART4_nCTS / UART1_RXD / PWM1_CH1 / BPWM1_CH4
48	PC.7 / EBI_AD9 / UART4_TXD / UART0_nCTS / PWM1_CH2 / BPWM1_CH0 / TM0 / INT3
49	PC.6 / EBI_AD8 / UART4_RXD / UART0_nRTS / PWM1_CH3 / BPWM1_CH1 / TM1 / INT2
50	PA.7 / EBI_AD7 / UART0_TXD / I2C1_SCL / PWM1_CH4 / BPWM1_CH2 / ACMP0_WLAT / TM2 / INT1
51	PA.6 / EBI_AD6 / UART0_RXD / I2C1_SDA / PWM1_CH5 / BPWM1_CH3 / ACMP1_WLAT / TM3 / INT0
52	VSS
53	V_{DD}
54	PD.15 / PWM0_CH5 / TM3 / INT1
55	PA.5 / QSPI0_MISO1 / UART0_nCTS / UART0_TXD / I2C0_SCL / UART5_TXD / BPWM0_CH5 / PWM0_CH0
56	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / UART5_RXD / BPWM0_CH4 / PWM0_CH1
57	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / I2C0_SMBAL / UART1_TXD / I2C1_SCL / BPWM0_CH3 / PWM0_CH2 / CLKO / PWM1_BRAKE1
58	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / BPWM0_CH2 / PWM0_CH3
59	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / BPWM0_CH1 / PWM0_CH4
60	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / BPWM0_CH0 / PWM0_CH5
61	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / TM2 / CLKO / INT4
62	PE.14 / EBI_AD8 / UART2_TXD
63	PE.15 / EBI_AD9 / UART2_RXD
64	nRESET
65	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / BPWM1_CH0 / ICE_DAT
66	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / BPWM1_CH1 / ICE_CLK
67	PD.9 / EBI_AD7 / UART2_nCTS
68	PD.8 / EBI_AD6 / UART2_nRTS
69	PC.5 / EBI_AD5 / QSPI0_MISO1 / UART2_TXD / I2C1_SCL / UART4_TXD / PWM1_CH0
70	PC.4 / EBI_AD4 / QSPI0_MOSI1 / UART2_RXD / I2C1_SDA / UART4_RXD / PWM1_CH1

Pin	M032KG8AE Pin Function
71	PC.3 / EBI_AD3 / QSPI0_SS / UART2_nRTS / I2C0_SMBAL / UART3_TXD / PWM1_CH2
72	PC.2 / EBI_AD2 / QSPI0_CLK / UART2_nCTS / I2C0_SMBSUS / UART3_RXD / PWM1_CH3
73	PC.1 / EBI_AD1 / QSPI0_MISO0 / UART2_TXD / I2C0_SCL / PWM1_CH4 / ACMP0_O / ADC0_ST
74	PC.0 / EBI_AD0 / QSPI0_MOSI0 / UART2_RXD / I2C0_SDA / PWM1_CH5 / ACMP1_O
75	VSS
76	V_{DD}
77	PG.9 / EBI_AD0 / BPWM0_CH5
78	PG.10 / EBI_AD1 / BPWM0_CH4
79	PG.11 / EBI_AD2 / BPWM0_CH3
80	PG.12 / EBI_AD3 / BPWM0_CH2
81	PG.13 / EBI_AD4 / BPWM0_CH1
82	PG.14 / EBI_AD5 / BPWM0_CH0
83	PG.15 / CLKO / ADC0_ST
84	PD.7 / UART1_TXD / I2C0_SCL / USCI1_CLK
85	PD.6 / UART1_RXD / I2C0_SDA / USCI1_DAT1
86	PD.5 / I2C1_SCL / USCI1_DAT0
87	PD.4 / USCI0_CTL0 / I2C1_SDA / USCI1_CTL1
88	PD.3 / EBI_AD10 / USCI0_CTL1 / SPI0_SS / UART3_nRTS / USCI1_CTL0 / UART0_TXD
89	PD.2 / EBI_AD11 / USCI0_DAT1 / SPI0_CLK / UART3_nCTS / UART0_RXD
90	PD.1 / EBI_AD12 / USCI0_DAT0 / SPI0_MISO / UART3_TXD
91	PD.0 / EBI_AD13 / USCI0_CLK / SPI0_MOSI / UART3_RXD / TM2
92	PD.13 / EBI_AD10 / SPI0_I2SMCLK
93	USB_VBUS
94	USB_D-
95	USB_D+
96	USB_V_{DD}33_CAP
97	PE.7 / UART5_TXD / PWM0_CH0 / BPWM0_CH5
98	PE.6 / USCI0_CTL0 / UART5_RXD / PWM0_CH1 / BPWM0_CH4
99	PE.5 / EBI_nRD / USCI0_CTL1 / PWM0_CH2 / BPWM0_CH3
100	PE.4 / EBI_nWR / USCI0_DAT1 / PWM0_CH3 / BPWM0_CH2
101	PE.3 / EBI_MCLK / USCI0_DAT0 / PWM0_CH4 / BPWM0_CH1
102	PE.2 / EBI_ALE / USCI0_CLK / PWM0_CH5 / BPWM0_CH0
103	VSS
104	V_{DD}

Pin	M032KG8AE Pin Function
105	PE.1 / EBI_AD10 / QSPI0_MISO0 / UART3_TXD / I2C1_SCL / UART4_nCTS
106	PE.0 / EBI_AD11 / QSPI0_MOSI0 / UART3_RXD / I2C1_SDA / UART4_nRTS
107	PH.8 / EBI_AD12 / QSPI0_CLK / UART3_nRTS / UART1_TXD
108	PH.9 / EBI_AD13 / QSPI0_SS / UART3_nCTS / UART1_RXD
109	PH.10 / EBI_AD14 / QSPI0_MISO1 / UART4_TXD / UART0_TXD
110	PH.11 / EBI_AD15 / QSPI0_MOSI1 / UART4_RXD / UART0_RXD / PWM0_CH5
111	PD.14 / EBI_nCS0 / SPI0_I2SMCLK / USCI0_CTL0 / PWM0_CH4
112	VSS
113	LDO_CAP
114	V_{DD}
115	PC.14 / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / TM1
116	PB.15 / ADC0_CH15 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / PWM1_CH0 / TM0_EXT / PWM0_BRAKE1
117	PB.14 / ADC0_CH14 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / PWM1_CH1 / TM1_EXT / CLKO
118	PB.13 / ADC0_CH13 / ACMP0_P3 / ACMP1_P3 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / PWM1_CH2 / TM2_EXT
119	PB.12 / ADC0_CH12 / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / PWM1_CH3 / TM3_EXT
120	AV_{DD}
121	V_{REF}
122	AVSS
123	PB.11 / ADC0_CH11 / EBI_ADR16 / UART0_nCTS / UART4_TXD / I2C1_SCL / SPI0_I2SMCLK / BPWM1_CH0
124	PB.10 / ADC0_CH10 / EBI_ADR17 / USCI1_CTL0 / UART0_nRTS / UART4_RXD / I2C1_SDA / BPWM1_CH1
125	PB.9 / ADC0_CH9 / EBI_ADR18 / USCI1_CTL1 / UART0_TXD / UART1_nCTS / BPWM1_CH2
126	PB.8 / ADC0_CH8 / EBI_ADR19 / USCI1_CLK / UART0_RXD / UART1_nRTS / BPWM1_CH3
127	PB.7 / ADC0_CH7 / EBI_nWRL / USCI1_DAT0 / UART1_TXD / EBI_nCS0 / BPWM1_CH4 / PWM1_BRAKE0 / PWM1_CH4 / INT5 / ACMP0_O
128	PB.6 / ADC0_CH6 / EBI_nWRH / USCI1_DAT1 / UART1_RXD / EBI_nCS1 / BPWM1_CH5 / PWM1_BRAKE1 / PWM1_CH5 / INT4 / ACMP1_O

Table 4.1-37 M032KG8AE Multi-function Pin Table

M032KIAAE

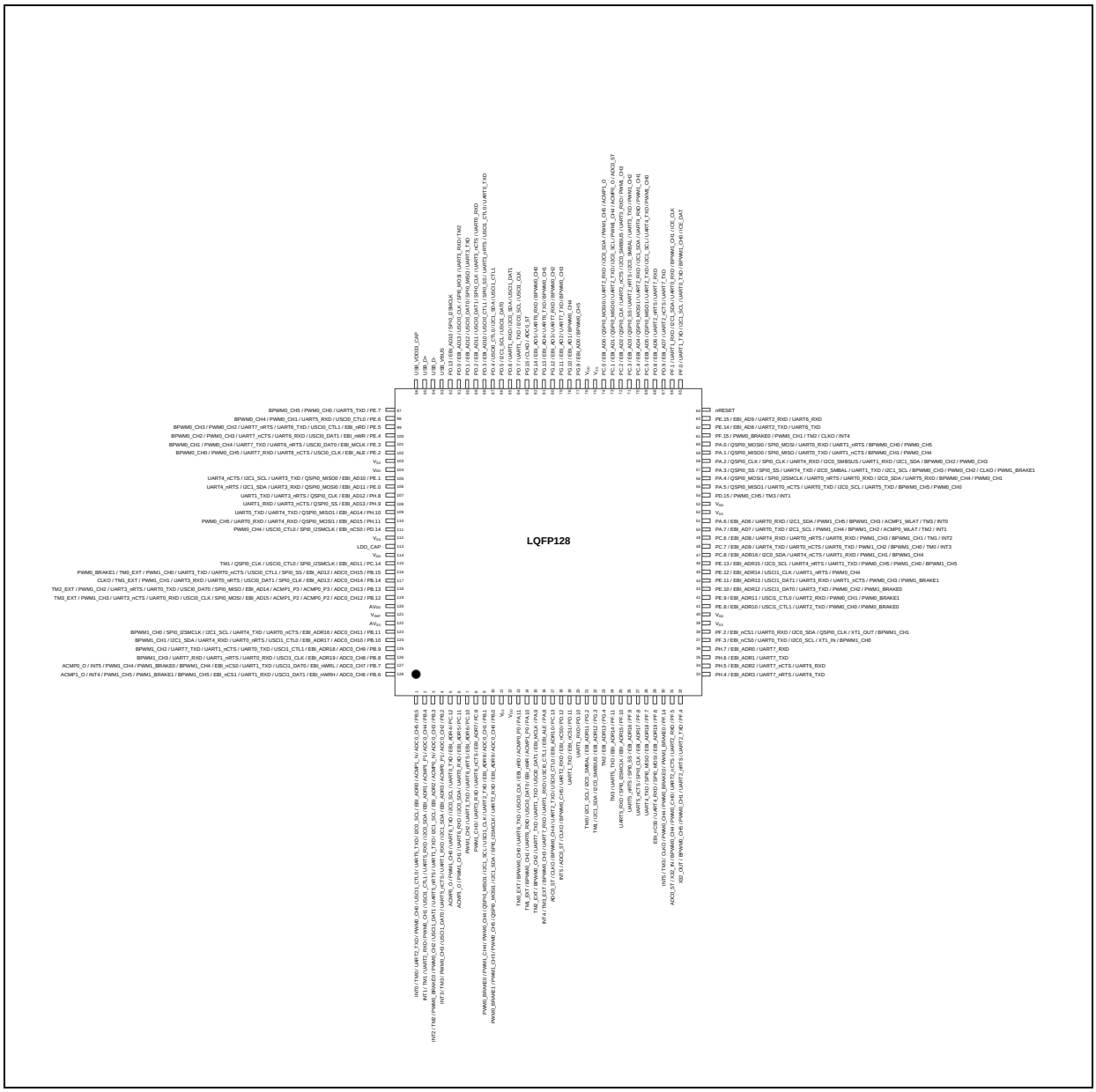


Figure 4.1-50 M032KIAAE Multi-function Pin Diagram

Pin	M032KIAAE Pin Function
1	PB.5 / ADC0_CH5 / ACMP1_N / EBI_ADR0 / I2C0_SCL / UART5_TXD / USCI1_CTL0 / PWM0_CH0 / UART2_TXD / TM0 / INT0
2	PB.4 / ADC0_CH4 / ACMP1_P1 / EBI_ADR1 / I2C0_SDA / UART5_RXD / USCI1_CTL1 / PWM0_CH1 / UART2_RXD / TM1 / INT1
3	PB.3 / ADC0_CH3 / ACMP0_N / EBI_ADR2 / I2C1_SCL / UART1_TXD / UART5_nRTS / USCI1_DAT1 / PWM0_CH2 / PWM0_BRAKE0 / TM2 / INT2
4	PB.2 / ADC0_CH2 / ACMP0_P1 / EBI_ADR3 / I2C1_SDA / UART1_RXD / UART5_nCTS / USCI1_DAT0 / PWM0_CH3 / TM3 / INT3

Pin	M032KIAAE Pin Function
5	PC.12 / EBI_ADR4 / UART0_TXD / I2C0_SCL / UART6_TXD / PWM1_CH0 / ACMP0_O
6	PC.11 / EBI_ADR5 / UART0_RXD / I2C0_SDA / UART6_RXD / PWM1_CH1 / ACMP1_O
7	PC.10 / EBI_ADR6 / UART6_nRTS / UART3_TXD / PWM1_CH2
8	PC.9 / EBI_ADR7 / UART6_nCTS / UART3_RXD / PWM1_CH3
9	PB.1 / ADC0_CH1 / EBI_ADR8 / UART2_TXD / USCI1_CLK / I2C1_SCL / QSPI0_MISO1 / PWM0_CH4 / PWM1_CH4 / PWM0_BRAKE0
10	PB.0 / ADC0_CH0 / EBI_ADR9 / UART2_RXD / SPI0_I2SMCLK / I2C1_SDA / QSPI0_MOSI1 / PWM0_CH5 / PWM1_CH5 / PWM0_BRAKE1
11	VSS
12	V_{DD}
13	PA.11 / ACMP0_P0 / EBI_nRD / USCI0_CLK / UART6_TXD / BPWM0_CH0 / TM0_EXT
14	PA.10 / ACMP1_P0 / EBI_nWR / USCI0_DAT0 / UART6_RXD / BPWM0_CH1 / TM1_EXT
15	PA.9 / EBI_MCLK / USCI0_DAT1 / UART1_TXD / UART7_TXD / BPWM0_CH2 / TM2_EXT
16	PA.8 / EBI_ALE / USCI0_CTL1 / UART1_RXD / UART7_RXD / BPWM0_CH3 / TM3_EXT / INT4
17	PC.13 / EBI_ADR10 / USCI0_CTL0 / UART2_TXD / BPWM0_CH4 / CLKO / ADC0_ST
18	PD.12 / EBI_nCS0 / UART2_RXD / BPWM0_CH5 / CLKO / ADC0_ST / INT5
19	PD.11 / EBI_nCS1 / UART1_TXD
20	PD.10 / UART1_RXD
21	PG.2 / EBI_ADR11 / I2C0_SMBAL / I2C1_SCL / TM0
22	PG.3 / EBI_ADR12 / I2C0_SMBSUS / I2C1_SDA / TM1
23	PG.4 / EBI_ADR13 / TM2
24	PF.11 / EBI_ADR14 / UART5_TXD / TM3
25	PF.10 / EBI_ADR15 / SPI0_I2SMCLK / UART5_RXD
26	PF.9 / EBI_ADR16 / SPI0_SS / UART5_nRTS
27	PF.8 / EBI_ADR17 / SPI0_CLK / UART5_nCTS
28	PF.7 / EBI_ADR18 / SPI0_MISO / UART4_TXD
29	PF.6 / EBI_ADR19 / SPI0_MOSI / UART4_RXD / EBI_nCS0
30	PF.14 / PWM1_BRAKE0 / PWM0_BRAKE0 / PWM0_CH4 / CLKO / TM3 / INT5
31	PF.5 / UART2_RXD / UART2_nCTS / PWM0_CH0 / BPWM0_CH4 / X32_IN / ADC0_ST
32	PF.4 / UART2_TXD / UART2_nRTS / PWM0_CH1 / BPWM0_CH5 / X32_OUT
33	PH.4 / EBI_ADR3 / UART7_nRTS / UART6_TXD
34	PH.5 / EBI_ADR2 / UART7_nCTS / UART6_RXD
35	PH.6 / EBI_ADR1 / UART7_TXD
36	PH.7 / EBI_ADR0 / UART7_RXD
37	PF.3 / EBI_nCS0 / UART0_TXD / I2C0_SCL / XT1_IN / BPWM1_CH0

Pin	M032KIAAE Pin Function
38	PF.2 / EBI_nCS1 / UART0_RXD / I2C0_SDA / QSPI0_CLK / XT1_OUT / BPWM1_CH1
39	VSS
40	V_{DD}
41	PE.8 / EBI_ADR10 / USCI1_CTL1 / UART2_TXD / PWM0_CH0 / PWM0_BRAKE0
42	PE.9 / EBI_ADR11 / USCI1_CTL0 / UART2_RXD / PWM0_CH1 / PWM0_BRAKE1
43	PE.10 / EBI_ADR12 / USCI1_DAT0 / UART3_TXD / PWM0_CH2 / PWM1_BRAKE0
44	PE.11 / EBI_ADR13 / USCI1_DAT1 / UART3_RXD / UART1_nCTS / PWM0_CH3 / PWM1_BRAKE1
45	PE.12 / EBI_ADR14 / USCI1_CLK / UART1_nRTS / PWM0_CH4
46	PE.13 / EBI_ADR15 / I2C0_SCL / UART4_nRTS / UART1_TXD / PWM0_CH5 / PWM1_CH0 / BPWM1_CH5
47	PC.8 / EBI_ADR16 / I2C0_SDA / UART4_nCTS / UART1_RXD / PWM1_CH1 / BPWM1_CH4
48	PC.7 / EBI_AD9 / UART4_TXD / UART0_nCTS / UART6_TXD / PWM1_CH2 / BPWM1_CH0 / TM0 / INT3
49	PC.6 / EBI_AD8 / UART4_RXD / UART0_nRTS / UART6_RXD / PWM1_CH3 / BPWM1_CH1 / TM1 / INT2
50	PA.7 / EBI_AD7 / UART0_TXD / I2C1_SCL / PWM1_CH4 / BPWM1_CH2 / ACMP0_WLAT / TM2 / INT1
51	PA.6 / EBI_AD6 / UART0_RXD / I2C1_SDA / PWM1_CH5 / BPWM1_CH3 / ACMP1_WLAT / TM3 / INT0
52	VSS
53	V_{DD}
54	PD.15 / PWM0_CH5 / TM3 / INT1
55	PA.5 / QSPI0_MISO1 / UART0_nCTS / UART0_TXD / I2C0_SCL / UART5_TXD / BPWM0_CH5 / PWM0_CH0
56	PA.4 / QSPI0_MOSI1 / SPI0_I2SMCLK / UART0_nRTS / UART0_RXD / I2C0_SDA / UART5_RXD / BPWM0_CH4 / PWM0_CH1
57	PA.3 / QSPI0_SS / SPI0_SS / UART4_TXD / I2C0_SMBAL / UART1_TXD / I2C1_SCL / BPWM0_CH3 / PWM0_CH2 / CLKO / PWM1_BRAKE1
58	PA.2 / QSPI0_CLK / SPI0_CLK / UART4_RXD / I2C0_SMBSUS / UART1_RXD / I2C1_SDA / BPWM0_CH2 / PWM0_CH3
59	PA.1 / QSPI0_MISO0 / SPI0_MISO / UART0_TXD / UART1_nCTS / BPWM0_CH1 / PWM0_CH4
60	PA.0 / QSPI0_MOSI0 / SPI0_MOSI / UART0_RXD / UART1_nRTS / BPWM0_CH0 / PWM0_CH5
61	PF.15 / PWM0_BRAKE0 / PWM0_CH1 / TM2 / CLKO / INT4
62	PE.14 / EBI_AD8 / UART2_TXD / UART6_TXD
63	PE.15 / EBI_AD9 / UART2_RXD / UART6_RXD
64	nRESET
65	PF.0 / UART1_TXD / I2C1_SCL / UART0_TXD / BPWM1_CH0 / ICE_DAT
66	PF.1 / UART1_RXD / I2C1_SDA / UART0_RXD / BPWM1_CH1 / ICE_CLK
67	PD.9 / EBI_AD7 / UART2_nCTS / UART7_TXD
68	PD.8 / EBI_AD6 / UART2_nRTS / UART7_RXD
69	PC.5 / EBI_AD5 / QSPI0_MISO1 / UART2_TXD / I2C1_SCL / UART4_TXD / PWM1_CH0
70	PC.4 / EBI_AD4 / QSPI0_MOSI1 / UART2_RXD / I2C1_SDA / UART4_RXD / PWM1_CH1

Pin	M032KIAAE Pin Function
71	PC.3 / EBI_AD3 / QSPI0_SS / UART2_nRTS / I2C0_SMBAL / UART3_TXD / PWM1_CH2
72	PC.2 / EBI_AD2 / QSPI0_CLK / UART2_nCTS / I2C0_SMBSUS / UART3_RXD / PWM1_CH3
73	PC.1 / EBI_AD1 / QSPI0_MISO0 / UART2_TXD / I2C0_SCL / PWM1_CH4 / ACMP0_O / ADC0_ST
74	PC.0 / EBI_AD0 / QSPI0_MOSI0 / UART2_RXD / I2C0_SDA / PWM1_CH5 / ACMP1_O
75	VSS
76	V_{DD}
77	PG.9 / EBI_AD0 / BPWM0_CH5
78	PG.10 / EBI_AD1 / BPWM0_CH4
79	PG.11 / EBI_AD2 / UART7_TXD / BPWM0_CH3
80	PG.12 / EBI_AD3 / UART7_RXD / BPWM0_CH2
81	PG.13 / EBI_AD4 / UART6_TXD / BPWM0_CH1
82	PG.14 / EBI_AD5 / UART6_RXD / BPWM0_CH0
83	PG.15 / CLKO / ADC0_ST
84	PD.7 / UART1_TXD / I2C0_SCL / USCI1_CLK
85	PD.6 / UART1_RXD / I2C0_SDA / USCI1_DAT1
86	PD.5 / I2C1_SCL / USCI1_DAT0
87	PD.4 / USCI0_CTL0 / I2C1_SDA / USCI1_CTL1
88	PD.3 / EBI_AD10 / USCI0_CTL1 / SPI0_SS / UART3_nRTS / USCI1_CTL0 / UART0_TXD
89	PD.2 / EBI_AD11 / USCI0_DAT1 / SPI0_CLK / UART3_nCTS / UART0_RXD
90	PD.1 / EBI_AD12 / USCI0_DAT0 / SPI0_MISO / UART3_TXD
91	PD.0 / EBI_AD13 / USCI0_CLK / SPI0_MOSI / UART3_RXD / TM2
92	PD.13 / EBI_AD10 / SPI0_I2SMCLK
93	USB_VBUS
94	USB_D-
95	USB_D+
96	USB_V_{DD}33_CAP
97	PE.7 / UART5_TXD / PWM0_CH0 / BPWM0_CH5
98	PE.6 / USCI0_CTL0 / UART5_RXD / PWM0_CH1 / BPWM0_CH4
99	PE.5 / EBI_nRD / USCI0_CTL1 / UART6_TXD / UART7_nRTS / PWM0_CH2 / BPWM0_CH3
100	PE.4 / EBI_nWR / USCI0_DAT1 / UART6_RXD / UART7_nCTS / PWM0_CH3 / BPWM0_CH2
101	PE.3 / EBI_MCLK / USCI0_DAT0 / UART6_nRTS / UART7_TXD / PWM0_CH4 / BPWM0_CH1
102	PE.2 / EBI_ALE / USCI0_CLK / UART6_nCTS / UART7_RXD / PWM0_CH5 / BPWM0_CH0
103	VSS
104	V_{DD}

Pin	M032KIAAE Pin Function
105	PE.1 / EBI_AD10 / QSPI0_MISO0 / UART3_TXD / I2C1_SCL / UART4_nCTS
106	PE.0 / EBI_AD11 / QSPI0_MOSI0 / UART3_RXD / I2C1_SDA / UART4_nRTS
107	PH.8 / EBI_AD12 / QSPI0_CLK / UART3_nRTS / UART1_TXD
108	PH.9 / EBI_AD13 / QSPI0_SS / UART3_nCTS / UART1_RXD
109	PH.10 / EBI_AD14 / QSPI0_MISO1 / UART4_TXD / UART0_TXD
110	PH.11 / EBI_AD15 / QSPI0_MOSI1 / UART4_RXD / UART0_RXD / PWM0_CH5
111	PD.14 / EBI_nCS0 / SPI0_I2SMCLK / USCI0_CTL0 / PWM0_CH4
112	VSS
113	LDO_CAP
114	V_{DD}
115	PC.14 / EBI_AD11 / SPI0_I2SMCLK / USCI0_CTL0 / QSPI0_CLK / TM1
116	PB.15 / ADC0_CH15 / EBI_AD12 / SPI0_SS / USCI0_CTL1 / UART0_nCTS / UART3_TXD / PWM1_CH0 / TM0_EXT / PWM0_BRAKE1
117	PB.14 / ADC0_CH14 / EBI_AD13 / SPI0_CLK / USCI0_DAT1 / UART0_nRTS / UART3_RXD / PWM1_CH1 / TM1_EXT / CLKO
118	PB.13 / ADC0_CH13 / ACMP0_P3 / ACMP1_P3 / EBI_AD14 / SPI0_MISO / USCI0_DAT0 / UART0_TXD / UART3_nRTS / PWM1_CH2 / TM2_EXT
119	PB.12 / ADC0_CH12 / ACMP0_P2 / ACMP1_P2 / EBI_AD15 / SPI0_MOSI / USCI0_CLK / UART0_RXD / UART3_nCTS / PWM1_CH3 / TM3_EXT
120	AV_{DD}
121	V_{REF}
122	AVSS
123	PB.11 / ADC0_CH11 / EBI_ADR16 / UART0_nCTS / UART4_TXD / I2C1_SCL / SPI0_I2SMCLK / BPWM1_CH0
124	PB.10 / ADC0_CH10 / EBI_ADR17 / USCI1_CTL0 / UART0_nRTS / UART4_RXD / I2C1_SDA / BPWM1_CH1
125	PB.9 / ADC0_CH9 / EBI_ADR18 / USCI1_CTL1 / UART0_TXD / UART1_nCTS / UART7_TXD / BPWM1_CH2
126	PB.8 / ADC0_CH8 / EBI_ADR19 / USCI1_CLK / UART0_RXD / UART1_nRTS / UART7_RXD / BPWM1_CH3
127	PB.7 / ADC0_CH7 / EBI_nWRL / USCI1_DAT0 / UART1_TXD / EBI_nCS0 / BPWM1_CH4 / PWM1_BRAKE0 / PWM1_CH4 / INT5 / ACMP0_O
128	PB.6 / ADC0_CH6 / EBI_nWRH / USCI1_DAT1 / UART1_RXD / EBI_nCS1 / BPWM1_CH5 / PWM1_BRAKE1 / PWM1_CH5 / INT4 / ACMP1_O

Table 4.1-38 M032KIAAE Multi-function Pin Table

4.2 Pin Mapping

Different part number with the same package might have different function. Please refer to the selection guide in section 3.2, Pin Configuration in section 4.1 or NuTool - PinConfig.

Corresponding Part Number: M031xB, M031xC, M031xD, M031xE, M031xG, M031xI, M032xC, M032xD, M032xE, M032xG, M032xI series.

	M031 Series						M032 Series					
Pin Name	**20 Pin**	**28 Pin**	**32 Pin**	**48 Pin**	**64 Pin**	**128 Pin**	**20 Pin**	**28 Pin**	**32 Pin**	**48 Pin**	**64 Pin**	**128 Pin**
PB.5	8	12	1	1	2	1		12	1	1	2	1
PB.4	9	13	2	2	3	2		13	2	2	3	2
PB.3	10	14	3	3	4	3		14	3	3	4	3
PB.2	11	15	4	4	5	4		15	4	4	5	4
PC.12						5						5
PC.11						6						6
PC.10						7						7
PC.9						8						8
PB.1		16	5	5	6	9		16	5	5	6	9
PB.0		17	6	6	7	10		17	6	6	7	10
V_{SS}						11						11
V_{DD}						12						12
PA.11				7	8	13				7	8	13
PA.10				8	9	14				8	9	14
PA.9				9	10	15				9	10	15
PA.8				10	11	16				10	11	16
PC.13						17						17
PD.12						18						18
PD.11						19						19
PD.10						20						20
PG.2						21						21
PG.3						22						22
PG.4						23						23
PF.11						24						24
PF.10						25						25
PF.9						26						26
PF.8						27						27
PF.7						28						28
PF.6					12	29					12	29

	M031 Series						M032 Series					
Pin Name	**20 Pin**	**28 Pin**	**32 Pin**	**48 Pin**	**64 Pin**	**128 Pin**	**20 Pin**	**28 Pin**	**32 Pin**	**48 Pin**	**64 Pin**	**128 Pin**
PF.14					13	30					13	30
PF.5			7	11	14	31			7	11	14	31
PF.4			8	12	15	32			8	12	15	32
PH.4						33						33
PH.5						34						34
PH.6						35						35
PH.7						36						36
PF.3	12	18	9	13	16	37	11	18	9	13	16	37
PF.2	13	19	10	14	17	38	12	19	10	14	17	38
V_{SS}						39						39
V_{DD}						40						40
PE.8						41						41
PE.9						42						42
PE.10						43						43
PE.11						44						44
PE.12						45						45
PE.13						46						46
PC.8						47						47
PC.7					18	48					18	48
PC.6					19	49					19	49
PA.7				15	20	50				15	20	50
PA.6				16	21	51				16	21	51
V_{SS}					22	52					22	52
V_{DD}					23	53					23	53
PD.15					24	54					24	54
PA.5				17	25	55				17	25	55
PA.4				18	26	56				18	26	56
PA.3	14	20	11	19	27	57	13	20	11	19	27	57
PA.2	15	21	12	20	28	58	14	21	12	20	28	58
PA.1	16	22	13	21	29	59	15	22	13	21	29	59
PA.0	17	23	14	22	30	60	16	23	14	22	30	60
PF.15			15	23	31	61			15	23	31	61
PE.14						62						62
PE.15						63						63

	M031 Series						M032 Series					
Pin Name	**20 Pin**	**28 Pin**	**32 Pin**	**48 Pin**	**64 Pin**	**128 Pin**	**20 Pin**	**28 Pin**	**32 Pin**	**48 Pin**	**64 Pin**	**128 Pin**
nRESET	18	24	16	24	32	64	17	24	16	24	32	64
PF.0	19	25	17	25	33	65	18	25	17	25	33	65
ICE_DAT												
PF.1	20	26	18	26	34	66	19	26	18	26	34	66
ICE_CLK												
PD.9						67						67
PD.8						68						68
PC.5				27	35	69				27	35	69
PC.4				28	36	70				28	36	70
PC.3				29	37	71				29	37	71
PC.2				30	38	72				30	38	72
PC.1		27	19	31	39	73		27	19	31	39	73
PC.0		28	20	32	40	74		28	20	32	40	74
V_{SS}						75						75
V_{DD}						76						76
PG.9						77						77
PG.10						78						78
PG.11						79						79
PG.12						80						80
PG.13						81						81
PG.14						82						82
PG.15						83						83
PD.7						84						84
PD.6						85						85
PD.5						86						86
PD.4						87						87
PD.3					41	88					41	88
PD.2					42	89					42	89
PD.1					43	90					43	90
PD.0					44	91					44	91
PD.13						92						92
NC												
NC												
NC												

	M031 Series						M032 Series					
Pin Name	**20 Pin**	**28 Pin**	**32 Pin**	**48 Pin**	**64 Pin**	**128 Pin**	**20 Pin**	**28 Pin**	**32 Pin**	**48 Pin**	**64 Pin**	**128 Pin**
NC												
PA.12		1	21	33	45	93						
PA.13		2	22	34	46	94						
PA.14		3	23	35	47	95						
PA.15		4	24	36	48	96						
USB_VBUS							20	1	21	33	45	93
USB_D-							1	2	22	34	46	94
USB_D+							2	3	23	35	47	95
USB_V_{DD}33_CAP							3	4	24	36	48	96
PE.7						97						97
PE.6						98						98
PE.5						99						99
PE.4						100						100
PE.3						101						101
PE.2						102						102
V_{SS}						103						103
V_{DD}						104						104
PE.1						105						105
PE.0						106						106
PH.8						107						107
PH.9						108						108
PH.10						109						109
PH.11						110						110
PD.14						111						111
V_{SS}	1	5	25	37	49	112	4	5	25	37	49	112
LDO_CAP	2	6	26	38	50	113	5	6	26	38	50	113
V_{DD}	3	7	27	39	51	114	6	7	27	39	51	114
PC.14				40	52	115				40	52	115
PB.15			28	41	53	116			28	41	53	116
PB.14	4	8	29	42	54	117	7	8	29	42	54	117
PB.13	5	9	30	43	55	118	8	9	30	43	55	118
PB.12	6	10	31	44	56	119	9	10	31	44	56	119
AV_{DD}	7	11	32	45	57	120	10	11	32	45	57	120
V_{REF}					58	121					58	121

	M031 Series						M032 Series					
Pin Name	**20 Pin**	**28 Pin**	**32 Pin**	**48 Pin**	**64 Pin**	**128 Pin**	**20 Pin**	**28 Pin**	**32 Pin**	**48 Pin**	**64 Pin**	**128 Pin**
AV_{SS}				46	59	122				46	59	122
PB.11					60	123					60	123
PB.10					61	124					61	124
PB.9					62	125					62	125
PB.8					63	126					63	126
PB.7				47	64	127				47	64	127
PB.6				48	1	128				48	1	128

4.3 Pin Function Description

Group	Pin Name	Type	Description
ACMP0	ACMP0_N	A	Analog comparator 0 negative input pin.
	ACMP0_O	O	Analog comparator 0 output pin.
	ACMP0_P0	A	Analog comparator 0 positive input 0 pin.
	ACMP0_P1	A	Analog comparator 0 positive input 1 pin.
	ACMP0_P2	A	Analog comparator 0 positive input 2 pin.
	ACMP0_P3	A	Analog comparator 0 positive input 3 pin.
	ACMP0_WLAT	I	Analog comparator 0 window latch input pin
ACMP1	ACMP1_N	A	Analog comparator 1 negative input pin.
	ACMP1_O	O	Analog comparator 1 output pin.
	ACMP1_P0	A	Analog comparator 1 positive input 0 pin.
	ACMP1_P1	A	Analog comparator 1 positive input 1 pin.
	ACMP1_P2	A	Analog comparator 1 positive input 2 pin.
	ACMP1_P3	A	Analog comparator 1 positive input 3 pin.
	ACMP1_WLAT	I	Analog comparator 1 window latch input pin
ADC0	ADC0_CH0	A	ADC0 channel 0 analog input.
	ADC0_CH1	A	ADC0 channel 1 analog input.
	ADC0_CH2	A	ADC0 channel 2 analog input.
	ADC0_CH3	A	ADC0 channel 3 analog input.
	ADC0_CH4	A	ADC0 channel 4 analog input.
	ADC0_CH5	A	ADC0 channel 5 analog input.
	ADC0_CH6	A	ADC0 channel 6 analog input.
	ADC0_CH7	A	ADC0 channel 7 analog input.
	ADC0_CH8	A	ADC0 channel 8 analog input.
	ADC0_CH9	A	ADC0 channel 9 analog input.
	ADC0_CH10	A	ADC0 channel 10 analog input.
	ADC0_CH11	A	ADC0 channel 11 analog input.
	ADC0_CH12	A	ADC0 channel 12 analog input.
	ADC0_CH13	A	ADC0 channel 13 analog input.
	ADC0_CH14	A	ADC0 channel 14 analog input.
	ADC0_CH15	A	ADC0 channel 15 analog input.
	ADC0_ST	I	ADC0 external trigger input pin.
BPWM0	BPWM0_CH0	I/O	BPWM0 channel 0 output/capture input.
	BPWM0_CH1	I/O	BPWM0 channel 1 output/capture input.
	BPWM0_CH2	I/O	BPWM0 channel 2 output/capture input.

Group	Pin Name	Type	Description
	BPWM0_CH3	I/O	BPWM0 channel 3 output/capture input.
	BPWM0_CH4	I/O	BPWM0 channel 4 output/capture input.
	BPWM0_CH5	I/O	BPWM0 channel 5 output/capture input.
BPWM1	BPWM1_CH0	I/O	BPWM1 channel 0 output/capture input.
	BPWM1_CH1	I/O	BPWM1 channel 1 output/capture input.
	BPWM1_CH2	I/O	BPWM1 channel 2 output/capture input.
	BPWM1_CH3	I/O	BPWM1 channel 3 output/capture input.
	BPWM1_CH4	I/O	BPWM1 channel 4 output/capture input.
	BPWM1_CH5	I/O	BPWM1 channel 5 output/capture input.
CLKO	CLKO	O	Clock Out
EBI	EBI_AD0	I/O	EBI address/data bus bit 0.
	EBI_AD1	I/O	EBI address/data bus bit 1.
	EBI_AD2	I/O	EBI address/data bus bit 2.
	EBI_AD3	I/O	EBI address/data bus bit 3.
	EBI_AD4	I/O	EBI address/data bus bit 4.
	EBI_AD5	I/O	EBI address/data bus bit 5.
	EBI_AD6	I/O	EBI address/data bus bit 6.
	EBI_AD7	I/O	EBI address/data bus bit 7.
	EBI_AD8	I/O	EBI address/data bus bit 8.
	EBI_AD9	I/O	EBI address/data bus bit 9.
	EBI_AD10	I/O	EBI address/data bus bit 10.
	EBI_AD11	I/O	EBI address/data bus bit 11.
	EBI_AD12	I/O	EBI address/data bus bit 12.
	EBI_AD13	I/O	EBI address/data bus bit 13.
	EBI_AD14	I/O	EBI address/data bus bit 14.
	EBI_AD15	I/O	EBI address/data bus bit 15.
	EBI_ADR0	O	EBI address bus bit 0.
	EBI_ADR1	O	EBI address bus bit 1.
	EBI_ADR2	O	EBI address bus bit 2.
	EBI_ADR3	O	EBI address bus bit 3.
	EBI_ADR4	O	EBI address bus bit 4.
	EBI_ADR5	O	EBI address bus bit 5.
	EBI_ADR6	O	EBI address bus bit 6.
	EBI_ADR7	O	EBI address bus bit 7.
	EBI_ADR8	O	EBI address bus bit 8.
	EBI_ADR9	O	EBI address bus bit 9.

Group	Pin Name	Type	Description
	EBI_ADR10	O	EBI address bus bit 10.
	EBI_ADR11	O	EBI address bus bit 11.
	EBI_ADR12	O	EBI address bus bit 12.
	EBI_ADR13	O	EBI address bus bit 13.
	EBI_ADR14	O	EBI address bus bit 14.
	EBI_ADR15	O	EBI address bus bit 15.
	EBI_ADR16	O	EBI address bus bit 16.
	EBI_ADR17	O	EBI address bus bit 17.
	EBI_ADR18	O	EBI address bus bit 18.
	EBI_ADR19	O	EBI address bus bit 19.
	EBI_ALE	O	EBI address latch enable output pin.
	EBI_MCLK	O	EBI external clock output pin.
	EBI_nCS0	O	EBI chip select 0 output pin.
	EBI_nCS1	O	EBI chip select 1 output pin.
	EBI_nRD	O	EBI read enable output pin.
	EBI_nWR	O	EBI write enable output pin.
	EBI_nWRH	O	EBI high byte write enable output pin
	EBI_nWRL	O	EBI low byte write enable output pin.
GPIO	PA.x~PH.x	I/O	General purpose digital I/O pin.
I2C0	I2C0_SCL	I/O	I2C0 clock pin.
	I2C0_SDA	I/O	I2C0 data input/output pin.
	I2C0_SMBAL	O	I2C0 SMBus SMBALTER pin
	I2C0_SMBSUS	O	I2C0 SMBus SMBSUS pin (PMBus CONTROL pin)
I2C1	I2C1_SCL	I/O	I2C1 clock pin.
	I2C1_SDA	I/O	I2C1 data input/output pin.
ICE	ICE_CLK	I	Serial wired debugger clock pin **Note:** It is recommended to use 100 kΩ pull-up resistor on ICE_CLK pin
	ICE_DAT	I/O	Serial wired debugger data pin **Note:** It is recommended to use 100 kΩ pull-up resistor on ICE_DAT pin
	nRESET	I	External reset input: active LOW, with an internal pull-up. Set this pin low reset to initial state. **Note:** It is recommended to use 10 kΩ pull-up resistor and 10 uF capacitor on nRESET pin.
INT0	INT0	I	External interrupt 0 input pin.
INT1	INT1	I	External interrupt 1 input pin.
INT3	INT3	I	External interrupt 3 input pin.
INT4	INT4	I	External interrupt 4 input pin.
INT5	INT5	I	External interrupt 5 input pin.
PWM0	PWM0_BRAKE0	I	PWM0 Brake 0 input pin.

Group	Pin Name	Type	Description
	PWM0_BRAKE1	I	PWM0 Brake 1 input pin.
	PWM0_CH0	I/O	PWM0 channel 0 output/capture input.
	PWM0_CH1	I/O	PWM0 channel 1 output/capture input.
	PWM0_CH2	I/O	PWM0 channel 2 output/capture input.
	PWM0_CH3	I/O	PWM0 channel 3 output/capture input.
	PWM0_CH4	I/O	PWM0 channel 4 output/capture input.
	PWM0_CH5	I/O	PWM0 channel 5 output/capture input.
PWM1	PWM1_BRAKE0	I	PWM1 Brake 0 input pin.
	PWM1_BRAKE1	I	PWM1 Brake 1 input pin.
	PWM1_CH0	I/O	PWM1 channel 0 output/capture input.
	PWM1_CH1	I/O	PWM1 channel 1 output/capture input.
	PWM1_CH2	I/O	PWM1 channel 2 output/capture input.
	PWM1_CH3	I/O	PWM1 channel 3 output/capture input.
	PWM1_CH4	I/O	PWM1 channel 4 output/capture input.
	PWM1_CH5	I/O	PWM1 channel 5 output/capture input.
Power	V_{DD}	P	Power supply for I/O ports and LDO source for internal PLL and digital circuit.
	V_{SS}	P	Ground pin for digital circuit.
	AV_{DD}	P	Power supply for internal analog circuit.
	AV_{SS}	P	Ground pin for analog circuit.
	V_{REF}	A	ADC reference voltage input. **Note:** This pin needs to be connected with a 1uF capacitor.
	LDO_CAP	A	LDO output pin. **Note:** This pin needs to be connected with a 1uF capacitor.
QSPI0	QSPI0_CLK	I/O	Quad SPI0 serial clock pin.
	QSPI0_MISO0	I/O	Quad SPI0 MISO0 (Master In, Slave Out) pin.
	QSPI0_MISO1	I/O	Quad SPI0 MISO1 (Master In, Slave Out) pin.
	QSPI0_MOSI0	I/O	Quad SPI0 MOSI0 (Master Out, Slave In) pin.
	QSPI0_MOSI1	I/O	Quad SPI0 MOSI1 (Master Out, Slave In) pin.
	QSPI0_SS	I/O	Quad SPI0 slave select pin.
SPI0	SPI0_CLK	I/O	SPI0 serial clock pin.
	SPI0_I2SMCLK	I/O	SPI0 I^2S master clock output pin
	SPI0_MISO	I/O	SPI0 MISO (Master In, Slave Out) pin.
	SPI0_MOSI	I/O	SPI0 MOSI (Master Out, Slave In) pin.
	SPI0_SS	I/O	SPI0 slave select pin.
TM0	TM0	I/O	Timer0 event counter input/toggle output pin.
	TM0_EXT	I/O	Timer0 external capture input/toggle output pin.
TM1	TM1	I/O	Timer1 event counter input/toggle output pin.

Group	Pin Name	Type	Description
	TM1_EXT	I/O	Timer1 external capture input/toggle output pin.
TM2	TM2	I/O	Timer2 event counter input/toggle output pin.
	TM2_EXT	I/O	Timer2 external capture input/toggle output pin.
TM3	TM3	I/O	Timer3 event counter input/toggle output pin.
	TM3_EXT	I/O	Timer3 external capture input/toggle output pin.
UART0	UART0_RXD	I	UART0 data receiver input pin.
	UART0_TXD	O	UART0 data transmitter output pin.
	UART0_nCTS	I	UART0 clear to Send input pin.
	UART0_nRTS	O	UART0 request to Send output pin.
UART1	UART1_RXD	I	UART1 data receiver input pin.
	UART1_TXD	O	UART1 data transmitter output pin.
	UART1_nCTS	I	UART1 clear to Send input pin.
	UART1_nRTS	O	UART1 request to Send output pin.
UART2	UART2_RXD	I	UART2 data receiver input pin.
	UART2_TXD	O	UART2 data transmitter output pin.
	UART2_nCTS	I	UART2 clear to Send input pin.
	UART2_nRTS	O	UART2 request to Send output pin.
UART3	UART3_RXD	I	UART3 data receiver input pin.
	UART3_TXD	O	UART3 data transmitter output pin.
	UART3_nCTS	I	UART3 clear to Send input pin.
	UART3_nRTS	O	UART3 request to Send output pin.
UART4	UART4_RXD	I	UART4 data receiver input pin.
	UART4_TXD	O	UART4 data transmitter output pin.
	UART4_nCTS	I	UART4 clear to Send input pin.
	UART4_nRTS	O	UART4 request to Send output pin.
UART5	UART5_RXD	I	UART5 data receiver input pin.
	UART5_TXD	O	UART5 data transmitter output pin.
	UART5_nCTS	I	UART5 clear to Send input pin.
	UART5_nRTS	O	UART5 request to Send output pin.
UART6	UART6_RXD	I	UART6 data receiver input pin.
	UART6_TXD	O	UART6 data transmitter output pin.
	UART6_nCTS	I	UART6 clear to Send input pin.
	UART6_nRTS	O	UART6 request to Send output pin.
UART7	UART7_RXD	I	UART7 data receiver input pin.
	UART7_TXD	O	UART7 data transmitter output pin.
	UART7_nCTS	I	UART7 clear to Send input pin.

Group	Pin Name	Type	Description
	UART7_nRTS	O	UART7 request to Send output pin.
USB	USB_VBUS	P	Power supply from USB host or HUB.
	USB_D-	A	USB differential signal D-.
	USB_D+	A	USB differential signal D+.
	USB_V_{DD}33_CAP	A	Internal power regulator output 3.3V decoupling pin.
USCI0	USCI0_CLK	I/O	USCI0 clock pin.
	USCI0_CTL0	I/O	USCI0 control 0 pin.
	USCI0_CTL1	I/O	USCI0 control 1 pin.
	USCI0_DAT0	I/O	USCI0 data 0 pin.
	USCI0_DAT1	I/O	USCI0 data 1 pin.
USCI1	USCI1_CLK	I/O	USCI1 clock pin.
	USCI1_CTL0	I/O	USCI1 control 0 pin.
	USCI1_CTL1	I/O	USCI1 control 1 pin.
	USCI1_DAT0	I/O	USCI1 data 0 pin.
	USCI1_DAT1	I/O	USCI1 data 1 pin.
X32	X32_IN	I	External 32.768 kHz crystal input pin.
	X32_OUT	O	External 32.768 kHz crystal output pin.
XT1	XT1_IN	I	External high speed crystal input pin.
	XT1_OUT	O	External high speed crystal output pin.

5 BLOCK DIAGRAM

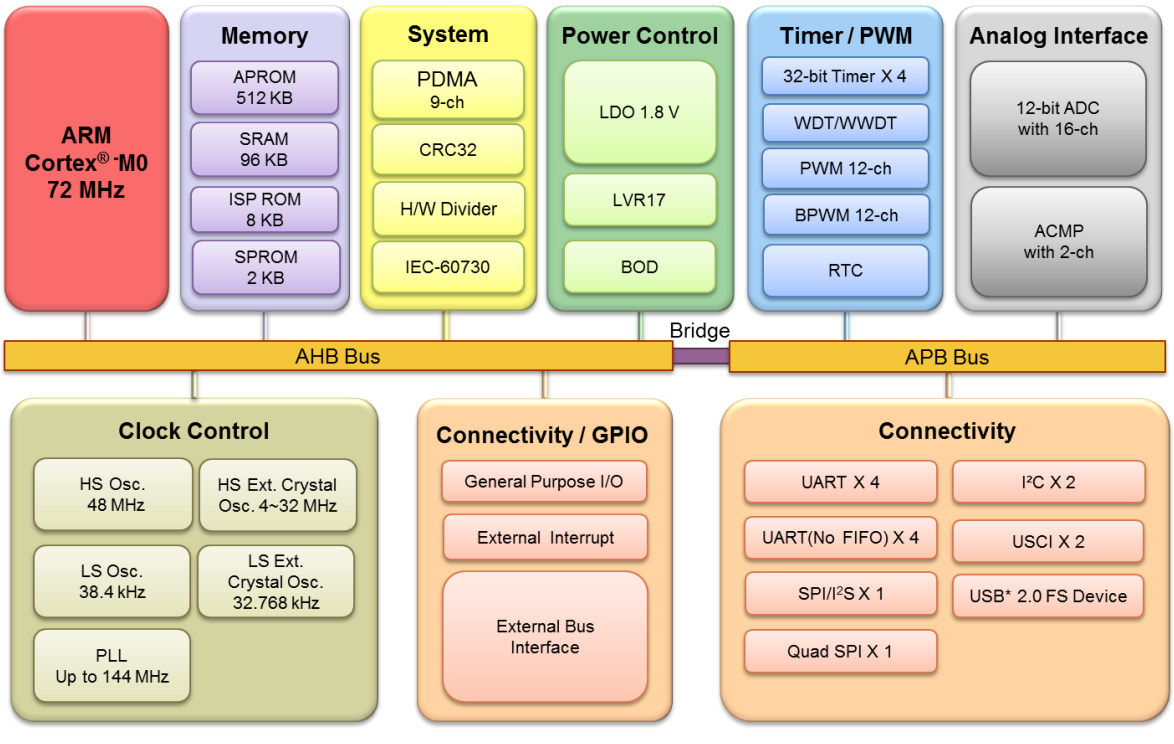


Figure 5-1 NuMicro® M031/M032 Block Diagram

6 FUNCTIONAL DESCRIPTION

6.1 Arm® Cortex®-M0 Core

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

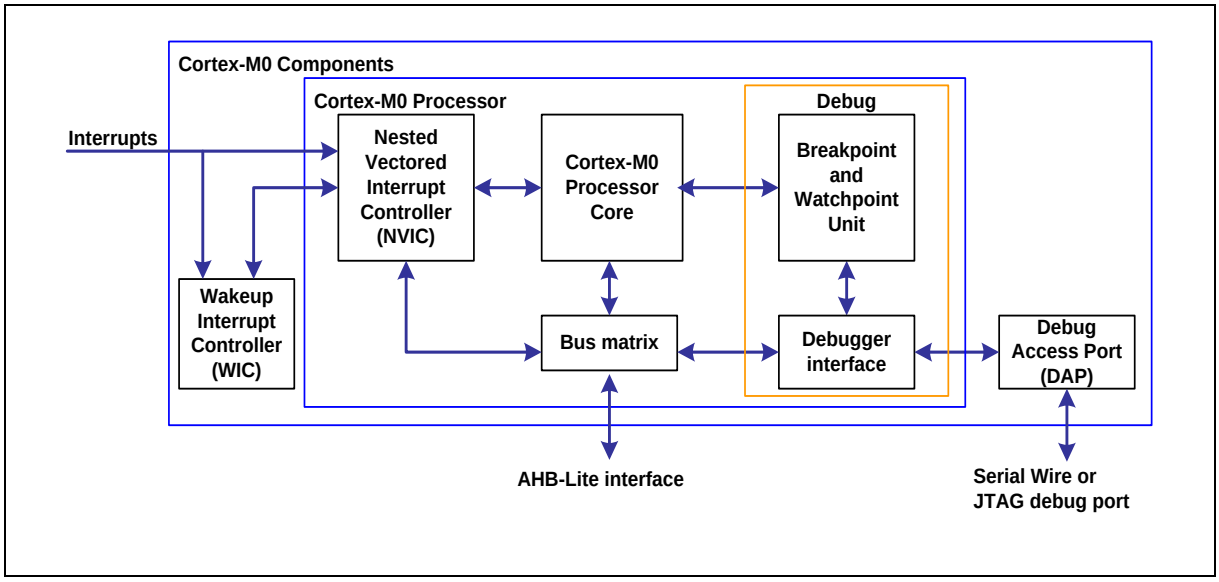


Figure 6-1 Functional Block Diagram

The implemented device provides:

- A low gate count processor:
  - Arm®6-M Thumb® instruction set
  - Thumb-2 technology
  - Arm®6-M compliant 24-bit SysTick timer
  - A 32-bit hardware multiplier
  - System interface supported with little-endian data accesses
  - Ability to have deterministic, fixed-latency, interrupt handling
  - Load/store-multiples and multicycle-multiplies that can be abandoned and restarted to facilitate rapid interrupt handling
  - C Application Binary Interface compliant exception model. This is the Armv6-M, C Application Binary Interface (C-ABI) compliant exception model that enables the use of pure C functions as interrupt handlers
  - Low Power Sleep mode entry using the Wait For Interrupt (WFI), Wait For Event (WFE) instructions, or return from interrupt sleep-on-exit feature
- NVIC:
  - 32 external interrupt inputs, each with four levels of priority

- Dedicated Non-maskable Interrupt (NMI) input
- Supports for both level-sensitive and pulse-sensitive interrupt lines
- Supports Wake-up Interrupt Controller (WIC) and, providing Ultra-low Power Sleep mode

- Debug support:
  - Four hardware breakpoints
  - Two watchpoints
  - Program Counter Sampling Register (PCSR) for non-intrusive code profiling
  - Single step and vector catch capabilities
- Bus interfaces:
  - Single 32-bit AMBA-3 AHB-Lite system interface that provides simple integration to all system peripherals and memory
  - Single 32-bit slave port that supports the DAP (Debug Access Port)

6.2 Clock Controller

6.2.1 Overview

The clock controller generates clocks for the whole chip, including system clocks and all peripheral clocks. The clock controller also implements the power control function with the individually clock ON/OFF control, clock source selection and a clock divider. The chip will not enter Power-down mode until CPU sets the Power-down enable bit PDEN(CLK_PWRCTL[7]) and Cortex®-M0 core executes the WFI instruction. After that, chip enters Power-down mode and wait for wake-up interrupt source triggered to leave Power-down mode. In Power-down mode, the clock controller turns off the 4~32 MHz external high speed crystal (HXT), 48 MHz internal high speed RC oscillator (HIRC) and Programmable PLL output clock frequency (PLLFOUT) to reduce the overall system power consumption.Figure 6.2-1 and Figure 6.2-2 shows the clock generator and the overview of the clock source control.

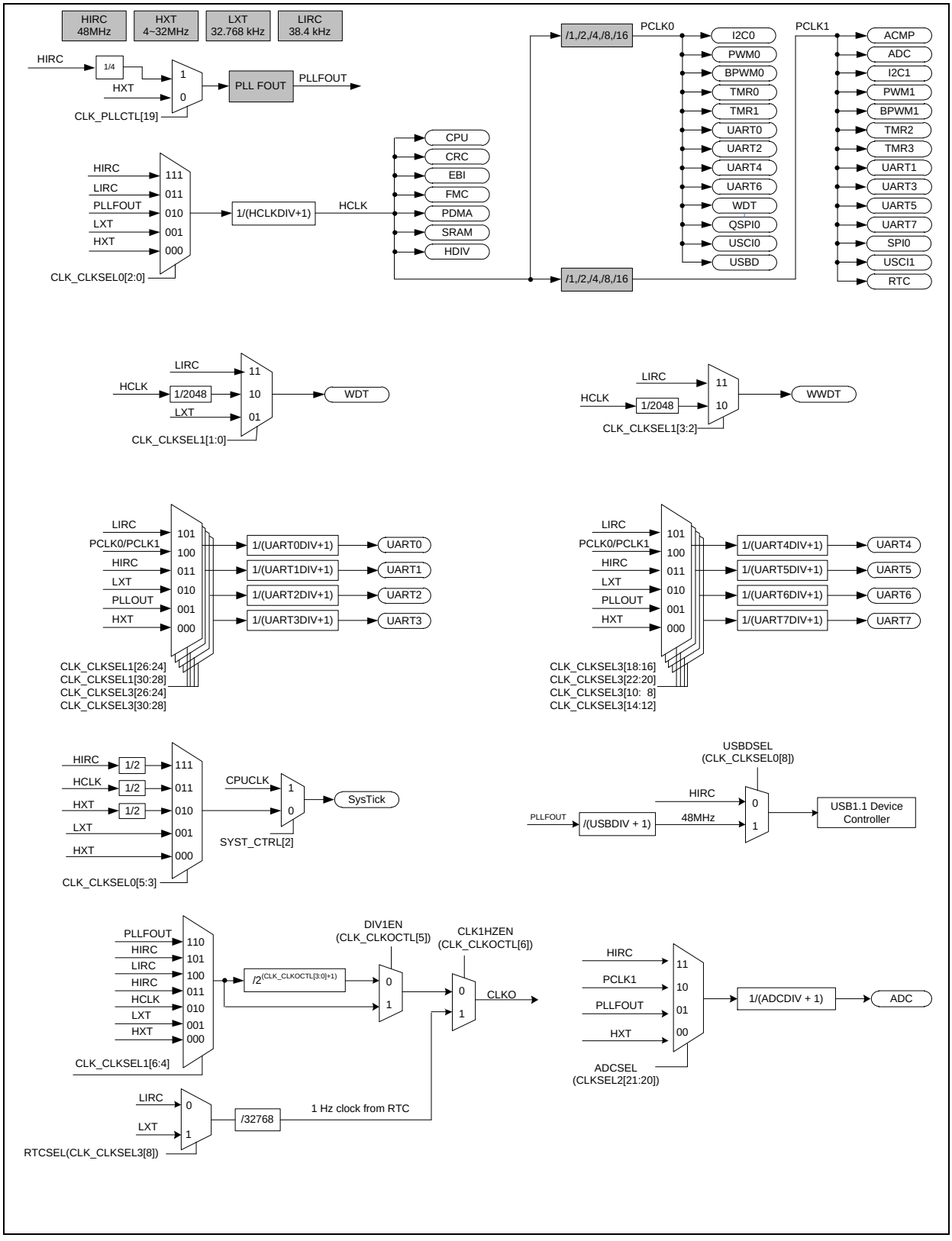


Figure 6.2-1 Clock Generator Global View Diagram (1/2)

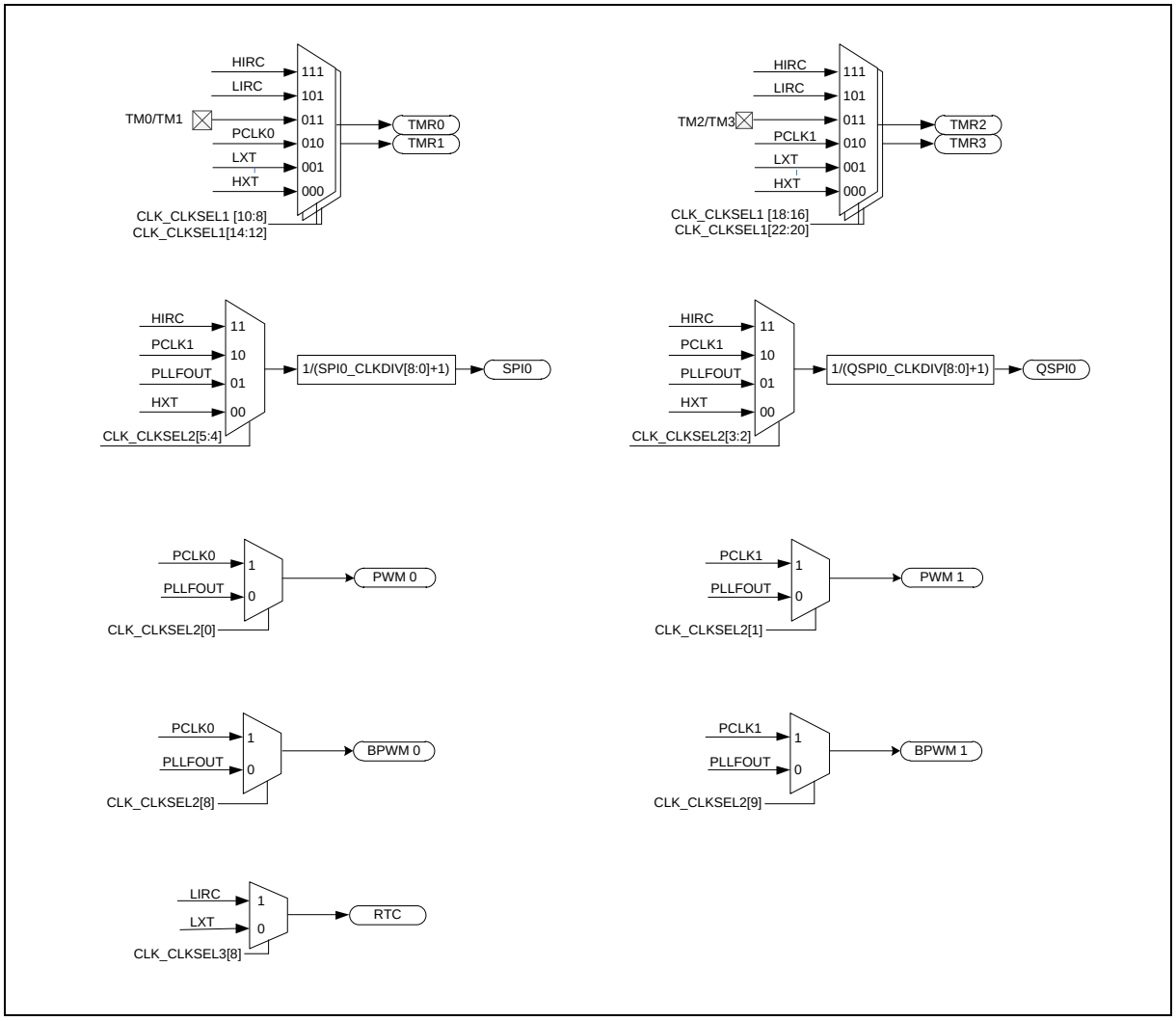


Figure 6.2-2 Clock Generator Global View Diagram (2/2)

6.2.2 Clock Generator

The clock generator consists of 6 clock sources, which are listed below:

- 32.768 kHz external low speed crystal oscillator (LXT)
- 4~32 MHz external high speed crystal oscillator (HXT)
- Programmable PLL output clock frequency (PLLFOUT), PLL source can be selected from external 4~32 MHz external high speed crystal (HXT) or 48 MHz internal high speed oscillator (HIRC/4)
- 48 MHz internal high speed RC oscillator (HIRC)
- 38.4 kHz internal low speed RC oscillator (LIRC)

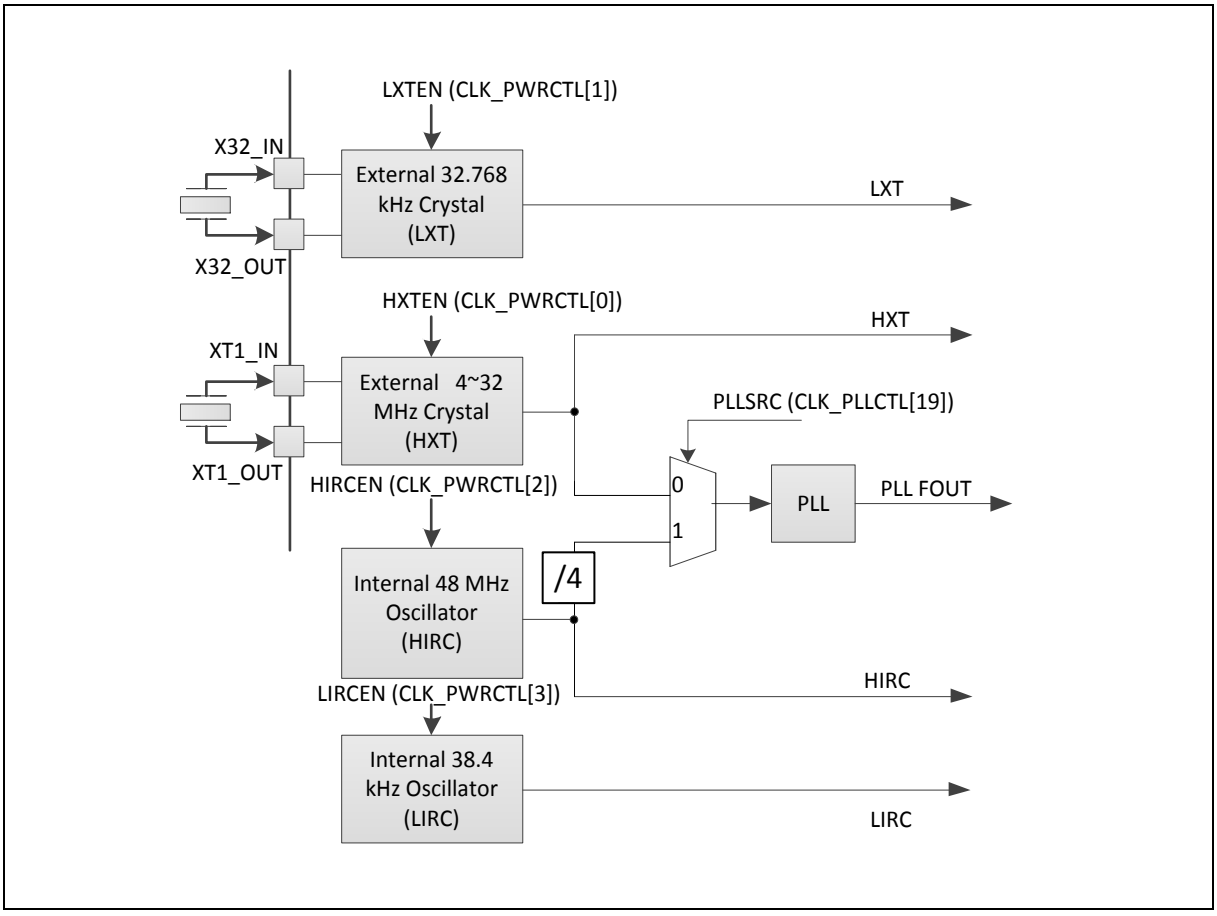


Figure 6.2-3 Clock Generator Block Diagram

6.2.3 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 6.2-4

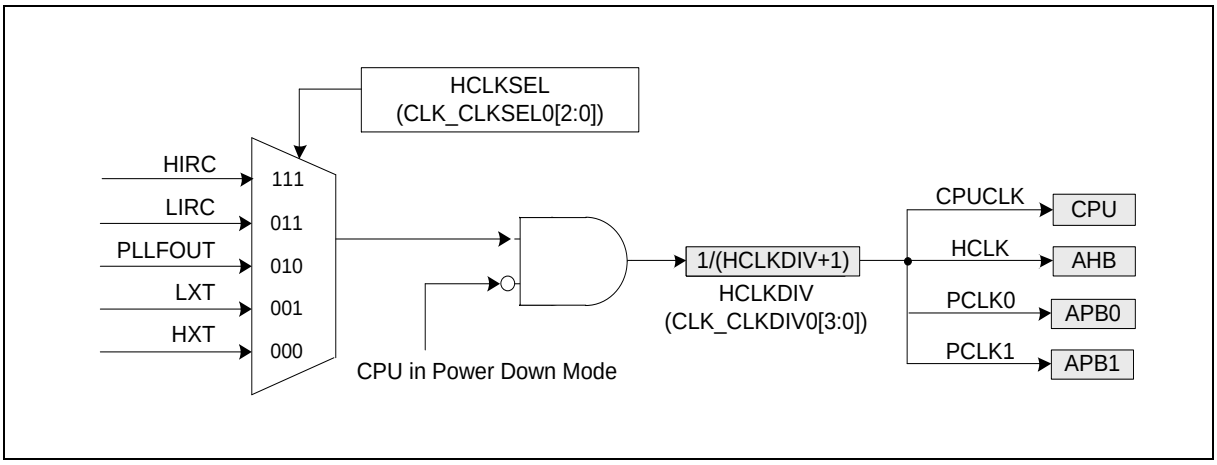


Figure 6.2-4 System Clock Block Diagram

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.

The HXT clock stop detect and system clock switch to HIRC procedure is shown in Figure 6.2-5.

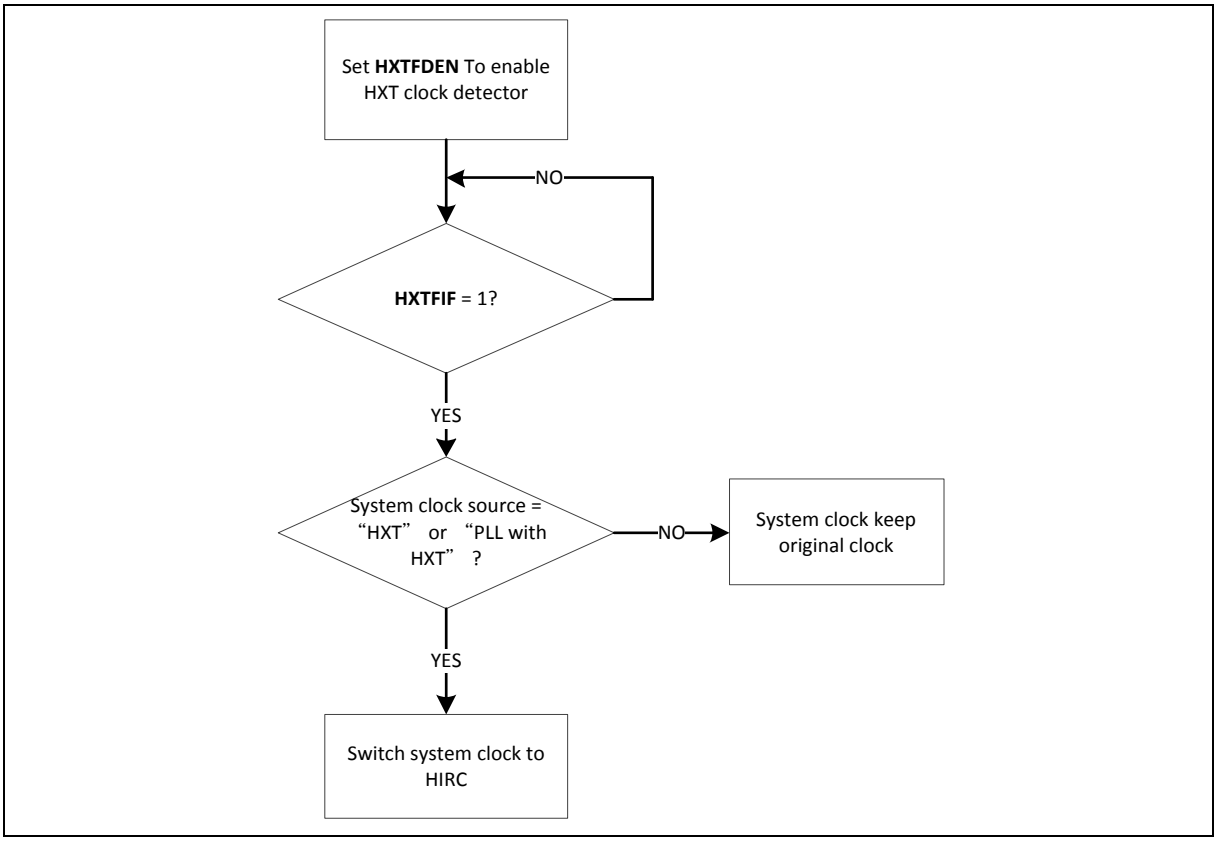


Figure 6.2-5 HXT Stop Protect Procedure

When LXT clock detector is enabled, the system clock will auto switch to LIRC if LXT clock stop being detected when system clock source comes from LXT. If LXT clock stop condition is detected, the LXTFIF (CLK_CLKDSTS[1]) is set to 1 and chip will enter interrupt if LXTFIEN (CLK_CLKDCTL[13]) is set to 1. User can trying to recover LXT by disable LXT and enable LXT again to check if the clock stable bit is set to 1 or not. If LXT clock stable bit is set to 1, it means LXT is recover to oscillate after re-enable action and user can switch system clock to LXT again.

The LXT clock stop detect and system clock switch to LIRC procedure is shown in Figure 6.2-6.

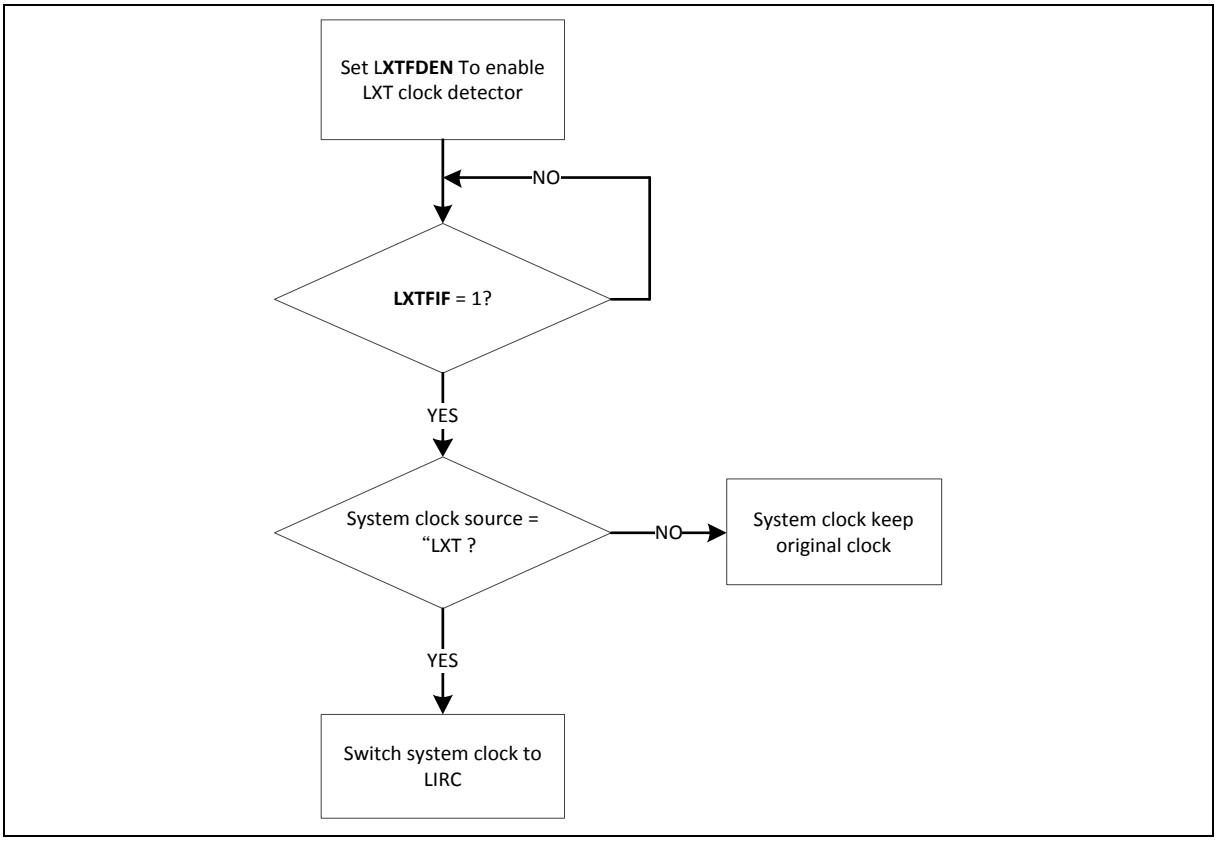


Figure 6.2-6 LXT Stop Protect Procedure

The clock source of SysTick in Cortex®-M0 core can use CPU clock or external clock (SYST_CTRL[2]). If using external clock, the SysTick clock (STCLK) has 5 clock sources. The clock source switch depends on the setting of the register STCLKSEL (CLK_CLKSEL0[5:3]). The block diagram is shown in Figure 6.2-7.

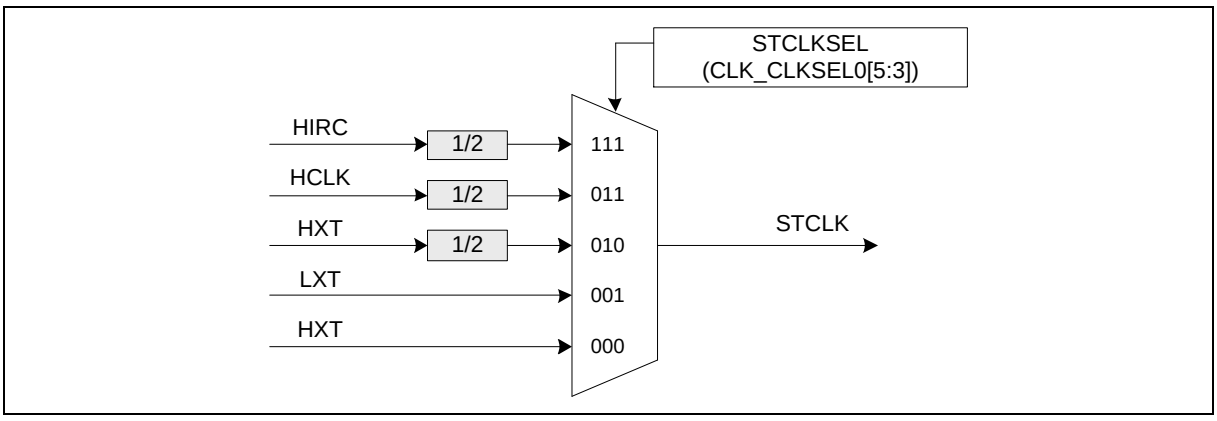


Figure 6.2-7 SysTick Clock Control Block Diagram

6.2.4 Peripherals Clock

The peripherals clock has different clock source switch setting, which depends on the different peripheral. Please refer to the CLK_CLKSELx register description in 6.2.9.

6.2.5 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)

6.2.6 Clock Output

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

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

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

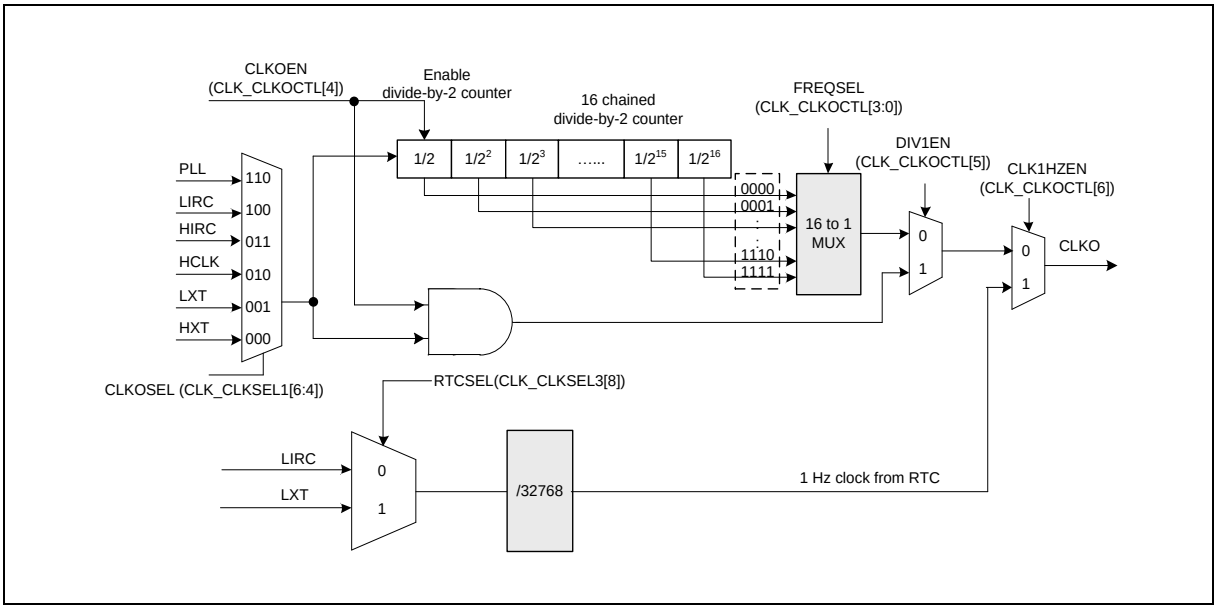


Figure 6.2-8 Clock Output Block Diagram

6.2.7 USB Clock Source

The clock source of USBD is generated from 48 MHz HIRC or programmable PLL output. The generated clocks are shown in Figure 6.2-9.

USBDIV is the clock divider output frequency, the output formula is

(PLLFOUT frequency) / (USBDIV + 1).

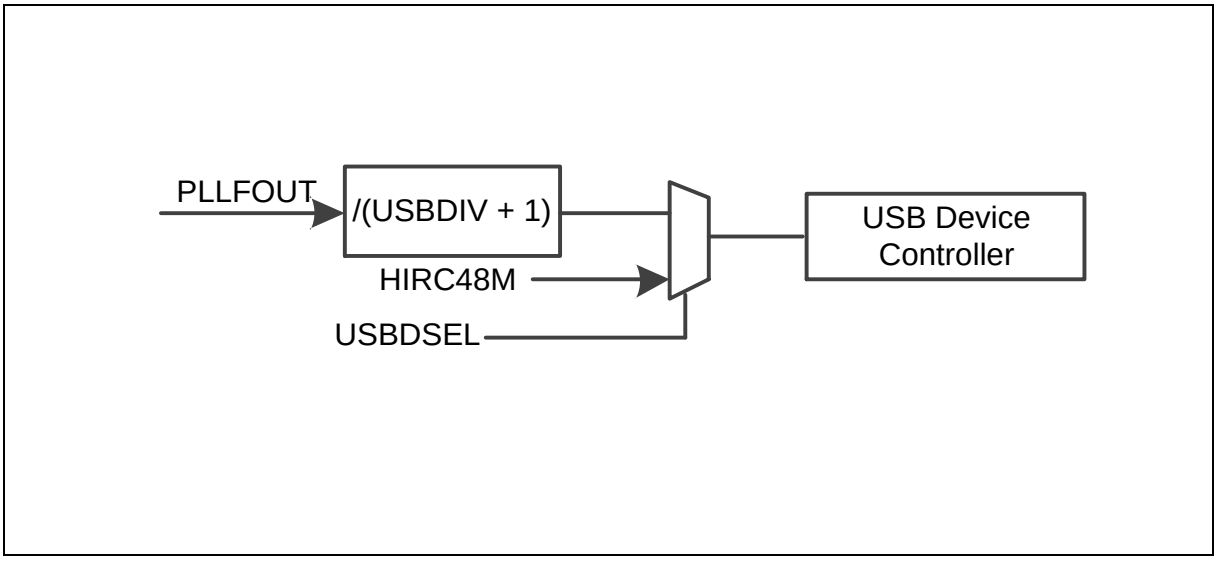


Figure 6.2-9 USBD Clock Source

6.2.8 Register Map

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

Register	Offset	R/W	Description	Reset Value
CLK Base Address: **CLK_BA = 0x4000_0200**				
CLK_PWRCTL	CLK_BA+0x00	R/W	System Power-down Control Register	0x0231_001X
CLK_AHBCLK	CLK_BA+0x04	R/W	AHB Devices Clock Enable Control Register	0x0000_0004
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_003F
CLK_CLKSEL1	CLK_BA+0x14	R/W	Clock Source Select Control Register 1	0x4477_773B
CLK_CLKSEL2	CLK_BA+0x18	R/W	Clock Source Select Control Register 2	0x0020_032B
CLK_CLKSEL3	CLK_BA+0x1C	R/W	Clock Source Select Control Register 3	0x4444_4400
CLK_CLKDIV0	CLK_BA+0x20	R/W	Clock Divider Number Register 0	0x0000_0000
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_C25E
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_LDOCTL	CLK_BA+0x80	R/W	LDO Control Register	0x0000_0000
CLK_HXTFSEL	CLK_BA+0xB4	R/W	HXT Filter Select Control Register	0x0000_0000

6.2.9 Register Description

System Power-down Control Register (CLK_PWRCTL)

Register	Offset	R/W	Description	Reset Value
CLK_PWRCTL	CLK_BA+0x00	R/W	System Power-down Control Register	0x0231_001X

31	30	29	28	27	26	25	24
Reserved					LXTGAIN		LXTSELXT
23	22	21	20	19	18	17	16
Reserved	HXTGAIN			Reserved			
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	
[31:27]	**Reserved**	Reserved.
[26:25]	**LXTGAIN**	**LXT Gain Control Bit (Write Protect)** 00 = LXT Crystal ESR = 35K, CL=12.5pF. 10 = LXT Crystal ESR = 70K, CL=12.5pF. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[24]	**LXTSELXT**	**LXT Mode Selection** 0 = LXT work as crystal mode. PF.4 and PF.5 are configured as external low speed crystal (LXT) pins. 1 = LXT work as external clock mode. PF.5 is configured as external clock input pin. **Note 1:** When LXTSELXT = 1, PF.5 MFP should be setting as GPIO mode. The DC characteristic of X32_IN is the same as GPIO. **Note 2:** This bit is write protected. Refer to the SYS_REGLCTL register.
[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 lower than from 4 MHz. 001 = HXT frequency is from 4 MHz to 8 MHz. 010 = HXT frequency is from 8 MHz to 12 MHz. 011 = HXT frequency is from 12 MHz to 16 MHz. 100 = HXT frequency is from 16 MHz to 24 MHz. 111 = HXT frequency is from 24 MHz to 32 MHz. Others: Reserved **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[19:8]	**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. If user disable LIRC before entering power-down mode, this bit should be set after LIRC disabled 50us. In Power-down mode, the PLL and system clock are disabled, and ignored the clock source selection. The clocks of peripheral are not controlled by Power-down mode, if the peripheral clock source is from LXT or LIRC. 0 = Chip operating normally or chip in idle mode because of WFI command. 1 = Chip enters Power-down mode instant or wait CPU sleep command WFI. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[6]	**PDWKIF**	**Power-down Mode Wake-up Interrupt Status** Set by “Power-down wake-up event”, it indicates that resume from Power-down mode” The flag is set if 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]) 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 external high speed crystal oscillator (HXT), and 512 clock cycles when chip works at internal high speed RC oscillator (HIRC). 0 = Clock cycles delay Disabled. 1 = Clock cycles delay Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[3]	**LIRCEN**	**LIRC Enable Bit (Write Protect)** 0 = Internal low speed RC oscillator (LIRC) Disabled. 1 = 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 = Internal high speed RC oscillator (HIRC) Disabled. 1 = 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 = External low speed crystal (LXT) Disabled. 1 = External low speed crystal (LXT) Enabled. **Note1** : **Note 2:** This bit is write protected. Refer to the SYS_REGLCTL register.
[0]	**HXTEN**	**HXT Enable Bit (Write Protect)** 0 = Eexternal high speed crystal (HXT) Disabled. 1 = External high speed crystal (HXT) Enabled.

		Note1 : reset by power on reset **Note 2:** This bit is write protected. Refer to the SYS_REGLCTL register.

AHB Devices Clock Enable Control Register (CLK_AHBCLK)

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

Register	Offset	R/W	Description	Reset Value
CLK_AHBCLK	CLK_BA+0x04	R/W	AHB Devices Clock Enable Control Register	0x0000_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
CRCCKEN	Reserved		HDIV_EN	EBICKEN	ISPCKEN	PDMACKEN	Reserved

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20]	**Reserved**	
[19:8]	**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]	**HDIV_EN**	**Divider Controller Clock Enable Control** 0 = Divider controller peripheral clock Disabled. 1 = Divider controller peripheral clock Enabled.
[3]	**EBICKEN**	**EBI Controller Clock Enable Bit** 0 = EBI peripheral clock Disabled. 1 = EBI peripheral clock Enabled.
[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			ADCCKEN	USBDCKEN	Reserved		
23	22	21	20	19	18	17	16
UART7CKEN	UART6CKEN	UART5CKEN	UART4CKEN	UART3CKEN	UART2CKEN	UART1CKEN	UART0CKEN
15	14	13	12	11	10	9	8
Reserved		SPI0CKEN	QSPI0CKEN	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]	**ADCCKEN**	**Analog-digital-converter (ADC) Clock Enable Bit** 0 = ADC clock Disabled. 1 = ADC clock Enabled.
[27]	**USBDCKEN**	**USB Device Clock Enable Bit** 0 = USB Device clock Disabled. 1 = USB Device clock Enabled.
[26:24]	**Reserved**	Reserved.
[23]	**UART7CKEN**	**UART7 Clock Enable Bit** 0 = UART7 clock Disabled. 1 = UART7 clock Enabled.
[22]	**UART6CKEN**	**UART6 Clock Enable Bit** 0 = UART6 clock Disabled. 1 = UART6 clock Enabled.
[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.
[15:14]	**Reserved**	Reserved.
[13]	**SPI0CKEN**	**SPI0 Clock Enable Bit** 0 = SPI0 clock Disabled. 1 = SPI0 clock Enabled.
[12]	**QSPI0CKEN**	**QSPI0 Clock Enable Bit** 0 = QSPI0 clock Disabled. 1 = QSPI0 clock Enabled.
[11:10]	**Reserved**	Reserved.
[9]	**I2C1CKEN**	**I2C1 Clock Enable Bit** 0 = I2C1 clock Disabled. 1 = I2C1 clock Enabled.
[8]	**I2C0CKEN**	**I2C0 Clock Enable Bit** 0 = I2C0 clock Disabled. 1 = I2C0 clock Enabled.
[7]	**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**	**RTC Clock Enable Bit** 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. **Note:** This bit is reset by power on reset, Watchdog reset or software chip reset.

APB Devices Clock Enable Control Register 1 (CLK_APBCLK1)

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

Register	Offset	R/W	Description	Reset Value
CLK_APBCLK1	CLK_BA+0x0C	R/W	APB Devices Clock Enable Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				BPWM1CKEN	BPWM0CKEN	PWM1CKEN	PWM0CKEN
15	14	13	12	11	10	9	8
Reserved						USCI1CKEN	USCI0CKEN
7	6	5	4	3	2	1	0
Reserved							

Bits	Description	
[31:20]	**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]	**PWM1CKEN**	**PWM1 Clock Enable Bit** 0 = PWM1 clock Disabled. 1 = PWM1 clock Enabled.
[16]	**PWM0CKEN**	**PWM0 Clock Enable Bit** 0 = PWM0 clock Disabled. 1 = PWM0 clock Enabled.
[15:10]	**Reserved**	Reserved.
[9]	**USCI1CKEN**	**USCI1 Clock Enable Bit** 0 = USCI1 clock Disabled. 1 = USCI1 clock Enabled.
[8]	**USCI0CKEN**	**USCI0 Clock Enable Bit** 0 = USCI0 clock Disabled. 1 = USCI0 clock Enabled.
[7:0]	**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_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							USBDSEL
7	6	5	4	3	2	1	0
Reserved		STCLKSEL			HCLKSEL		

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**USBDSEL**	**USB Device Clock Source Selection (Write Protect)** These bits are protected bit. It means programming this bit needs to write “59h”, “16h”, “88h” to address 0x4000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100. 0 = Clock source from HIRC. 1 = Clock source from PLL divided. **Note:** If PLL is not supported, clock source of selection “1” will be changed to HIRC. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[7:6]	**Reserved**	Reserved.
[5:3]	**STCLKSEL**	**Cortex®-M0 SysTick Clock Source Selection (Write Protect)** If SYST_CTRL[2]=0, SysTick uses listed clock source below. 000 = Clock source from HXT. 001 = Clock source from LXT. 010 = Clock source from HXT/2. 011 = Clock source from HCLK/2. 111 = Clock source from HIRC/2. Other = Reserved. **Note:** If SysTick clock source is not from HCLK (i.e. SYST_CTRL[2] = 0), SysTick clock source must less than or equal to HCLK/2. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note:** If LXT or HXT is not supported, clock source of selection “000”, “001”, or “010” will be stopped. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[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. 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. **Note :** reset by power on reset **Note:** If PLL is not supported, clock source of selection “010” will be changed to HIRC. **Note:** If LXT or HXT is not supported, clock source of selection “000”or “001” will be kept previous clock selection. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.

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	0x4477_773B

31	30	29	28	27	26	25	24
Reserved	UART1SEL			Reserved	UART0SEL		
23	22	21	20	19	18	17	16
Reserved	TMR3SEL			Reserved	TMR2SEL		
15	14	13	12	11	10	9	8
Reserved	TMR1SEL			Reserved	TMR0SEL		
7	6	5	4	3	2	1	0
Reserved	CLKOSEL			WWDTSEL		WDTSEL	

Bits	Description	
[31]	**Reserved**	Reserved.
[30:28]	**UART1SEL**	**UART1 Clock Source Selection** 000 = Clock source from external high speed crystal oscillator (HXT). 001 = Clock source from PLL. 010 = Clock source from external low speed crystal oscillator (LXT). 011 = Clock source from internal high speed RC oscillator (HIRC). 100 = Clock source from PCLK1. 101 = Clock source from internal low speed RC oscillator (LIRC). Other = Reserved. **Note:** If PLL is not supported, clock source of selection “001” will be changed to PCLK1. **Note:** If LXT or HXT is not supported, clock source of selection “000” or “010” will be stopped. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[27]	**Reserved**	Reserved.
[26:24]	**UART0SEL**	**UART0 Clock Source Selection** 000 = Clock source from external high speed crystal oscillator (HXT). 001 = Clock source from PLL. 010 = Clock source from external low speed crystal oscillator (LXT). 011 = Clock source from internal high speed RC oscillator (HIRC). 100 = Clock source from PCLK0. 101 = Clock source from internal low speed RC oscillator (LIRC). Other = Reserved. **Note:** If PLL is not supported, clock source of selection “001” will be changed to PCLK0. **Note:** If LXT or HXT is not supported, clock source of selection “000” or “010” will be stopped.

		Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[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. **Note:** If LXT or HXT is not supported, clock source of selection “000” or “001” will be stopped. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[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. **Note:** If LXT or HXT is not supported, clock source of selection “000” or “001” will be stopped. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[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. **Note:** If LXT or HXT is not supported, clock source of selection “000” or “001” will be stopped. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[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. **Note:** If LXT or HXT is not supported, clock source of selection “000” or “001” will be stopped. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for

		detailed information.
[7]	**Reserved**	Reserved.
[6:4]	**CLKOSEL**	**Clock Divider 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 HCLK. 011 = Clock source from internal high speed RC oscillator (HIRC). 100 = Clock source from internal low speed RC oscillator (LIRC). 101 = Clock source from internal high speed RC oscillator (HIRC). 110 = Clock source from PLL. **Note:** If PLL is not supported, clock source of selection “110” will be changed to HIRC. **Note:** If LXT or HXT is not supported, clock source of selection “000” or “001” will be stopped. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[3:2]	**WWDTSEL**	**Window Watchdog Timer Clock Source Selection (Write Protect)** 10 = Clock source from HCLK/2048. 11 = Clock source from internal low speed RC oscillator (LIRC). Others = 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. 2. Will be forced to 11 when CONFIG0[31], CONFIG0[4], CONFIG0[3] are all ones. **Note:** If LXT is not supported, clock source of selection “01” will be stopped. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.

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	0x0020_032B

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved		ADCSEL		Reserved			
15	14	13	12	11	10	9	8
Reserved						BPWM1SEL	BPWM0SEL
7	6	5	4	3	2	1	0
Reserved		SPI0SEL		QSPI0SEL		PWM1SEL	PWM0SEL

Bits	Description	
[31:22]	**Reserved**	Reserved.
[21:20]	**ADCSEL**	**ADC Clock Source Selection** 00 = Clock source from external high speed crystal oscillator (HXT) clock. 01 = Clock source from PLL. 10 = Clock source from PCLK1. 11 = Clock source from internal high speed RC oscillator (HIRC) clock. **Note:** If PLL is not supported, clock source of selection “01” will be changed to PCLK1. **Note:** If HXT is not supported, clock source of selection “00” will be stopped. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[19:10]	**Reserved**	Reserved.
[9]	**BPWM1SEL**	**BPWM1 Clock Source Selection** The peripheral clock source of BPWM1 is defined by BPWM1SEL. 0 = Clock source from PLL. 1 = Clock source from PCLK1. **Note:** If PLL is not supported, clock source of selection “0” will be changed to PCLK1. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[8]	**BPWM0SEL**	**BPWM0 Clock Source Selection** The peripheral clock source of BPWM0 is defined by BPWM0SEL. 0 = Clock source from PLL. 1 = Clock source from PCLK0. **Note:** If PLL is not supported, clock source of selection “0” will be changed to PCLK0. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[7:6]	**Reserved**	Reserved.

[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). **Note:** If PLL is not supported, clock source of selection "01" will be changed to PCLK1. **Note:** If HXT is not supported, clock source of selection "00" will be stopped. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[3:2]	**QSPI0SEL**	**QSPI0 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). **Note:** If PLL is not supported, clock source of selection "01" will be changed to PCLK0. **Note:** If HXT is not supported, clock source of selection "00" will be stopped. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[1]	**PWM1SEL**	**PWM1 Clock Source Selection** The peripheral clock source of PWM1 is defined by PWM1SEL. 0 = Clock source from PLL. 1 = Clock source from PCLK1. **Note:** If PLL is not supported, clock source of selection "0" will be changed to PCLK1. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[0]	**PWM0SEL**	**PWM0 Clock Source Selection** The peripheral clock source of PWM0 is defined by PWM0SEL. 0 = Clock source from PLL. 1 = Clock source from PCLK0. **Note:** If PLL is not supported, clock source of selection "0" will be changed to PCLK0. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.

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	0x4444_4400

31	30	29	28	27	26	25	24
Reserved	UART3SEL			Reserved	UART2SEL		
23	22	21	20	19	18	17	16
Reserved	UART5SEL			Reserved	UART4SEL		
15	14	13	12	11	10	9	8
Reserved	UART7SEL			Reserved	UART6SEL		
7	6	5	4	3	2	1	0
Reserved							

Bits	Description	
[31]	**Reserved**	Reserved.
[30:28]	**UART3SEL**	**UART3 Clock Source Selection** 000 = Clock source from external high speed crystal oscillator (HXT). 001 = Clock source from PLL. 010 = Clock source from external low speed crystal oscillator (LXT). 011 = Clock source from internal high speed RC oscillator (HIRC). 100 = Clock source from PCLK1. 101 = Clock source from internal low speed RC oscillator (LIRC). Other = Reserved. **Note:** If PLL is not supported, clock source of selection “001” will be changed to PCLK1. **Note:** If LXT or HXT is not supported, clock source of selection “000” or “010” will be stopped. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[27]	**Reserved**	Reserved.
[26:24]	**UART2SEL**	**UART2 Clock Source Selection** 000 = Clock source from external high speed crystal oscillator (HXT). 001 = Clock source from PLL. 010 = Clock source from external low speed crystal oscillator (LXT). 011 = Clock source from internal high speed RC oscillator (HIRC). 100 = Clock source from PCLK0. 101 = Clock source from internal low speed RC oscillator (LIRC). Other = Reserved. **Note:** If PLL is not supported, clock source of selection “001” will be changed to PCLK0. **Note:** If LXT or HXT is not supported, clock source of selection “000” or “010” will be stopped. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for

		detailed information.
[23]	**Reserved**	Reserved.
[22:20]	**UART5SEL**	**UART5 Clock Source Selection** 000 = Clock source from external high speed crystal oscillator (HXT). 001 = Clock source from PLL. 010 = Clock source from external low speed crystal oscillator (LXT). 011 = Clock source from internal high speed RC oscillator (HIRC). 100 = Clock source from PCLK1. 101 = Clock source from internal low speed RC oscillator (LIRC). Other = Reserved. **Note:** If PLL is not supported, clock source of selection “001” will be changed to PCLK1. **Note:** If LXT or HXT is not supported, clock source of selection “000” or “010” will be stopped. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[19]	**Reserved**	Reserved.
[18:16]	**UART4SEL**	**UART4 Clock Source Selection** 000 = Clock source from external high speed crystal oscillator (HXT). 001 = Clock source from PLL. 010 = Clock source from external low speed crystal oscillator (LXT). 011 = Clock source from internal high speed RC oscillator (HIRC). 100 = Clock source from PCLK0. 101 = Clock source from internal low speed RC oscillator (LIRC). Other = Reserved. **Note:** If PLL is not supported, clock source of selection “001” will be changed to PCLK0. **Note:** If LXT or HXT is not supported, clock source of selection “000” or “010” will be stopped. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[15]	**Reserved**	Reserved.
[14:12]	**UART7SEL**	**UART7 Clock Source Selection** 000 = Clock source from external high speed crystal oscillator (HXT). 001 = Clock source from PLL. 010 = Clock source from external low speed crystal oscillator (LXT). 011 = Clock source from internal high speed RC oscillator (HIRC). 100 = Clock source from PCLK1. 101 = Clock source from internal low speed RC oscillator (LIRC). Other = Reserved. **Note:** If PLL is not supported, clock source of selection “001” will be changed to PCLK1. **Note:** If LXT or HXT is not supported, clock source of selection “000” or “010” will be stopped. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[11]	**Reserved**	Reserved.
[10:8]	**UART6SEL**	**UART6 Clock Source Selection** 000 = Clock source from external high speed crystal oscillator (HXT). 001 = Clock source from PLL. 010 = Clock source from external low speed crystal oscillator (LXT). 011 = Clock source from internal high speed RC oscillator (HIRC). 100 = Clock source from PCLK0. 101 = Clock source from internal low speed RC oscillator (LIRC).

		Other = Reserved. **Note:** If PLL is not supported, clock source of selection “001” will be changed to PCLK0. **Note:** If LXT or HXT is not supported, clock source of selection “000” or “010” will be stopped. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[7:0]	**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_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
ADCDIV							
15	14	13	12	11	10	9	8
UART1DIV				UART0DIV			
7	6	5	4	3	2	1	0
USBDIV				HCLKDIV			

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**ADCDIV**	**ADC Clock Divide Number From ADC Clock Source** ADC clock frequency = (ADC clock source frequency) / (ADCDIV + 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]	**USBDIV**	**USB Clock Divide Number From PLL Clock** USB clock frequency = (PLL frequency) / (USBDIV + 1).
[3:0]	**HCLKDIV**	**HCLK Clock Divide Number From HCLK Clock Source** HCLK clock frequency = (HCLK clock source frequency) / (HCLKDIV + 1).

Clock 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
Reserved							
23	22	21	20	19	18	17	16
UART7DIV				UART6DIV			
15	14	13	12	11	10	9	8
UART5DIV				UART4DIV			
7	6	5	4	3	2	1	0
UART3DIV				UART2DIV			

Bits	Description	
[31:4]	**Reserved**	Reserved.
[23:20]	**UART7DIV**	**UART7 Clock Divide Number From UART7 Clock Source** UART7 clock frequency = (UART7 clock source frequency) / (UART7DIV + 1).
[19:16]	**UART6DIV**	**UART6 Clock Divide Number From UART6 Clock Source** UART6 clock frequency = (UART6 clock source frequency) / (UART6DIV + 1).
[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)

The PLL reference clock input is from the 4~32 MHz external high speed crystal oscillator (HXT) clock input or from the 48 MHz internal high speed oscillator (HIRC/4). This register is used to control the PLL output frequency and PLL operation mode.

Programming these bits needs to write “59h”, “16h”, “88h” to address 0x4000_0100 to disable register protection. Refer to the register SYS_REGLCTL at address SYS_BA+0x100.

CLK_PLLCTL only can be modified while PD(CLK_PLLCTL[16]) is set.

Register	Offset	R/W	Description	Reset Value
CLK_PLLCTL	CLK_BA+0x40	R/W	PLL Control Register	0x0005_C25E

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 external high-speed crystal oscillator (HXT). 1 = PLL source clock from 48 MHz internal high-speed oscillator (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 6.2-1 The symbol definition of PLL Output Frequency formula

Clock Status Monitor Register (CLK_STATUS)

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

Register	Offset	R/W	Description	Reset Value
CLK_STATUS	CLK_BA+0x50	R	Clock Status Monitor Register	0x0000_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 = Internal high speed RC oscillator (HIRC) clock is not stable or disabled. 1 = Internal high speed RC oscillator (HIRC) clock is stable and enabled.
[3]	**LIRCSTB**	**LIRC Clock Source Stable Flag (Read Only)** 0 = Internal low speed RC oscillator (LIRC) clock is not stable or disabled. 1 = 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. **Note:** If PLL is not supported, this bit field will be invalid. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[1]	**LXTSTB**	**LXT Clock Source Stable Flag (Read Only)** 0 = External low speed crystal oscillator (LXT) clock is not stable or disabled. 1 = External low speed crystal oscillator (LXT) clock is stabled and enabled. **Note:** If LXT is not supported, this bit field will be invalid. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed

		information.
[0]	**HXTSTB**	**HXT Clock Source Stable Flag (Read Only)** 0 = External high speed crystal oscillator (HXT) clock is not stable or disabled. 1 = External high speed crystal oscillator (HXT) clock is stable and enabled. **Note:** If HXT is not supported, this bit field will be invalid. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.

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	
[31:6]	**Reserved**	Reserved.
[6]	**CLK1HZEN**	**Clock Output 1Hz Enable Bit** 0 = 1 Hz clock output for 32.768 kHz frequency compensation Disabled. 1 = 1 Hz clock output for 32.768 kHz frequency compensation Enabled.
[5]	**DIV1EN**	**Clock Output Divide One Enable Bit** 0 = Clock Output will output clock with source frequency divided by FREQSEL. 1 = Clock Output will output clock with source frequency.
[4]	**CLKOEN**	**Clock Output Enable Bit** 0 = Clock Output function Disabled. 1 = Clock Output function Enabled.
[3:0]	**FREQSEL**	**Clock Output Frequency Selection** The formula of output frequency is $\mathbf{F_{out} = F_{in}/2^{(N+1)}}$. F_{in} is the input clock frequency. F_{out} is the frequency of divider output clock. N is the 4-bit value of FREQSEL[3:0].

Clock Fail Detector Control Register (CLK_CLKDCTL)

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

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

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

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 (Write Protect)** 0 = External high speed crystal oscillator (HXT) clock frequency is normal. 1 = External high speed crystal oscillator (HXT) clock frequency is abnormal. **Note:** Write 1 to clear the bit to 0.
[7:2]	**Reserved**	Reserved.
[1]	**LXTFIF**	**LXT Clock Fail Interrupt Flag (Write Protect)** 0 = External low speed crystal oscillator (LXT) clock is normal. 1 = External low speed crystal oscillator (LXT) stops. **Note:** Write 1 to clear the bit to 0.
[0]	**HXTFIF**	**HXT Clock Fail Interrupt Flag (Write Protect)** 0 = External high speed crystal oscillator (HXT) clock is normal. 1 = External high speed crystal oscillator (HXT) clock stops. **Note:** Write 1 to clear the bit to 0.

Clock Frequency 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	
[31:10]	**Reserved**	Reserved.
[9:0]	**UPERBD**	**HXT Clock Frequency Range Detector Upper Boundary Value** The bits define the maximum value of frequency range detector window. When HXT frequency is 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	
[31:10]	**Reserved**	Reserved.
[9:0]	**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 set to 1.

Frequency out of range will be asserted when HIRC_period*1024> HXT_period*CLK_DUPB or HIRC_period*1024 < HXT_period*CLK_CDLOWB

HXT Filter Select Control Register (CLK_HXTFSEL)

Register	Offset	R/W	Description	Reset Value
CLK_HXTFSEL	CLK_BA+0xB4	R/W	HXT Filter Select Control Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**HXTFSEL**	**HXT Filter Select** 0 = HXT frequency is greater than12 MHz. 1 = HXT frequency is less than or equal to 12 MHz. **Note:** This bit should not be changed during HXT running.

6.3 System Manager

6.3.1 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

6.3.2 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® -M0 core Only by writing 1 to CPURST (SYS_IPRST0[1])
  - nRESET glitch filter time 32us

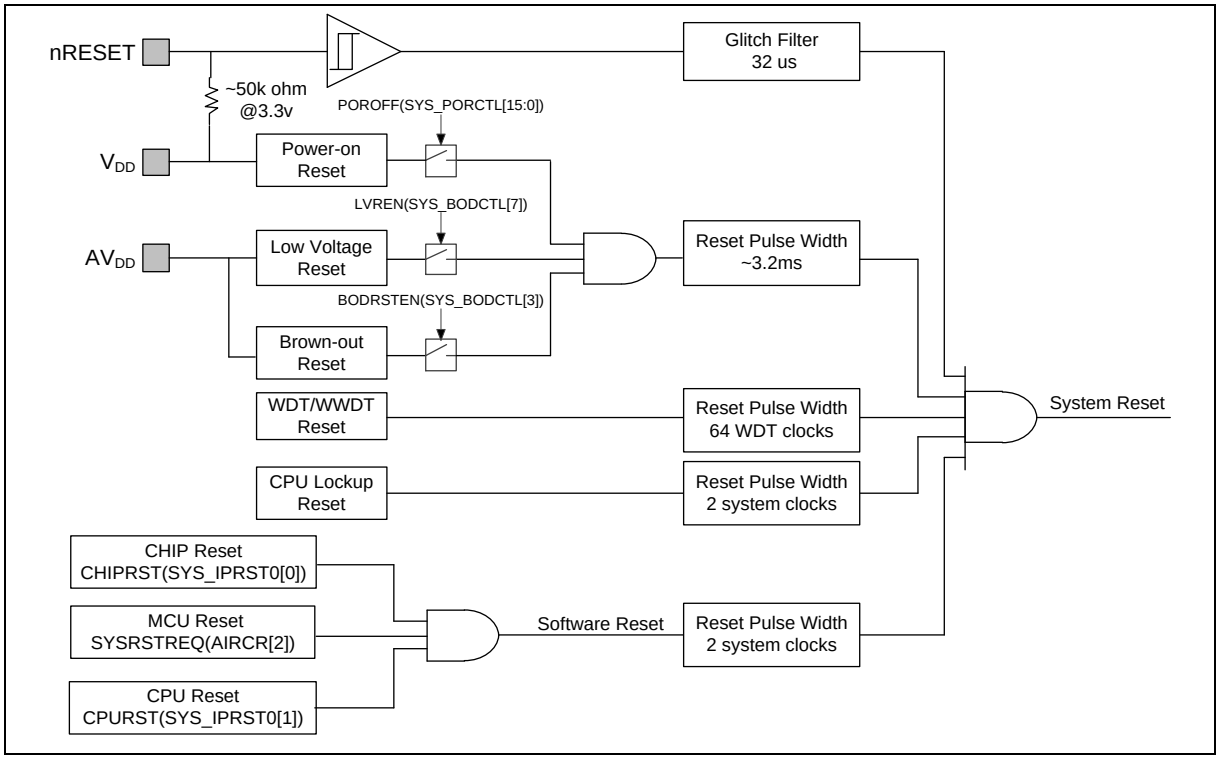


Figure 6.3-1 System Reset Sources

There are a total of 9 reset sources in the NuMicro® family. In general, CPU reset is used to reset Cortex®-M0 only; the other reset sources will reset Cortex®-M0 and all peripherals. However, there are small differences between each reset source and they are listed in Table 6.3-1.

Reset Sources Register	POR	nRESET	WDT	LVR	BOD	Lockup	CHIP	MCU	CPU
SYS_RSTSTS	0x001	Bit 1 = 1	Bit 2 = 1	Bit 3 = 1	Bit 4 = 1	Bit 8 = 1	Bit 0 = 1	Bit 5 = 1	Bit 7 = 1
CHIPRST (SYS_IPRST0[0])	0x0	-	-	-	-	-	-	-	-
BODEN (SYS_BODCTL[0])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-
BODVL (SYS_BODCTL[16])									
BODRSTEN (SYS_BODCTL[3])									
HXTEN (CLK_PWRCTL[0])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	
LXTEN (CLK_PWRCTL[1])	0x0	-	-	-	-	-	-	-	-
LXTSELXT (CLK_PWRCTL[24])	0x0	-	-	-	-	-	-	-	-
LXTGAIN	0x1	-	-	-	-	-	-	-	-

(CLK_PWRCTL[25:26])									
WDTCKEN (CLK_APBCLK0[0])	0x1	-	0x1	-	-	-	0x1	-	-
HCLKSEL (CLK_CLKSEL0[2:0])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-
WDTSEL (CLK_CLKSEL1[1:0])	0x3	0x3	-	-	-	-	-	-	-
HXTSTB (CLK_STATUS[0])	0x0	-	-	-	-	-	-	-	-
LXTSTB (CLK_STATUS[1])	0x0	-	-	-	-	-	-	-	-
PLLSTB (CLK_STATUS[2])	0x0	-	-	-	-	-	-	-	-
HIRCSTB (CLK_STATUS[4])	0x0	-	-	-	-	-	-	-	-
CLKSFAIL (CLK_STATUS[7])	0x0	0x0	-	-	-	-	-	-	-
RSTEN (WDT_CTL[1])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	Reload from CONFIG0	-	-
WDTEN (WDT_CTL[7])									
WDT_CTL except bit 1 and bit 7.	0x0700	0x0700	0x0700	0x0700	0x0700	-	0x0700	-	-
WDT_ALTCTL	0x0000	0x0000	0x0000	0x0000	0x0000	-	0x0000	-	-
WWDT_RLDCNT	0x0000	0x0000	0x0000	0x0000	0x0000	-	0x0000	-	-
WWDT_CTL	0x3F0800	0x3F0800	0x3F0800	0x3F0800	0x3F0800	-	0x3F0800	-	-
WWDT_STATUS	0x0000	0x0000	0x0000	0x0000	0x0000	-	0x0000	-	-
WWDT_CNT	0x3F	0x3F	0x3F	0x3F	0x3F	-	0x3F	-	-
BS (FMC_ISPCTL[1])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	Reload from CONFIG0	-	-
FMC_DFBA	Reload from CONFIG1	Reload from CONFIG1	Reload from CONFIG1	Reload from CONFIG1	Reload from CONFIG1	-	Reload from CONFIG1	-	-
CBS (FMC_ISPSTS[2:1))	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	Reload from CONFIG0	-	-
VECMAP (FMC_ISPSTS[23:9])	Reload base on CONFIG0	Reload base on CONFIG0	Reload base on CONFIG0	Reload base on CONFIG0	Reload base on CONFIG0	-	Reload base on CONFIG0	-	-
Other Peripheral Registers	Reset Value								-

FMC Registers	Reset Value
Note: '-' means that the value of register keeps original setting.	

Table 6.3-1 Reset Value of Registers

6.3.2.1 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. Table 6.3-2 shows the nRESET reset waveform.

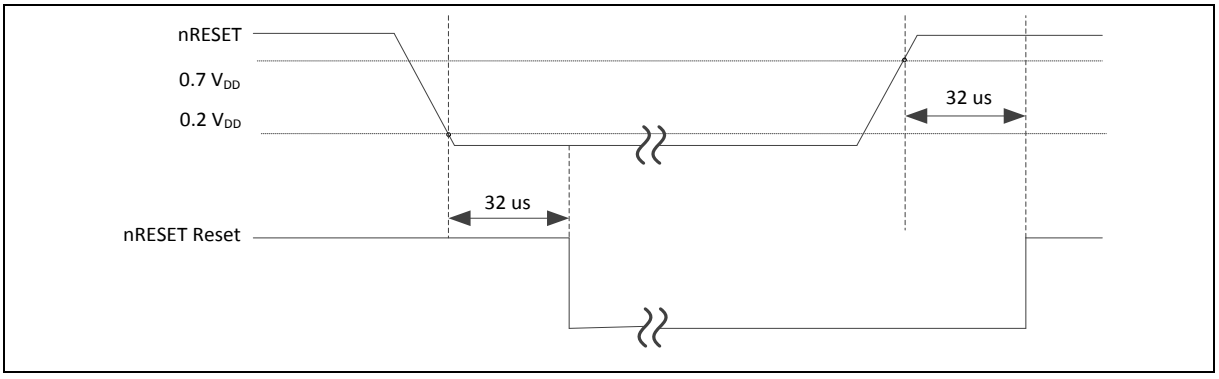


Figure 6.3-2 nRESET Reset Waveform

6.3.2.2 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 6.3-3 shows the power-on reset waveform.

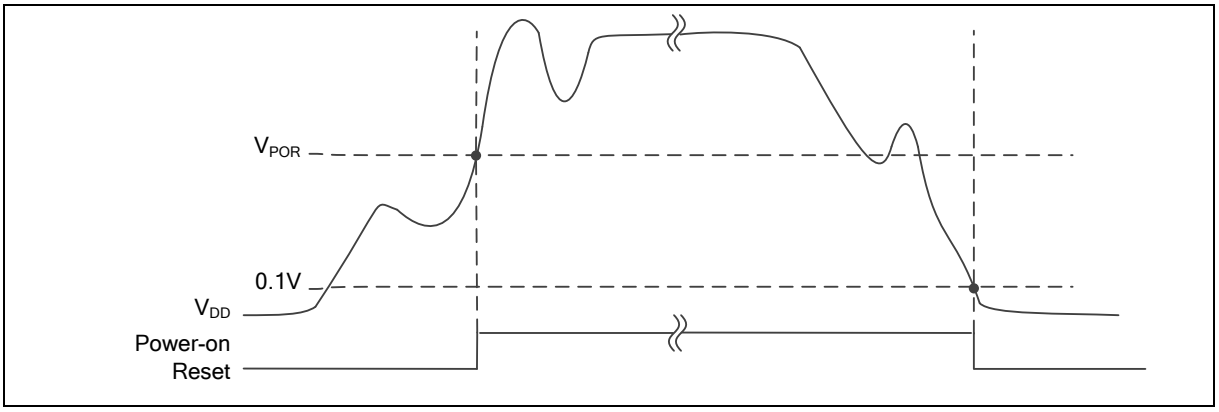


Figure 6.3-3 Power-on Reset (POR) Waveform

6.3.2.3 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 6.3-4 shows the Low Voltage Reset waveform.

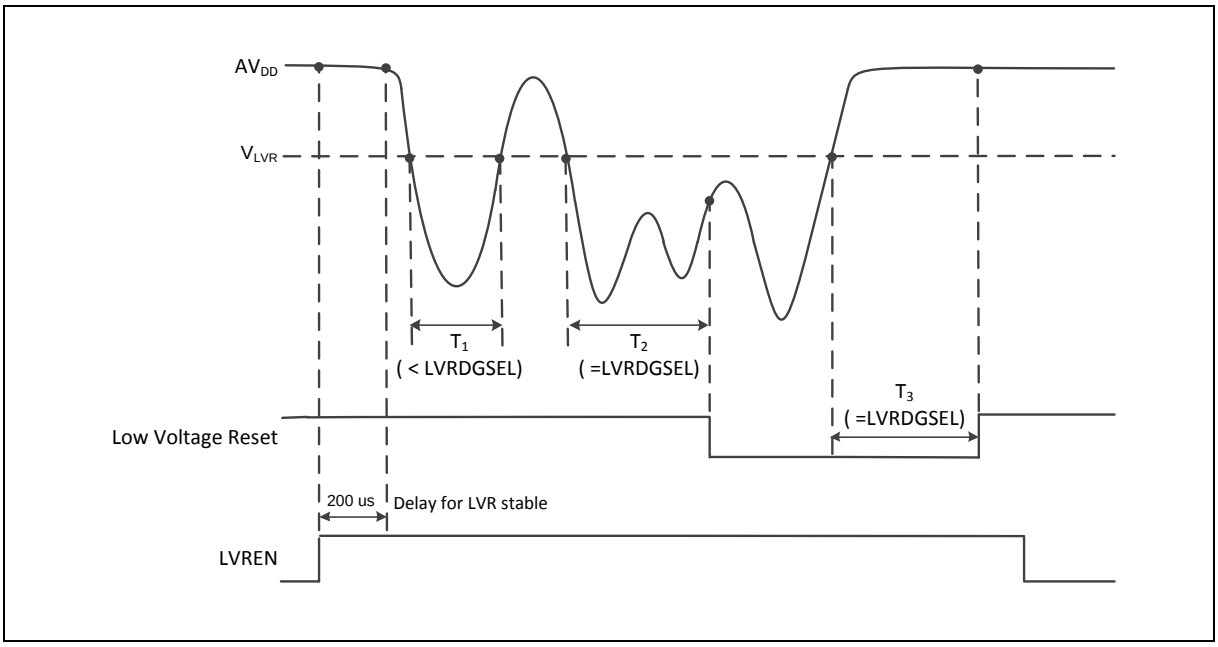


Figure 6.3-4 Low Voltage Reset (LVR) Waveform

6.3.2.4 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[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 [23:21]) and CBORST(CONFIG0[20]) respectively. User can determine the initial BOD setting by setting the CONFIG0 register. Figure 6.3-5 shows the Brown-out Detector waveform.

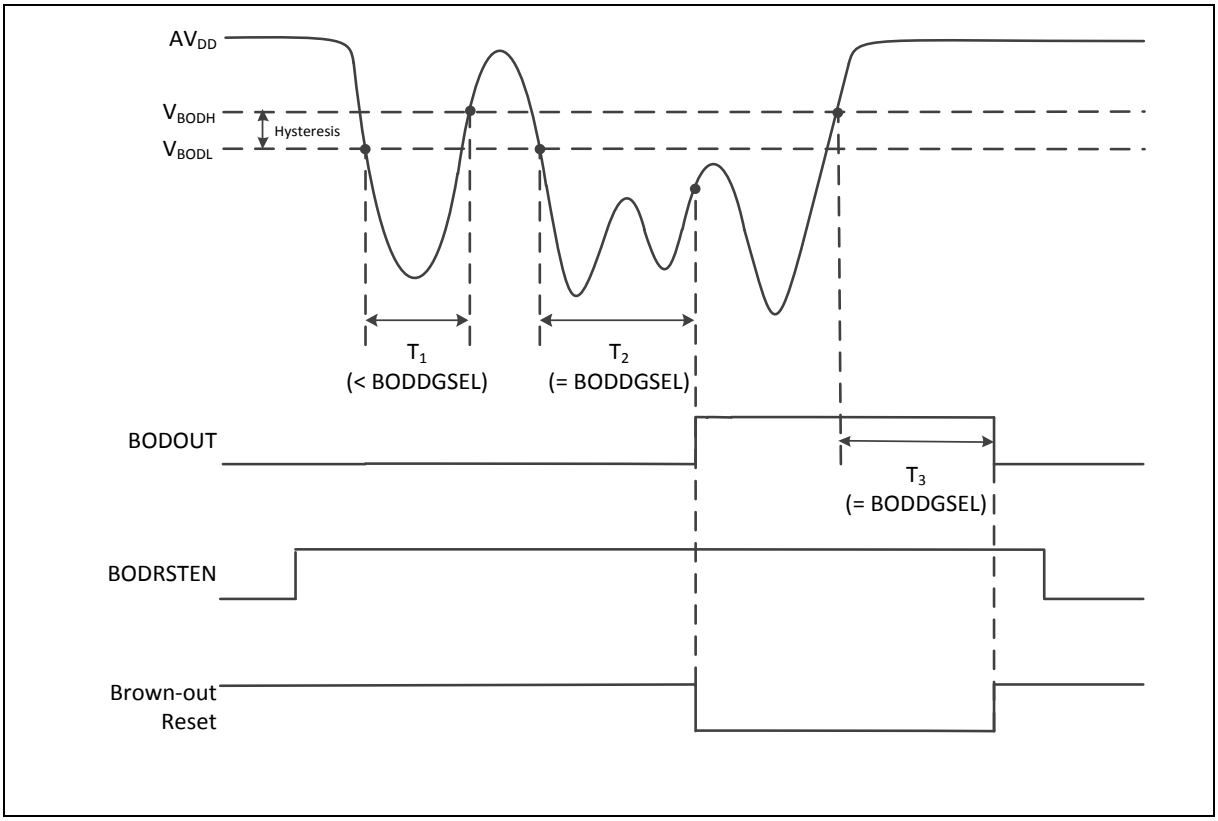


Figure 6.3-5 Brown-out Detector (BOD) Waveform

6.3.2.5 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]).

6.3.2.6 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.

6.3.2.7 CPU Reset, CHIP Reset and MCU Reset

The CPU Reset means only Cortex®-M0 core is reset and all other peripherals remain the same status after CPU reset. User can set the CPURST(SYS_IPRST0[1]) to 1 to assert the CPU Reset signal.

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

6.3.3 System Power Distribution

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

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

The outputs of internal voltage regulators, LDO and V_{DD}, require an external capacitor which should be located close to the corresponding pin. Figure 6.3-6 shows the NuMicro® M031 power distribution.

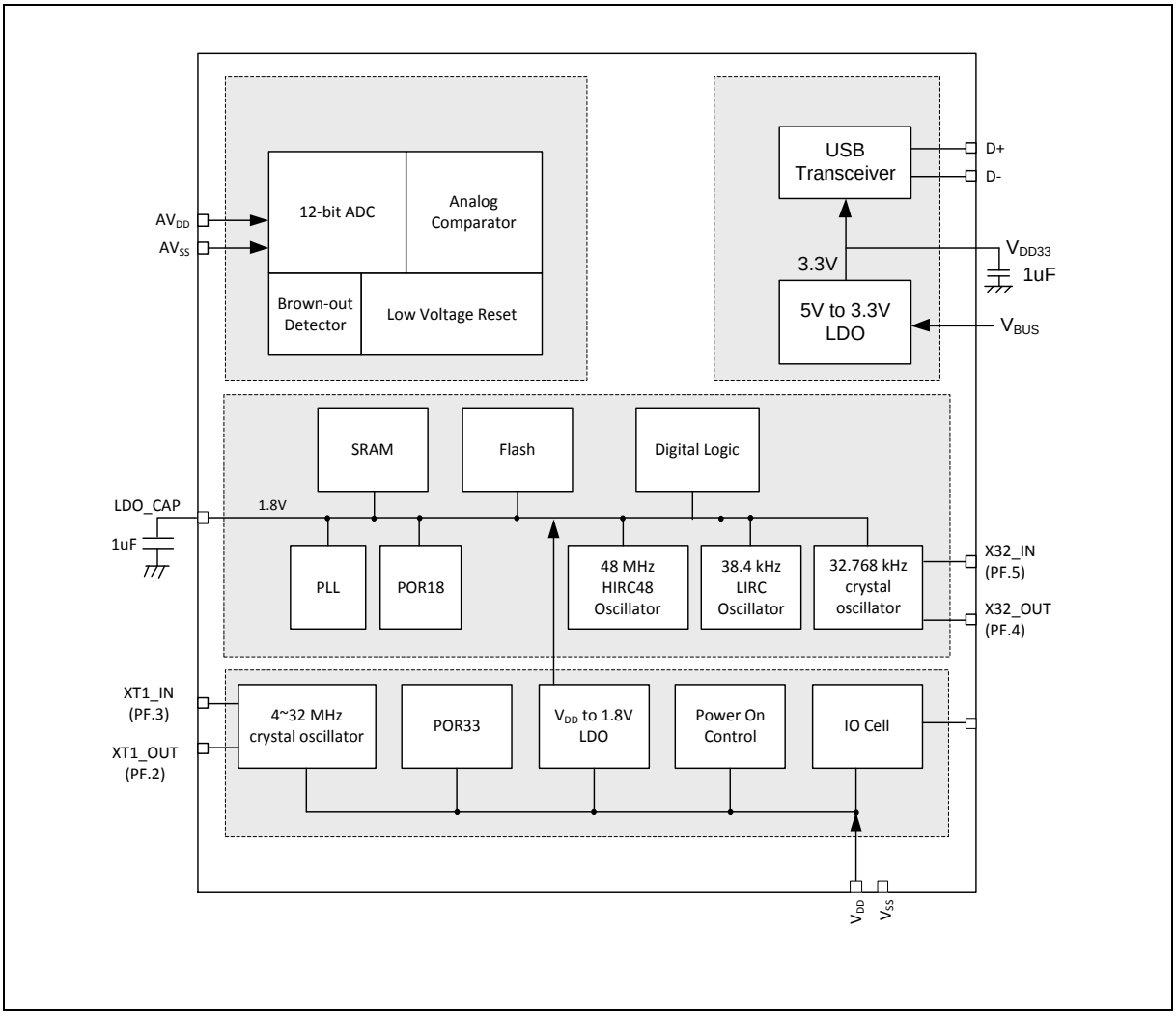


Figure 6.3-6 NuMicro® M031 Power Distribution Diagram

6.3.4 Power Modes and Wake-up Sources

The M031/M032 series has power manager unit to support several operating modes for saving power. Table 6.3-2 lists all power modes in the M031/M032 series.

Mode	CPU Operating Maximum Speed (MHz)	LDO_CAP(V)	Clock Disable
Normal mode	72 MHz at 2.0V-3.6V	1.8	All clocks are disabled by control register.

	48 MHz at 1.8V-3.6V		
Idle mode	CPU enter Sleep mode	1.8	Only CPU clock is disabled.
Power-down mode	CPU enters Power-down mode	1.8	Most clocks are disabled except LIRC/LXT, and only WDT/Timer/UART/RTC peripheral clocks still enable if their clock sources are selected as LIRC/LXT.

Table 6.3-2 Power Mode Table

There are different power mode entry settings and leaving condition for each power mode. Table 6.3-3 shows the entry setting for each power mode. When chip power-on, chip is running in normal mode. User can enter each mode by setting SLEEPDEEP (SCR[2]), PDEN (CLK_PWRCT:[7]) and execute WFI instruction.

Register/Instruction Mode	**SLEEPDEEP (SCR[2])**	**PDEN (CLK_PWRCTL[7])**	**CPU Run WFI Instruction**
Normal mode	0	0	NO
Idle mode (CPU enter Sleep mode)	0	0	YES
Power-down mode (CPU enters Deep Sleep mode)	1	1	YES

Table 6.3-3 Power Mode Difference Table

There are several wake-up sources in Idle mode and Power-down mode. Table 6.3-4 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	WDT, I²C, Timer, UART, BOD, GPIO, EINT, USCI, USBD, ACMP, and RTC
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 6.3-4 Power Mode Difference Table

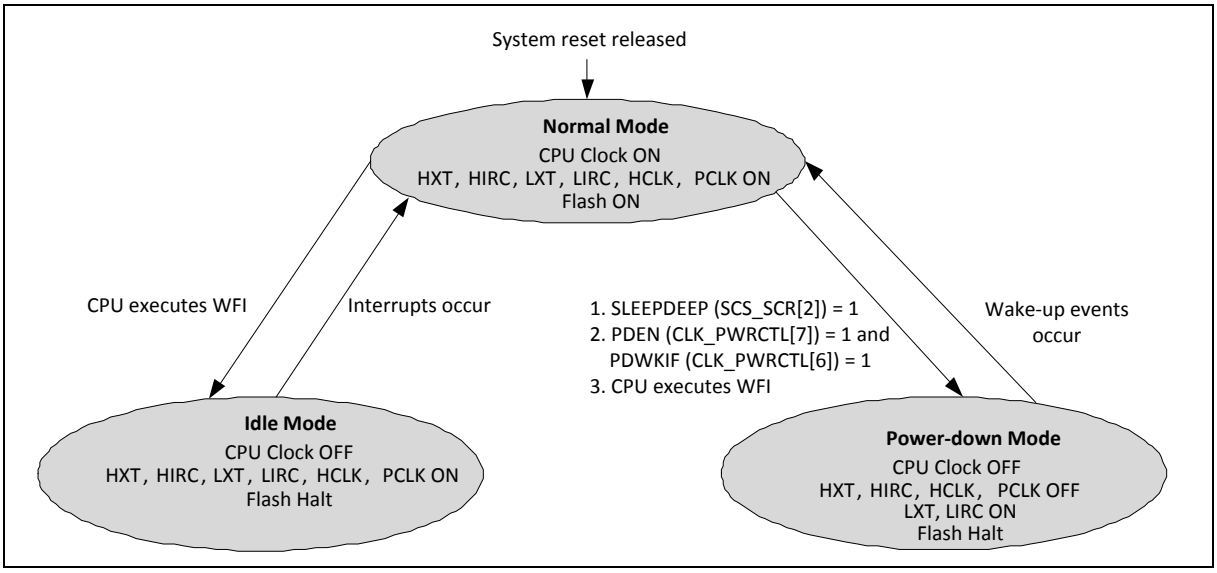


Figure 6.3-7 Power Mode State Machine

1. LXT (32768 Hz XTL) ON or OFF depends on SW setting in normal mode.
2. LIRC (38.4 kHz OSC) ON or OFF depends on SW 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 UART clock source is selected as LXT and LXT is on.
6. If RTC clock source is selected as LIRC/LXT and LIRC/LXT is on.

	Normal Mode	Idle Mode	Power-Down Mode
HXT (4~32 MHz XTL)	ON	ON	Halt
HIRC48 (48 MHz OSC)	ON	ON	Halt
LXT (32768 Hz XTL)	ON	ON	ON/OFF[1]
LIRC (38.4 kHz OSC)	ON	ON	ON/OFF[2]
PLL	ON/OFF	ON/OFF	Halt
LDO	ON	ON	ON
CPU	ON	Halt	Halt
HCLK/PCLK	ON	ON	Halt
SRAM retention	ON	ON	ON
FLASH	ON	ON	Halt
GPIO	ON	ON	Halt
PDMA	ON	ON	Halt
TIMER	ON	ON	ON/OFF[3]
PWM	ON	ON	Halt
BPWM	ON	ON	Halt

WDT	ON	ON	ON/OFF[4]
WWDT	ON	ON	Halt
UART	ON	ON	ON/OFF[5]
USCI	ON	ON	Halt
I^2C	ON	ON	Halt
SPI	ON	ON	Halt
QSPI	ON	ON	Halt
USBD	ON	ON	Halt
ADC	ON	ON	Halt
ACMP	ON	ON	Halt
RTC	ON	ON	ON/OFF[6]

Table 6.3-5 Clocks in Power Modes

Wake-up sources in Power-down mode:

WDT, I²C, Timer, UART, USCI, BOD, GPIO, USBD, ACMP, and RTC.

After chip enters power down, the following wake-up sources can wake chip up to normal mode. Table 6.3-5 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	System Can Enter Power-Down Mode Again Condition*
BOD	Brown-Out Detector Interrupt	After software writes 1 to clear (SYS_BODCTL[4]).
INT	External Interrupt	After software write 1 to clear the Px_INTSRC[n] bit.
GPIO	GPIO Interrupt	After software write 1 to clear the Px_INTSRC[n] bit.
TIMER	Timer Interrupt	After software writes 1 to clear TWKF (TIMERx_INTSTS[1]) and TIF (TIMERx_INTSTS[0]).
WDT	WDT Interrupt	After software writes 1 to clear WKF (WDT_CTL[5]) (Write Protect).
RTC	Alarm Interrupt	After software writes 1 to clear ALMIF (RTC_INTSTS[0]).
	Time Tick Interrupt	After software writes 1 to clear TICKIF (RTC_INTSTS[1]).
UART0/1/4/5	nCTS wake-up	After software writes 1 to clear CTSWKF (UARTx_WKSTS[0]).
	Incoming Data wake-up	After software writes 1 to clear DATWKF (UARTx_WKSTS[1]).
	Received FIFO Threshold Wake-up	After software writes 1 to clear RFRTWKF (UARTx_WKSTS[2]).
	RS-485 AAD Mode Wake-up	After software writes 1 to clear RS485WKF (UARTx_WKSTS[3]).
	Received FIFO Threshold Time-out Wake-up	After software writes 1 to clear TOUTWKF (UARTx_WKSTS[4]).
UART2/3/6/7	nCTS wake-up	After software writes 1 to clear CTSWKF (UARTx_WKSTS[0]).
	Incoming Data wake-up	After software writes 1 to clear DATWKF (UARTx_WKSTS[1]).

USCI UART	CTS Toggle	After software writes 1 to clear WKF (UUART_WKSTS[0]).
	Data Toggle	After software writes 1 to clear WKF (UUART_WKSTS[0]).
USCI I^2C	Data toggle	After software writes 1 to clear WKF (UI2C_WKSTS[0]).
	Address match	After software writes 1 to clear WKAKDONE (UI2C_PROTSTS[16], and then writes 1 to clear WKF (UI2C_WKSTS[0]).
USCI SPI	SS Toggle	After software writes 1 to clear WKF (USPI_WKSTS[0]).
I^2C	Address match	After software writes 1 to clear WKIF (I2C_WKSTS[0]).
USBD	Remote Wake-up	After software writes 1 to clear BUSIF (USBD_INTSTS[0]).
ACMP	Comparator Power-Down Wake-Up Interrupt	After software writes 1 to clear WKIF0 (ACMP_STATUS[8]) and WKIF1 (ACMP_STATUS[9]).

Table 6.3-6 Condition of Entering Power-down Mode Again

6.3.5 System Memory Map

The NuMicro® M031/M032 series provides 4G-byte addressing space. The memory locations assigned to each on-chip controllers are shown inTable 6.3-7. The detailed register definition, memory space, and programming will be described in the following sections for each on-chip peripheral. The M031/M032 series only supports little-endian data format.

Address Space	Token	Controllers
Flash and SRAM Memory Space		
0x0000_0000 – 0x0007_FFFF	FLASH_BA	FLASH Memory Space (512 Kbytes)
0x2000_0000 – 0x2001_7FFF	SRAM0_BA	SRAM Memory Space (96 Kbytes)
0x6000_0000 – 0x6FFF_FFFF	EXTMEM_BA	External Memory Space (256 Mbytes)
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
0x4001_0000 – 0x4001_0FFF	EBI_BA	External Bus Interface Control Registers
0x4001_4000 – 0x4001_7FFF	HDIV_BA	Hardware Divider Register
0x4003_1000 – 0x4003_1FFF	CRC_BA	CRC Generator Registers
APB Controllers Space (0x4000_0000 ~ 0x400F_FFFF)		
0x4004_0000 – 0x4004_0FFF	WDT_BA	Watchdog Timer Control Registers
0x4004_1000 – 0x4004_1FFF	RTC_BA	RTC Control Registers
0x4004_3000 – 0x4004_3FFF	ADC_BA	Analog-Digital-Converter (ADC) Control Registers
0x4004_5000 – 0x4004_5FFF	ACMP01_BA	Analog Comparator 0/ 1 Control Registers
0x4005_0000 – 0x4005_0FFF	TMR01_BA	Timer0/Timer1 Control Registers

0x4005_1000 – 0x4005_1FFF	TMR23_BA	Timer2/Timer3 Control Registers
0x4005_8000 – 0x4005_8FFF	PWM0_BA	PWM0 Control Registers
0x4005_9000 – 0x4005_9FFF	PWM1_BA	PWM1 Control Registers
0x4005_A000 – 0x4005_AFFF	BPWM0_BA	BPWM0 Control Registers
0x4005_B000 – 0x4005_BFFF	BPWM1_BA	BPWM1 Control Registers
0x4006_0000 – 0x4006_0FFF	QSPI0_BA	QSPI0 Control Registers
0x4006_1000 – 0x4006_1FFF	SPI0_BA	SPI0 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
0x4007_6000 – 0x4007_6FFF	UART6_BA	UART6 Control Registers
0x4007_7000 – 0x4007_7FFF	UART7_BA	UART7 Control Registers
0x4008_0000 – 0x4008_0FFF	I2C0_BA	I2C0 Control Registers
0x4008_1000 – 0x4008_1FFF	I2C1_BA	I2C1 Control Registers
0x400C_0000 – 0x400C_0FFF	USBD_BA	USB Device Control Register
0x400D_0000 – 0x400D_0FFF	USCI0_BA	USCI0 Control Registers
0x400D_1000 – 0x400D_1FFF	USCI1_BA	USCI1 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 6.3-7 Address Space Assignments for On-Chip Controllers

6.3.6 SRAM Memory Organization

The M031 supports embedded SRAM with total 16 Kbytes size

- Supports total 16 Kbytes SRAM
- Supports byte / half word / word write
- Supports oversize response error

Table 6.3-9 shows the SRAM organization of M031. The address between 0x2000_4000 to 0x3FFF_FFFF is illegal memory space and chip will enter hardfault if CPU accesses these illegal memory addresses.

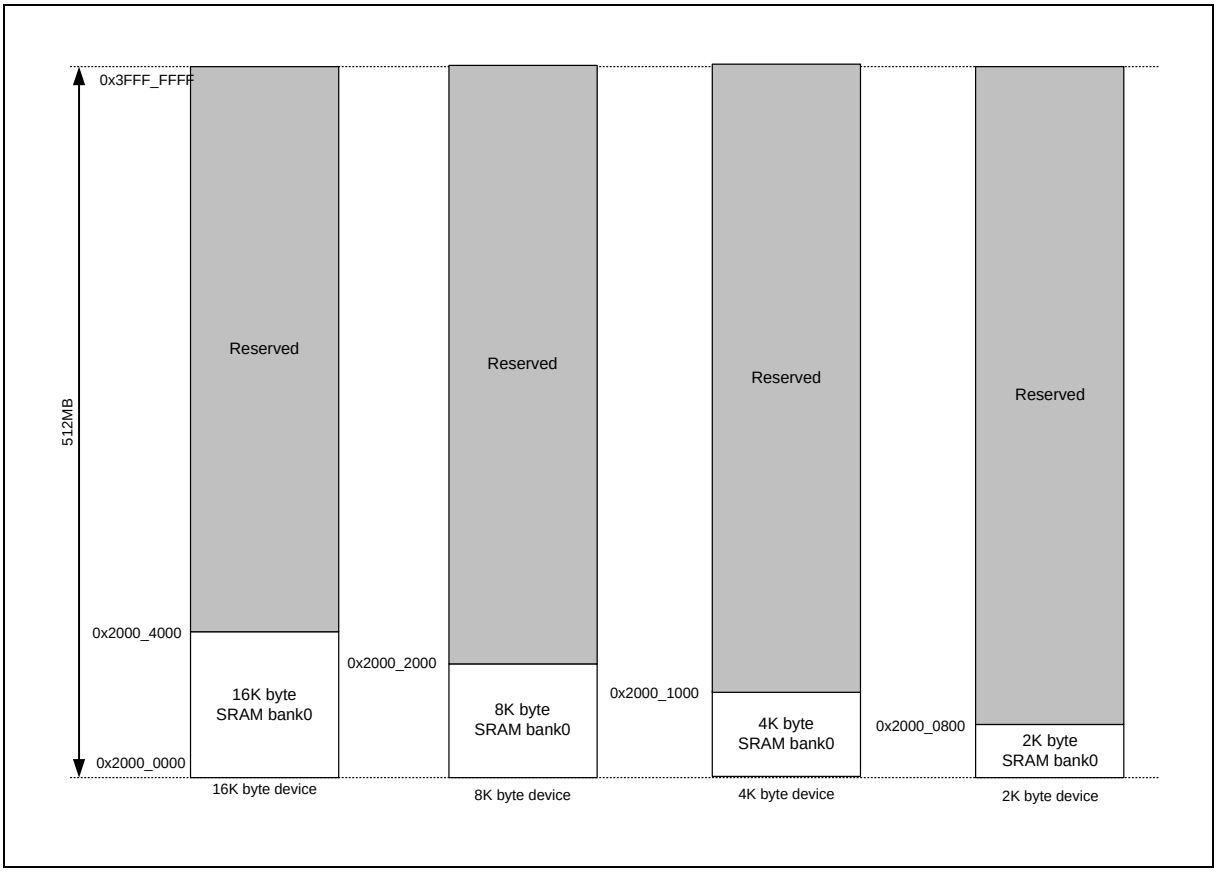


Figure 6.3-8 SRAM Memory Organization

6.3.7 SRAM Memory Organization with parity function

The M031 supports embedded SRAM with total 96 Kbytes size

- Supports total 96 Kbytes SRAM
- Supports parity error check function for SRAM bank0 section 0(32 Kbytes)
- Supports byte / half word / word write
- Supports oversize response error

Table 6.3-9 shows the SRAM organization of M031. The address between 0x2001_8000 to 0x3FFF_FFFF is illegal memory space and chip will enter hardfault if CPU accesses these illegal memory addresses. There are three section in SRAM bank0. The section 0 is addressed to 32 Kbytes with parity function, the section 1 is addressed to 32 Kbytes and the section 2 is addressed to 32 Kbytes. SRAM section 0 has byte parity error check function. When CPU is accessing SRAM section

0, 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.

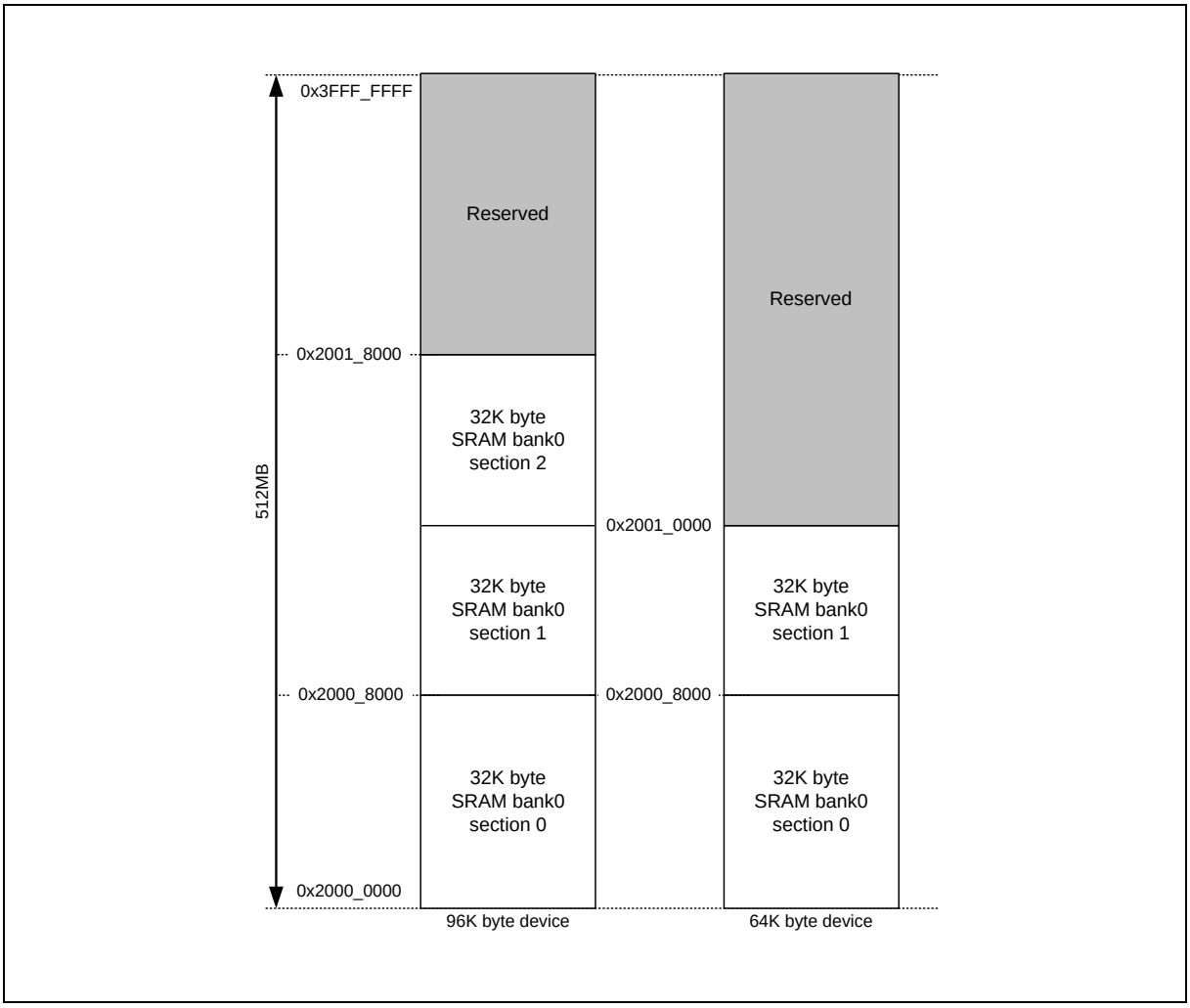


Figure 6.3-10 SRAM Memory Organization

6.3.8 Chip Bus Matrix

The M031/M032 series supports Bus Matrix to manage the access arbitration between masters. The access arbitration use round-robin algorithm as the bus priority.

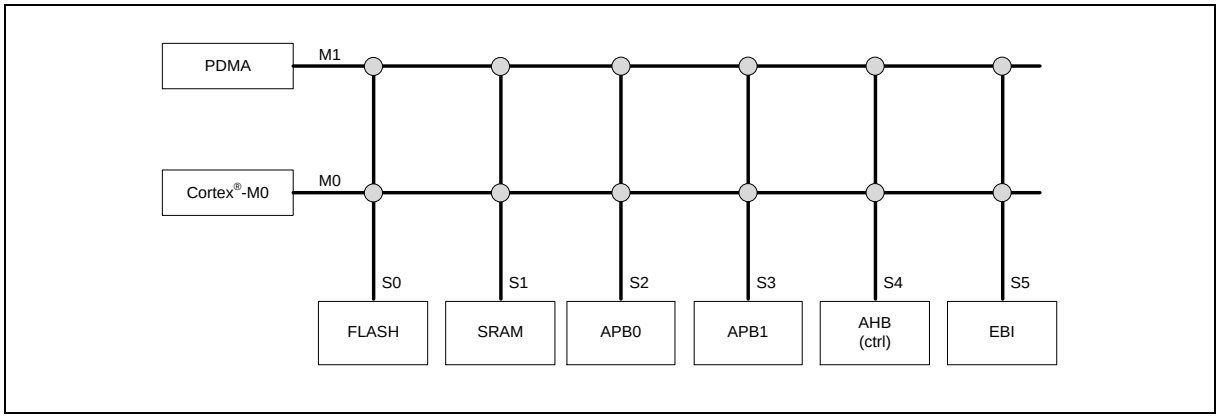


Figure 6.3-9 NuMicro® M031 Bus Matrix Diagram

6.3.9 IRC Auto Trim

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

In HIRC case, the system needs an accurate 48 MHz clock. In such case, if neither using use PLL as the system clock source nor soldering 32.768 kHz crystal in system, user has to set REFCKSEL (SYS_HIRCTRIMCTL [10] reference clock selection) to “1”, set FREQSEL (SYS_HIRCTRIMCTL [1:0] trim frequency selection) to “01”, and the auto-trim function will be enabled. Interrupt status bit FREQLOCK (SYS_HIRCTRIMSTS[0] HIRC frequency lock status) “1” indicates the HIRC output frequency is accurate within 0.25% deviation.

6.3.10 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 to disable register protection. For user to program these protected registers, a register protection disable sequence needs to be followed by a special programming. The register protection disable sequence is writing the data “59h”, “16h” “88h” to the register SYS_REGLCTL address at 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. All proteced control registers are noted “(Write Protect)” and add an note “**Note:** This bit is write protected. Refer to the SYS_REGLCTL register “ in register description field.

Register	Bit	Description
CLK_PWRCTL	[26:25] LXTGAIN	LXT Gain Control Bit (Write Protect)
CLK_PWRCTL	[22:20] HXTGAIN	HXT Gain Control Bit (Write Protect)
CLK_PWRCTL	[7] PDEN	System Power-down Enable (Write Protect)
CLK_PWRCTL	[5] PDWKIEN	Power-down Mode Wake-up Interrupt Enable Bit (Write Protect)
CLK_PWRCTL	[4] PDWKDLY	Enable the Wake-up Delay Counter (Write Protect)
CLK_PWRCTL	[3] LIRCEN	LIRC Enable Bit (Write Protect)
CLK_PWRCTL	[2] HIRCEN	HIRC Enable Bit (Write Protect)
CLK_PWRCTL	[1] LXTEN	LXT Enable Bit (Write Protect)
CLK_PWRCTL	[0] HXTEN	HXT Enable Bit (Write Protect)
CLK_APBCLK0	[0] WDTCKEN	Watchdog Timer Clock Enable Bit (Write Protect)
CLK_CLKSEL0	[8] USBDSEL	USB Device Clock Source Selection (Write Protect)
CLK_CLKSEL0	[5:3] STCLKSEL	Cortex®-M0 SysTick Clock Source Selection (Write Protect)
CLK_CLKSEL0	[2:0] HCLKSEL	HCLK Clock Source Selection (Write Protect)
CLK_CLKSEL1	[3:2] WWDTSEL	Window Watchdog Timer Clock Source Selection (Write Protect)
CLK_CLKSEL1	[1:0] WDTSEL	Watchdog Timer Clock Source Selection (Write Protect)
CLK_PLLCTL	[23] STBSEL	PLL Stable Counter Selection (Write Protect)
CLK_PLLCTL	[19] PLLSRC	PLL Source Clock Selection (Write Protect)
CLK_PLLCTL	[18] OE	PLL OE (FOUT Enable) Pin Control (Write Protect)
CLK_PLLCTL	[17] BP	PLL Bypass Control (Write Protect)
CLK_PLLCTL	[16] PD	Power-down Mode (Write Protect)
CLK_PLLCTL	[15:14] OUTDIV	PLL Output Divider Control (Write Protect)
CLK_PLLCTL	[13:9] INDIV	PLL Input Divider Control (Write Protect)
CLK_PLLCTL	[8:0] FBDIV	PLL Feedback Divider Control (Write Protect)
CLK_CLKDSTS	[8] HXTFQIF	HXT Clock Frequency Range Detector Interrupt Flag (Write Protect)
CLK_CLKDSTS	[1] LXTFIF	LXT Clock Fail Interrupt Flag (Write Protect)
CLK_CLKDSTS	[0] HXTFIF	HXT Clock Fail Interrupt Flag (Write Protect)

SYS_IPRST0	[7] CRCRST	CRC Calculation Controller Reset (Write Protect)
SYS_IPRST0	[4] HDIV_RST	HDIV Controller Reset (Write Protect)
SYS_IPRST0	[3] EBIRST	EBI Controller Reset (Write Protect)
SYS_IPRST0	[2] PDMARST	PDMA Controller Reset (Write Protect)
SYS_IPRST0	[1] CPURST	Processor Core One-shot Reset (Write Protect)
SYS_IPRST0	[0] CHIPRST	Chip One-shot Reset (Write Protect)
SYS_BODCTL	[20] LVRVL	LVR Detector Threshold Voltage Selection (Write Protect)
SYS_BODCTL	[16] BODVL	Brown-out Detector Threshold Voltage Selection (Write Protect)
SYS_BODCTL	[14:12] LVRDGSEL	LVR Output De-glitch Time Select (Write Protect)
SYS_BODCTL	[10:8] BODDGSEL	Brown-out Detector Output De-glitch Time Select (Write Protect)
SYS_BODCTL	[7] LVREN	Low Voltage Reset Enable Bit (Write Protect)
SYS_BODCTL	[5] BODLPM	Brown-out Detector Low Power Mode (Write Protect)
SYS_BODCTL	[3] BODRSTEN	Brown-out Reset Enable Bit (Write Protect)
SYS_BODCTL	[0] BODEN	Brown-out Detector Enable Bit (Write Protect)
SYS_PORCTL	[15:0] POROFF	Power-on Reset Enable Bit (Write Protect)
SYS_SRAM_BISTCTL	[18] SRS2	SRAM Bank0 Section 2 BIST Select (Write Protect)
SYS_SRAM_BISTCTL	[17] SRS1	SRAM Bank0 Section 1 BIST Select (Write Protect)
SYS_SRAM_BISTCTL	[16] SRS0	SRAM Bank0 Section 0 BIST Select (Write Protect)
SYS_SRAM_BISTCTL	[7] PDMABIST	PDMA BIST Enable Bit (Write Protect)
SYS_SRAM_BISTCTL	[4] USBBIST	USB BIST Enable Bit (Write Protect)
SYS_SRAM_BISTCTL	[2] FMCBIST	FMC CACHE BIST Enable Bit (Write Protect)
SYS_SRAM_BISTCTL	[0] SRBIST	SRAM BIST Enable Bit (Write Protect)
SYS_PORDISAN	[15:0] POROFFAN	Power-on Reset Enable Bit (Write Protect)
NMIEN	[15] UART1_INT	UART1 NMI Source Enable (Write Protect)
NMIEN	[14] UART0_INT	UART0 NMI Source Enable (Write Protect)
NMIEN	[13] EINT5	External Interrupt From PB.7, PD.12 or PF.14 Pin NMI Source Enable (Write Protect)
NMIEN	[12] EINT4	External Interrupt From PA.8, PB.6 or PF.15 Pin NMI Source Enable (Write Protect)
NMIEN	[11] EINT3	External Interrupt From PB.2 or PC.7 Pin NMI Source Enable (Write Protect)
NMIEN	[10] EINT2	External Interrupt From PB.3 or PC.6 Pin NMI Source Enable (Write Protect)
NMIEN	[9] EINT1	External Interrupt From PA.7, PB.4 or PD.15 Pin NMI Source Enable (Write Protect)
NMIEN	[8] EINT0	External Interrupt From PA.6 or PB.5 Pin NMI Source Enable (Write Protect)
NMIEN	[6] RTC_INT	RTC NMI Source Enable (Write Protect)

NMIEN	[4] CLKFAIL	Clock Fail Detected and IRC Auto Trim Interrupt NMI Source Enable (Write Protect)
NMIEN	[3] SRAM_PERR	SRAM ParityCheck Error NMI Source Enable (Write Protect)
NMIEN	[2] PWRWU_INT	Power-down Mode Wake-up NMI Source Enable (Write Protect)
NMIEN	[1] IRC_INT	IRC TRIM NMI Source Enable (Write Protect)
NMIEN	[0] BODOUT	BOD NMI Source Enable (Write Protect)
FMC_ISPCTL	[24] INTEN	ISP Interrupt Enabled Bit (Write Protect)
FMC_ISPCTL	[6] ISPFF	ISP Fail Flag (Write Protect)
FMC_ISPCTL	[5] LDUEN	LDROM Update Enable Bit (Write Protect)
FMC_ISPCTL	[4] CFGUEN	CONFIG Update Enable Bit (Write Protect)
FMC_ISPCTL	[3] APUEN	APROM Update Enable Bit (Write Protect)
FMC_ISPCTL	[2] SPUEN	SPROM Update Enable Bit (Write Protect)
FMC_ISPCTL	[1] BS	Boot Selection (Write Protect)
FMC_ISPCTL	[0] ISPEN	ISP Enable Bit (Write Protect)
FMC_ISPTRG	[0] ISPGO	ISP Start Trigger (Write Protect)
FMC_FTCTL	[9] CACHEINV	Flash Cache Invalidation (Write Protect)
FMC_FTCTL	[6:4] FOM	Frequency Optimization Mode (Write Protect)
FMC_ISPSTS	[6] ISPFF	ISP Fail Flag (Write Protect)
TIMER0_CTL	[31] ICEDEBUG	ICE Debug Mode Acknowledge Disable Bit (Write Protect)
TIMER1_CTL	[31] ICEDEBUG	ICE Debug Mode Acknowledge Disable Bit (Write Protect)
TIMER2_CTL	[31] ICEDEBUG	ICE Debug Mode Acknowledge Disable Bit (Write Protect)
TIMER3_CTL	[31] ICEDEBUG	ICE Debug Mode Acknowledge Disable Bit (Write Protect)
WDT_CTL	[31] ICEDEBUG	ICE Debug Mode Acknowledge Disable Bit (Write Protect)
WDT_CTL	[11:8] TOUTSEL	WDT Time-out Interval Selection (Write Protect)
WDT_CTL	[7] WDTEN	WDT Enable Bit (Write Protect)
WDT_CTL	[6] INTEN	WDT Time-out Interrupt Enable Bit (Write Protect)
WDT_CTL	[5] WKF	WDT Time-out Wake-up Flag (Write Protect)
WDT_CTL	[4] WKEN	WDT Time-out Wake-up Function Control (Write Protect)
WDT_CTL	[1] RSTEN	WDT Time-out Reset Enable Bit (Write Protect)
WDT_ALTCTL	[1:0] RSTDSEL	WDT Reset Delay Selection (Write Protect)
BPWM_CTL0	[31] DBGTRIOFF	ICE Debug Mode Acknowledge Disable (Write Protect)
BPWM_CTL0	[30] DBGHALT	ICE Debug Mode Counter Halt (Write Protect)
PWM_CTL0	[31] DBGTRIOFF	ICE Debug Mode Acknowledge Disable Bit (Write Protect)
PWM_CTL0	[30] DBGHALT	ICE Debug Mode Counter Halt (Write Protect)
PWM_DTCTL0_1	[24] DTCKSEL	Dead-time Clock Select (Write Protect)
PWM_DTCTL0_1	[16] DTEN	Enable Dead-time Insertion for PWM Pair (PWM_CH0, PWM_CH1)

		(PWM_CH2, PWM_CH3) (PWM_CH4, PWM_CH5) (Write Protect)
PWM_DTCTL0_1	[11:0] DTCNT	Dead-time Counter (Write Protect)
PWM_DTCTL2_3	[24] DTCKSEL	Dead-time Clock Select (Write Protect)
PWM_DTCTL2_3	[16] DTEN	Enable Dead-time Insertion for PWM Pair (PWM_CH0, PWM_CH1) (PWM_CH2, PWM_CH3) (PWM_CH4, PWM_CH5) (Write Protect)
PWM_DTCTL2_3	[11:0] DTCNT	Dead-time Counter (Write Protect)
PWM_DTCTL4_5	[24] DTCKSEL	Dead-time Clock Select (Write Protect)
PWM_DTCTL4_5	[16] DTEN	Enable Dead-time Insertion for PWM Pair (PWM_CH0, PWM_CH1) (PWM_CH2, PWM_CH3) (PWM_CH4, PWM_CH5) (Write Protect)
PWM_DTCTL4_5	[11:0] DTCNT	Dead-time Counter (Write Protect)
PWM_BRKCTL0_1	[19:18] BRKAODD	PWM Brake Action Select for Odd Channel (Write Protect)
PWM_BRKCTL0_1	[17:16] BRKAEVEN	PWM Brake Action Select for Even Channel (Write Protect)
PWM_BRKCTL0_1	[15] SYSLBEN	Enable System Fail As Level-detect Brake Source (Write Protect)
PWM_BRKCTL0_1	[13] BRKP1LEN	Enable BKP1 Pin As Level-detect Brake Source (Write Protect)
PWM_BRKCTL0_1	[12] BRKP0LEN	Enable BKP0 Pin As Level-detect Brake Source (Write Protect)
PWM_BRKCTL0_1	[9] CPO1LBEN	Enable ACMP1_O Digital Output As Level-detect Brake Source (Write Protect)
PWM_BRKCTL0_1	[8] CPO0LBEN	Enable ACMP0_O Digital Output As Level-detect Brake Source (Write Protect)
PWM_BRKCTL0_1	[7] SYSEBEN	Enable System Fail As Edge-detect Brake Source (Write Protect)
PWM_BRKCTL0_1	[5] BRKP1EEN	Enable PWMx_BRAKE1 Pin As Edge-detect Brake Source (Write Protect)
PWM_BRKCTL0_1	[4] BRKP0EEN	Enable PWMx_BRAKE0 Pin As Edge-detect Brake Source (Write Protect)
PWM_BRKCTL0_1	[1] CPO1EBEN	Enable ACMP1_O Digital Output As Edge-detect Brake Source (Write Protect)
PWM_BRKCTL0_1	[0] CPO0EBEN	Enable ACMP0_O Digital Output As Edge-detect Brake Source (Write Protect)
PWM_BRKCTL2_3	[19:18] BRKAODD	PWM Brake Action Select for Odd Channel (Write Protect)
PWM_BRKCTL2_3	[17:16] BRKAEVEN	PWM Brake Action Select for Even Channel (Write Protect)
PWM_BRKCTL2_3	[15] SYSLBEN	Enable System Fail As Level-detect Brake Source (Write Protect)
PWM_BRKCTL2_3	[13] BRKP1LEN	Enable BKP1 Pin As Level-detect Brake Source (Write Protect)
PWM_BRKCTL2_3	[12] BRKP0LEN	Enable BKP0 Pin As Level-detect Brake Source (Write Protect)
PWM_BRKCTL2_3	[9] CPO1LBEN	Enable ACMP1_O Digital Output As Level-detect Brake Source (Write Protect)
PWM_BRKCTL2_3	[8] CPO0LBEN	Enable ACMP0_O Digital Output As Level-detect Brake Source (Write Protect)
PWM_BRKCTL2_3	[7] SYSEBEN	Enable System Fail As Edge-detect Brake Source (Write Protect)
PWM_BRKCTL2_3	[5] BRKP1EEN	Enable PWMx_BRAKE1 Pin As Edge-detect Brake Source (Write Protect)
PWM_BRKCTL2_3	[4] BRKP0EEN	Enable PWMx_BRAKE0 Pin As Edge-detect Brake Source (Write

		Protect)
PWM_BRKCTL2_3	[1] CPO1EBEN	Enable ACMP1_O Digital Output As Edge-detect Brake Source (Write Protect)
PWM_BRKCTL2_3	[0] CPO0EBEN	Enable ACMP0_O Digital Output As Edge-detect Brake Source (Write Protect)
PWM_BRKCTL4_5	[19:18] BRKAODD	PWM Brake Action Select for Odd Channel (Write Protect)
PWM_BRKCTL4_5	[17:16] BRKAEVEN	PWM Brake Action Select for Even Channel (Write Protect)
PWM_BRKCTL4_5	[15] SYSLBEN	Enable System Fail As Level-detect Brake Source (Write Protect)
PWM_BRKCTL4_5	[13] BRKP1LEN	Enable BKP1 Pin As Level-detect Brake Source (Write Protect)
PWM_BRKCTL4_5	[12] BRKP0LEN	Enable BKP0 Pin As Level-detect Brake Source (Write Protect)
PWM_BRKCTL4_5	[9] CPO1LBEN	Enable ACMP1_O Digital Output As Level-detect Brake Source (Write Protect)
PWM_BRKCTL4_5	[8] CPO0LBEN	Enable ACMP0_O Digital Output As Level-detect Brake Source (Write Protect)
PWM_BRKCTL4_5	[7] SYSEBEN	Enable System Fail As Edge-detect Brake Source (Write Protect)
PWM_BRKCTL4_5	[5] BRKP1EEN	Enable PWMx_BRAKE1 Pin As Edge-detect Brake Source (Write Protect)
PWM_BRKCTL4_5	[4] BRKP0EEN	Enable PWMx_BRAKE0 Pin As Edge-detect Brake Source (Write Protect)
PWM_BRKCTL4_5	[1] CPO1EBEN	Enable ACMP1_O Digital Output As Edge-detect Brake Source (Write Protect)
PWM_BRKCTL4_5	[0] CPO0EBEN	Enable ACMP0_O Digital Output As Edge-detect Brake Source (Write Protect)
PWM_SWBRK	[8+n/2] n=0,2,4 BRKLTRGn	PWM Level Brake Software Trigger (Write Only) (Write Protect)
PWM_SWBRK	[n/2] n=0,2,4 BRKETRGn	PWM Edge Brake Software Trigger (Write Only) (Write Protect)
PWM_INTEN1	[10] BRKLIEN4_5	PWM Level-detect Brake Interrupt Enable for Channel4/5 (Write Protect)
PWM_INTEN1	[9] BRKLIEN2_3	PWM Level-detect Brake Interrupt Enable for Channel2/3 (Write Protect)
PWM_INTEN1	[8] BRKLIEN0_1	PWM Level-detect Brake Interrupt Enable for Channel0/1 (Write Protect)
PWM_INTEN1	[2] BRKEIEN4_5	PWM Edge-detect Brake Interrupt Enable for Channel4/5 (Write Protect)
PWM_INTEN1	[1] BRKEIEN2_3	PWM Edge-detect Brake Interrupt Enable for Channel2/3 (Write Protect)
PWM_INTEN1	[0] BRKEIEN0_1	PWM Edge-detect Brake Interrupt Enable for Channel0/1 (Write Protect)
PWM_INTSTS1	[8+n] n=0,1..5 BRKLIFn	PWM Channel n Level-detect Brake Interrupt Flag (Write Protect)
PWM_INTSTS1	[n] n=0,1..5 BRKEIFn	PWM Channel n Edge-detect Brake Interrupt Flag (Write Protect)

ADC_ADCR	[12] RESET	ADC RESET (Write Protect)

6.3.11 UART0_TXD/USCI0_DAT1 modulation with PWM

This chip supports UART0_TXD/USCI0_DAT1 to modulate with PWM channel. User can set MODPWMSEL(SYS_MODCTL[7:4]) to choose which PWM0 channel to modulate with UART0_TXD/USCI0_DAT1 and set MODEN(SYS_MODCTL[0]) to enable modulation function. User can set TXDINV(UART_LINE[8]) to inverse UART0_TXD or DATOINV(UUART_LINECTL[5]) to inverse USCI0_DAT1 before modulating with PWM.

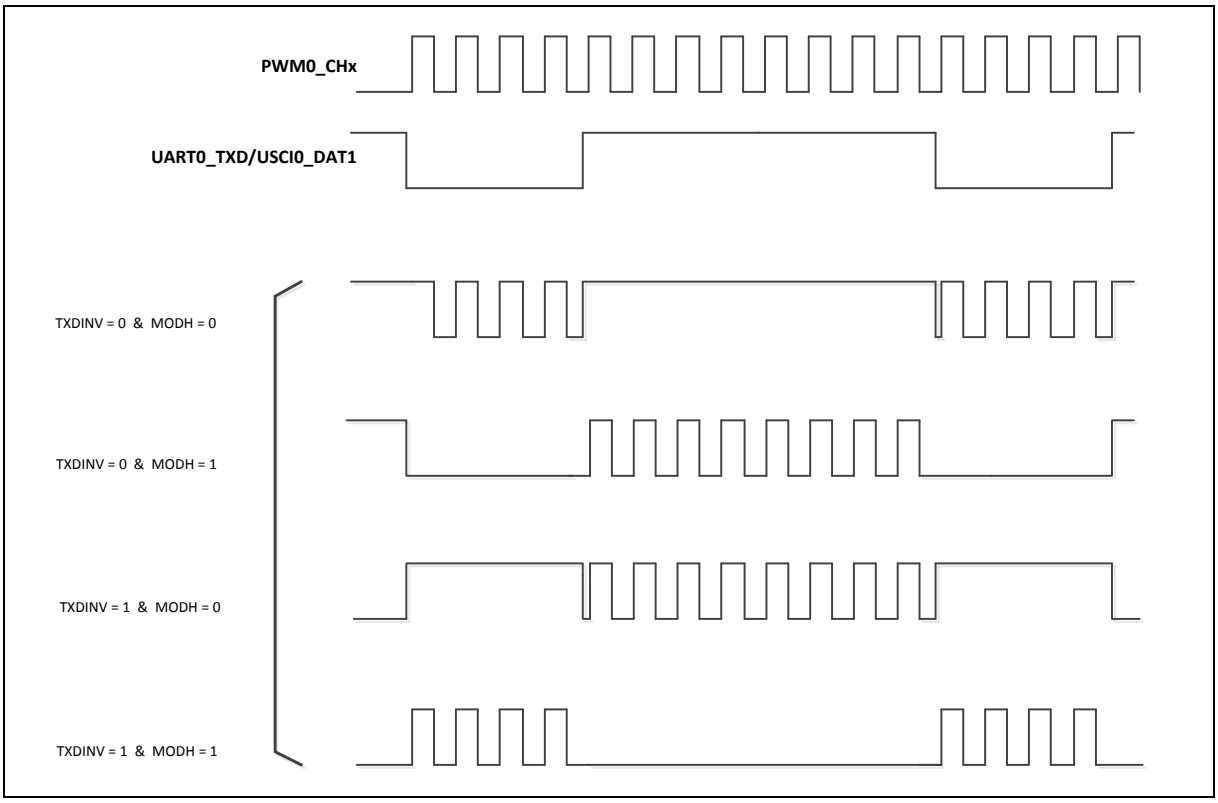


Figure 6.3-11 UART0_TXD/USCI0_DAT1 Modulated with PWM Channel

6.3.12 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	0xXXXX_XXXX
SYS_RSTSTS	SYS_BA+0x04	R/W	System Reset Status Register	0x0000_0043
SYS_IPRST0	SYS_BA+0x08	R/W	Peripheral Reset Control Register 0	0x0000_0000
SYS_IPRST1	SYS_BA+0x0C	R/W	Peripheral Reset Control Register 1	0x0000_0000
SYS_IPRST2	SYS_BA+0x10	R/W	Peripheral Reset Control Register 2	0x0000_0000
SYS_BODCTL	SYS_BA+0x18	R/W	Brown-out Detector Control Register	0x00XX_038X
SYS_PORCTL	SYS_BA+0x24	R/W	Power-On-reset Controller 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_MODCTL	SYS_BA+0xC0	R/W	Modulation Control Register	0x0000_0000
SYS_SRAM_BISTCTL	SYS_BA+0xD0	R/W	System SRAM BIST Test Control Register	0x000X_0000
SYS_SRAM_BISTSTS	SYS_BA+0xD4	R	System SRAM BIST Test Status Register	0x00xx_00xx

SYS_SRAM_INTCTL	SYS_BA+0xDC	R/W	System SRAM Interrupt Enable Control Register	0x0000_0000
SYS_SRAM_STATUS	SYS_BA+0xE0	R/W	System SRAM Parity Error Status Register	0x0000_0000
SYS_SRAM_ERRADDR	SYS_BA+0xE4	R	System SRAM Parity Check Error Address Register	0x0000_0000
SYS_HIRCTRIMCTL	SYS_BA+0xF0	R/W	HIRC Trim Control Register	0x0008_0000
SYS_HIRCTRIMIEN	SYS_BA+0xF4	R/W	HIRC Trim Interrupt Enable Register	0x0000_0000
SYS_HIRCTRIMSTS	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

6.3.13 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	0xXXXX_XXXX

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

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

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

System Reset Status Register (SYS_RSTSTS)

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

Register	Offset	R/W	Description	Reset Value
SYS_RSTSTS	SYS_BA+0x04	R/W	System Reset Status Register	0x0000_0043

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**CPULKRF**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CPURF	**Reserved**	**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®-M0 lockup happened and chip is reset. **Note:** 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®- M0 Core and Flash Memory Controller (FMC). 0 = No reset from CPU. 1 = The Cortex®-M0 Core and FMC are reset by software setting CPURST to 1. **Note:** Write to clear this bit to 0.
[6]	**Reserved**	Reserved.
[5]	**SYSRF**	**System Reset Flag** The system reset flag is set by the "Reset Signal" from the Cortex®-M0 Core to indicate the previous reset source. 0 = No reset from Cortex®-M0. 1 = The Cortex®- M0 had issued the reset signal to reset the system by writing 1 to the bit SYSRESETREQ(AIRCR[2], Application Interrupt and Reset Control Register, address = 0xE000ED0C) in system control registers of Cortex®-M0 core. **Note:** 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		HDIV_RST	EBIRST	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.
[6:5]	**Reserved**	Reserved.
[4]	**HDIV_RST**	**HDIV Controller Reset (Write Protect)** Set this bit to 1 will generate a reset signal to the hardware divider. User need to set this bit to 0 to release from the reset state. 0 = Hardware divider controller normal operation. 1 = Hardware divider controller reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[3]	**EBIRST**	**EBI Controller Reset (Write Protect)** Set this bit to 1 will generate a reset signal to the EBI. User needs to set this bit to 0 to release from the reset state. 0 = EBI controller normal operation. 1 = EBI controller reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[2]	**PDMARST**	**PDMA Controller Reset (Write Protect)** Setting this bit to 1 will generate a reset signal to the PDMA. User needs to set this bit to 0 to release from reset state. 0 = PDMA controller normal operation. 1 = PDMA controller reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[1]	**CPURST**	**Processor Core One-shot Reset (Write Protect)** Setting this bit will only reset the processor core and Flash Memory Controller(FMC), and

		this bit will automatically return to 0 after the 2 clock cycles. 0 = Processor core normal operation. 1 = Processor core one-shot reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[0]	**CHIPRST**	**Chip One-shot Reset (Write Protect)** Setting this bit will reset the whole chip, including Processor core and all peripherals, and this bit will automatically return to 0 after the 2 clock cycles. The CHIPRST is same as the POR reset, all the chip controllers is reset and the chip setting from Flash are also reload. About the difference between CHIPRST and SYSRESETREQ(AIRCR[2]), please refer to section 6.3.2. 0 = Chip normal operation. 1 = Chip one-shot reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register. Note : reset by powr on reset

Peripheral Reset Control Register 1 (SYS_IPRST1)

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

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

31	30	29	28	27	26	25	24
Reserved			ADCRST	USBDRST	Reserved		
23	22	21	20	19	18	17	16
UART7RST	UART6RST	UART5RST	UART4RST	UART3RST	UART2RST	UART1RST	UART0RST
15	14	13	12	11	10	9	8
Reserved		SPI0RST	QSPI0RST	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]	**ADCRST**	**ADC Controller Reset** 0 = ADC controller normal operation. 1 = ADC controller reset.
[27]	**USBDRST**	**USBD Controller Reset** 0 = USBD controller normal operation. 1 = USBD controller reset.
[26:24]	**Reserved**	Reserved.
[23]	**UART7RST**	**UART7 Controller Reset** 0 = UART7 controller normal operation. 1 = UART7 controller reset.
[22]	**UART6RST**	**UART6 Controller Reset** 0 = UART6 controller normal operation. 1 = UART6 controller reset.
[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.
[15:14]	**Reserved**	Reserved.
[13]	**SPI0RST**	**SPI0 Controller Reset** 0 = SPI0 controller normal operation. 1 = SPI0 controller reset.
[12]	**QSPI0RST**	**QSPI0 Controller Reset** 0 = QSPI0 controller normal operation. 1 = QSPI0 controller reset.
[11:10]	**Reserved**	Reserved.
[9]	**I2C1RST**	**I2C1 Controller Reset** 0 = I2C1 controller normal operation. 1 = I2C1 controller reset.
[8]	**I2C0RST**	**I2C0 Controller Reset** 0 = I2C0 controller normal operation. 1 = I2C0 controller reset.
[7]	**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							
23	22	21	20	19	18	17	16
Reserved				BPWM1RST	BPWM0RST	PWM1RST	PWM0RST
15	14	13	12	11	10	9	8
Reserved						USCI1RST	USCI0RST
7	6	5	4	3	2	1	0
Reserved							

Bits	Description	
[31:20]	**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]	**PWM1RST**	**PWM1 Controller Reset** 0 = PWM1 controller normal operation. 1 = PWM1 controller reset.
[16]	**PWM0RST**	**PWM0 Controller Reset** 0 = PWM0 controller normal operation. 1 = PWM0 controller reset.
[15:10]	**Reserved**	Reserved.
[9]	**USCI1RST**	**USCI1 Controller Reset** 0 = USCI1 controller normal operation. 1 = USCI1 controller reset.
[8]	**USCI0RST**	**USCI0 Controller Reset** 0 = USCI0 controller normal operation. 1 = USCI0 controller reset.
[7:0]	**Reserved**	Reserved.

Brown-out Detector Control Register (SYS_BODCTL)

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

Register	Offset	R/W	Description	Reset Value
SYS_BODCTL	SYS_BA+0x18	R/W	Brown-out Detector Control Register	0x00XX_038X

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			LVRVL	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	
[31:21]	**Reserved**	Reserved.
[20]	**LVRVL**	**LVR Detector Threshold Voltage Selection (Write Protect)** The default value is set by Flash controller user configuration register LVRLVSEL (CONFIG0 [29]). 0 = LVR-Out Detector threshold voltage is 1.6V. 1 = LVR-Out Detector threshold voltage is 1.7V. **Note 1:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note 2:** This bit is only for special case. **Note 3:** reset by powr on reset
[19:17]	**Reserved**	Reserved.
[16]	**BODVL**	**Brown-out Detector Threshold Voltage Selection (Write Protect)** The default value is set by Flash controller user configuration register CBOV (CONFIG0 [21]). 0 = Brown-Out Detector threshold voltage is 2.0V. 1 = Brown-Out Detector threshold voltage is 2.5V. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register. Note : reset by powr on reset
[15]	**Reserved**	Reserved.

Bits	Description	
[14:12]	**LVRDGSEL**	**LVR Output De-glitch Time Select (Write Protect)** 000 = Without de-glitch function. 001 = 64 system clock (HCLK). 010 = 128 system clock (HCLK). 011 = 256 system clock (HCLK). 100 = 512 system clock (HCLK). 101 = 1024 system clock (HCLK). 110 = 2048 system clock (HCLK). 111 = 4096 system clock (HCLK). **Note:** These bits are write protected. Refer to the SYS_REGLCTL register.
[11]	**Reserved**	Reserved.
[10:8]	**BODDGSEL**	**Brown-out Detector Output De-glitch Time Select (Write Protect)** 000 = BOD output is sampled by LIRC/4 clock. 001 = 64 system clock (HCLK). 010 = 128 system clock (HCLK). 011 = 256 system clock (HCLK). 100 = 512 system clock (HCLK). 101 = 1024 system clock (HCLK). 110 = 2048 system clock (HCLK). 111 = 4096 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 200us 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 0000.
[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.

Bits	Description	
[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.
[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 keep till to the BODEN set to 0. BOD interrupt can be blocked by disabling the NVIC BOD interrupt or disabling BOD function (set BODEN low). **Note 2:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note 3:** Reset by powr on reset
[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 1:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note 2:** Reset by powr on reset

Power-on Reset Controller Register (SYS_PORCTL)

Register	Offset	R/W	Description	Reset Value
SYS_PORCTL	SYS_BA+0x24	R/W	Power-On-reset Controller Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**POROFF**	**Power-on Reset Enable Bit (Write Protect)** When powered on, the POR circuit generates a reset signal to reset the whole chip function, but noise on the power may cause the POR active again. User can disable internal POR circuit to avoid unpredictable noise to cause chip reset by writing 0x5AA5 to this field. The POR function will be active again when this field is set to another value or chip is reset by other reset source, including: nRESET, Watchdog, LVR reset, BOD reset, ICE reset command and the software-chip reset function. **Note:** 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
PC15MFP				PC14MFP			
23	22	21	20	19	18	17	16
PC13MFP				PC12MFP			
15	14	13	12	11	10	9	8
PC11MFP				PC10MFP			
7	6	5	4	3	2	1	0
PC9MFP				PC8MFP			

Bits	Description	
[31:28]	PC15MFP	PC.15 Multi-function Pin Selection
[27:24]	PC14MFP	PC.14 Multi-function Pin Selection
[23:20]	PC13MFP	PC.13 Multi-function Pin Selection
[19:16]	PC12MFP	PC.12 Multi-function Pin Selection
[15:12]	PC11MFP	PC.11 Multi-function Pin Selection
[11:8]	PC10MFP	PC.10 Multi-function Pin Selection
[7:4]	PC9MFP	PC.9 Multi-function Pin Selection
[3:0]	PC8MFP	PC.8 Multi-function Pin Selection

GPIOD Low Byte Multiple Function Control Register (SYS_GPD_MFPL)

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

31	30	29	28	27	26	25	24
PD7MFP				PD6MFP			
23	22	21	20	19	18	17	16
PD5MFP				PD4MFP			
15	14	13	12	11	10	9	8
PD3MFP				PD2MFP			
7	6	5	4	3	2	1	0
PD1MFP				PD0MFP			

Bits	Description	
[31:28]	PD7MFP	PD.7 Multi-function Pin Selection
[27:24]	PD6MFP	PD.6 Multi-function Pin Selection
[23:20]	PD5MFP	PD.5 Multi-function Pin Selection
[19:16]	PD4MFP	PD.4 Multi-function Pin Selection
[15:12]	PD3MFP	PD.3 Multi-function Pin Selection
[11:8]	PD2MFP	PD.2 Multi-function Pin Selection
[7:4]	PD1MFP	PD.1 Multi-function Pin Selection
[3:0]	PD0MFP	PD.0 Multi-function Pin Selection

GPIOD High Byte Multiple Function Control Register (SYS_GPD_MFPH)

Register	Offset	R/W	Description	Reset Value
SYS_GPD_MFPH	SYS_BA+0x4C	R/W	GPIOD High Byte Multiple Function Control Register	0x0000_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
PG7MFP				PG6MFP			
23	22	21	20	19	18	17	16
PG5MFP				PG4MFP			
15	14	13	12	11	10	9	8
PG3MFP				PG2MFP			
7	6	5	4	3	2	1	0
PG1MFP				PG0MFP			

Bits	Description	
[31:28]	PG7MFP	PG.7 Multi-function Pin Selection
[27:24]	PG6MFP	PG.6 Multi-function Pin Selection
[23:20]	PG5MFP	PG.5 Multi-function Pin Selection
[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
[7:4]	PG1MFP	PG.1 Multi-function Pin Selection
[3:0]	PG0MFP	PG.0 Multi-function Pin Selection

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
PH3MFP				PH2MFP			
7	6	5	4	3	2	1	0
PH1MFP				PH0MFP			

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
[15:12]	PH3MFP	PH.3 Multi-function Pin Selection
[11:8]	PH2MFP	PH.2 Multi-function Pin Selection
[7:4]	PH1MFP	PH.1 Multi-function Pin Selection
[3:0]	PH0MFP	PH.0 Multi-function Pin Selection

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
PH15MFP				PH14MFP			
23	22	21	20	19	18	17	16
PH13MFP				PH12MFP			
15	14	13	12	11	10	9	8
PH11MFP				PH10MFP			
7	6	5	4	3	2	1	0
PH9MFP				PH8MFP			

Bits	Description	
[31:28]	PH15MFP	PH.15 Multi-function Pin Selection
[27:24]	PH14MFP	PH.14 Multi-function Pin Selection
[23:20]	PH13MFP	PH.13 Multi-function Pin Selection
[19:16]	PH12MFP	PH.12 Multi-function Pin Selection
[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

Modulation Control Register (SYS_MODCTL)

Register	Offset	R/W	Description	Reset Value
SYS_MODCTL	SYS_BA+0xC0	R/W	Modulation Control Register	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:4]	**MODPWMSEL**	**PWM0 Channel Select for Modulation** Select the PWM0 channel to modulate with the UART0_TXD or USCI0_DAT1. 0000: PWM0 Channel 0 modulate with UART0_TXD. 0001: PWM0 Channel 1 modulate with UART0_TXD. 0010: PWM0 Channel 2 modulate with UART0_TXD. 0011: PWM0 Channel 3 modulete with UART0_TXD. 0100: PWM0 Channel 4 modulete with UART0_TXD. 0101: PWM0 Channel 5 modulete with UART0_TXD. 0110: Reserved. 0111: Reserved. 1000: PWM0 Channel 0 modulate with USCI0_DAT1. 1001: PWM0 Channel 1 modulate with USCI0_DAT1. 1010: PWM0 Channel 2 modulate with USCI0_DAT1. 1011: PWM0 Channel 3 modulete with USCI0_DAT1. 1100: PWM0 Channel 4 modulete with USCI0_DAT1. 1101: PWM0 Channel 5 modulete with USCI0_DAT1. 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(PWM0) at high or low of UART0_TXD or USCI0_DAT1 0: Modulation pulse at UART0_TXD low or USCI0_DAT1 low. 1: Modulation pulse at UART0_TXD high or USCI0_DAT1 high.
[0]	**MODEN**	**Modulation Function Enable Bit** This bit enables modulation funcion by modulating with PWM0 channel output and

		USCI0(USCI0_DAT1) or UART0(UART0_TXD) output. 0 = Modulation Function Disabled. 1 = Modulation Function Enabled.

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	0x000X_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					SRS2	SRS1	SRS0
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PDMABIST	Reserved		USBBIST	Reserved	FMCBIST	Reserved	SRBIST

Bits	Description	
[31:19]	**Reserved**	Reserved.
[18]	**SRS2**	**SRAM Bank0 Section 2 BIST Select (Write Protect)** This bit define if the bank0 section2 (0x2001_0000~0x2001_7FFF) of SRAM is selected or not when doing bist test. 0 = SRAM back0 section2 is deselected when doing bist test. 1 = SRAM back0 section2 is selected when doing bist test. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note:** At least one section of SRAM should be selected when doing SRAM bist test.
[17]	**SRS1**	**SRAM Bank0 Section 1 BIST Select (Write Protect)** This bit define if the bank0 section1 (0x2000_8000~0x2000_FFFF) of SRAM is selected or not when doing bist test. 0 = SRAM bank0 section1 is deselected when doing bist test. 1 = SRAM bank0 section1 is selected when doing bist test. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note:** At least one section of SRAM should be selected when doing SRAM bist test.
[16]	**SRS0**	**SRAM Bank0 Section 0 BIST Select (Write Protect)** This bit define if the bank0 section0 (0x2000_0000~0x2000_7FFF) of SRAM is selected or not when doing bist test. 0 = SRAM bank0 section0 is deselected when doing bist test. 1 = SRAM bank0 section0 is selected when doing bist test. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note:** At least one section of SRAM should be selected when doing SRAM bist test.
[15:8]	**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.
[6:5]	**Reserved**	Reserved.
[4]	**USBBIST**	**USB BIST Enable Bit (Write Protect)** This bit enables BIST test for USB RAM 0 = system USB BIST Disabled. 1 = system USB BIST Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[3]	**Reserved**	Reserved.
[2]	**FMCBIST**	**FMC 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]	**Reserved**	Reserved.
[0]	**SRBIST**	**SRAM BIST Enable Bit (Write Protect)** This bit enables BIST test for SRAM. 0 = system SRAM BIST Disabled. 1 = system SRAM 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
PDMAEND	Reserved		USBBEND	Reserved		CRBEND	SRBEND
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PDMABISTF	Reserved		USBBEF	Reserved		CRBISTEF	SRBISTEF

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23]	**PDMAEND**	**PDMA SRAM BIST Test Finish** 0 = PDMA SRAM BIST is active. 1 = PDMA SRAM BIST test finish.
[22:21]	**Reserved**	Reserved.
[20]	**USBBEND**	**USB SRAM BIST Test Finish** 0 = USB SRAM BIST is active. 1 = USB SRAM BIST test finish.
[19:18]	**Reserved**	Reserved.
[17]	**CRBEND**	**CACHE SRAM BIST Test Finish** 0 = System CACHE RAM BIST is active. 1 = System CACHE RAM BIST test finished.
[16]	**SRBEND**	**System SRAM BIST Test Finish** 0 = System SRAM BIST active. 1 = System SRAM BIST finish.
[15:8]	**Reserved**	Reserved.
[7]	**PDMABISTF**	**PDMA SRAM BIST Failed Flag** 0 = PDMA SRAM BIST pass. 1 = PDMA SRAM BIST failed.
[6:5]	**Reserved**	Reserved.
[4]	**USBBEF**	**USB SRAM BIST Fail Flag** 0 = USB SRAM BIST test pass. 1 = USB SRAM BIST test fail.

[3:2]	**Reserved**	Reserved.
[1]	**CR0BISTEF**	**CACHE SRAM BIST Fail Flag** 0 = System CACHE RAM BIST test pass. 1 = System CACHE RAM BIST test failed.
[0]	**SRBISTEF**	**System SRAM BIST Fail Flag** 0 = System SRAM BIST test pass. 1 = System SRAM BIST test fail.

System SRAM Parity Error Interrupt Enable Control Register (SYS_SRAM_INTCTL)

Register	Offset	R/W	Description	Reset Value
SYS_SRAM_INTCTL	SYS_BA+0xDC	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+0xE0	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+0xE4	R	System SRAM Parity Check Error Address Register	0x0000_0000

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

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

HIRC Trim Control Register (SYS_HIRCTRIMCTL)

Register	Offset	R/W	Description	Reset Value
SYS_HIRCTRIMCTL	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 0x1F, 0x0 is reserved. **Note:** This field is effective only when the BOUNDEN(SYS_HIRCTRIMCTL[9]) is enable.
[15:11]	**Reserved**	Reserved.
[10]	**REFCKSEL**	**Reference Clock Selection** 0 = HIRC trim reference clock is from LXT (32.768 kHz). 1 = HIRC trim reference clock is from internal USB synchronous mode. **Note 1:** HIRC trim reference clock supports LXT or internal USB synchronous mode depending on the chip spec. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information. **Note 2:** If there is no reference clock (LXT or internal USB synchronous mode) when the rc_trim is enabled, **CLKERIF (SYS_HIRCTRIMCTL[2]) will be set to 1.**
[9]	**BOUNDEN**	**Boundary Enable Bit** 0 = Boundary function Disabled. 1 = Boundary function Enabled.
[8]	**CESTOPEN**	**Clock Error Stop Enable Bit** 0 = The trim operation is keep going if clock is inaccuracy. 1 = The trim operation is stopped if clock is inaccuracy.
[7:6]	**RETRYCNT**	**Trim Value Update Limitation Count** This field defines that how many times the auto trim circuit will try to update the HIRC trim value before the frequency of HIRC locked. Once the HIRC locked, the internal trim value update counter will be reset. If the trim value update counter reached this limitation value and frequency of HIRC still doesn't lock, the auto trim operation will be disabled and FREQSEL will be cleared to 00. 00 = Trim retry count limitation is 64 loops. 01 = Trim retry count limitation is 128 loops.

		10 = Trim retry count limitation is 256 loops. 11 = Trim retry count limitation is 512 loops.
[5:4]	**LOOPSEL**	**Trim Calculation Loop Selection** This field defines that trim value calculation is based on how many reference clocks. 00 = Trim value calculation is based on average difference in 4 clocks of reference clock. 01 = Trim value calculation is based on average difference in 8 clocks of reference clock. 10 = Trim value calculation is based on average difference in 16 clocks of reference clock. 11 = Trim value calculation is based on average difference in 32 clocks of reference clock. **Note:** For example, if LOOPSEL is set as 00, auto trim circuit will calculate trim value based on the average frequency difference in 4 clocks of reference clock.
[3:2]	**Reserved**	Reserved.
[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.

HIRC Trim Interrupt Enable Register (SYS_HIRCTRIMIEN)

Register	Offset	R/W	Description	Reset Value
SYS_HIRCTRIMIEN	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	TFALIEN	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 inaccuracy during auto trim operation. If this bit is set to1, and CLKERRIF(SYS_HIRCTRIMSTS[2]) is set during auto trim operation, an interrupt will be triggered to notify the clock frequency is inaccuracy. 0 = Disable CLKERRIF(SYS_HIRCTRIMSTS[2]) status to trigger an interrupt to CPU. 1 = Enable CLKERRIF(SYS_HIRCTRIMSTS[2]) status to trigger an interrupt to CPU.
[1]	**TFALIEN**	**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_HIRCTRIMCTL[1:0]). If this bit is high and TFAILIF(SYS_HIRCTRIMSTS[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_HIRCTRIMSTS[1]) status to trigger an interrupt to CPU. 1 = Enable TFAILIF(SYS_HIRCTRIMSTS[1]) status to trigger an interrupt to CPU.
[0]	**Reserved**	Reserved.

HIRC Trim Interrupt Status Register (SYS_HIRCTRIMSTS)

Register	Offset	R/W	Description	Reset Value
SYS_HIRCTRIMSTS	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	CLKERIF	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:** Write 1 to clear this flag.
[2]	**CLKERIF**	**Clock Error Interrupt Status** When the frequency relation between reference clock (LXT or USB sync signals) and 48 MHz internal high speed RC oscillator (HIRC) is shift larger to unreasonable value, this bit will be set and to be an indicate that clock frequency is inaccuracy Once this bit is set to 1, the auto trim operation stopped and FREQSEL(SYS_HIRCTRIMCTL[1:0]) will be cleared to 00 by hardware automatically if CESTOPEN(SYS_HIRCTRIMCTL[8]) is set to 1. If this bit is set and CLKEIEN(SYS_HIRCTIEN[2]) is high, an interrupt will be triggered to notify the clock frequency is inaccuracy. Write 1 to clear this to 0. 0 = Clock frequency is accuracy. 1 = Clock frequency is inaccuracy. **Note** : Reset by powr on reset
[1]	**TFAILIF**	**Trim Failure Interrupt Status** This bit indicates that HIRC trim value update limitation count reached and the HIRC clock frequency still doesn't be locked. Once this bit is set, the auto trim operation stopped and FREQSEL(SYS_HIRCTRIMCTL[1:0]) will be cleared to 00 by hardware automatically. If this bit is set and TFAILIEN(SYS_HIRCIEN[1]) is high, an interrupt will be triggered to notify that HIRC trim value update limitation count was reached. Write 1 to clear this to 0. 0 = Trim value update limitation count does not reach. 1 = Trim value update limitation count reached and HIRC frequency still not locked. **Note** : Reset by powr on reset
[0]	**FREQLOCK**	**HIRC Frequency Lock Status**

		This bit indicates the HIRC frequency is locked. This is a status bit and doesn't trigger any interrupt Write 1 to clear this to 0. This bit will be set automatically, if the frequecy is lock and the RC_TRIM is enabled. 0 = The internal high-speed oscillator frequency doesn't lock at 48 MHz yet. 1 = The internal high-speed oscillator frequency locked at 48 MHz. **Note** : Reset by powr on reset.

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 to disable register protection. For user to program these protected registers, a register protection disable sequence needs to be followed by a special programming. The register protection disable sequence is writing the data “59h”, “16h” “88h” to the register SYS_REGLCTL address at 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 writen 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.

6.3.14 System Timer (SysTick)

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

When system timer is enabled, it will count down from the value in the SysTick Current Value Register (SYST_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®-M0 Technical Reference Manual” and “Arm® v6-M Architecture Reference Manual”.

6.3.14.1 System Timer Control Register Map

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

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

6.3.14.2 System Timer Control Register Description

SysTick Control and Status Register (SYST_CTRL)

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

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

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

SysTick Reload Value Register (SYST_LOAD)

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

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**RELOAD**	**System Tick Reload Value** The 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.

6.3.15 Nested Vectored Interrupt Controller (NVIC)

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

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

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

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

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

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

6.3.15.1 Exception Model and System Interrupt Map

Table 6.3-8 lists the exception model supported by the M031/M032 series. Software can set four levels of priority on some of these exceptions as well as on all interrupts. The highest user-configurable priority is denoted as "0" and the lowest priority is denoted as "3". The default priority of all the user-configurable interrupts is "0". Note that priority "0" is treated as the fourth priority on the system, after three system exceptions "Reset", "NMI" and "Hard Fault".

Exception Name	Vector Number	Priority
Reset	1	-3
NMI	2	-2
Hard Fault	3	-1
Reserved	4 ~ 10	Reserved
SVCall	11	Configurable
Reserved	12 ~ 13	Reserved
PendSV	14	Configurable

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

Table 6.3-8 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	WDT_INT	Watchdog Timer interrupt
18	2	EINT024	External interrupt fromEINT0,2,4.
19	3	EINT135	External interrupt fromEINT1.3.5
20	4	GPABGH_INT	External interrupt from PA, PB, PG, PH pin
21	5	GPCDEF_INT	External interrupt from PC, PD, PE, PF pin
22	6	PWM0_INT	PWM0 interrupt
23	7	PWM1_INT	PWM1 interrupt
24	8	TMR0_INT	Timer 0 interrupt
25	9	TMR1_INT	Timer 1 interrupt
26	10	TMR2_INT	Timer 2 interrupt
27	11	TMR3_INT	Timer 3 interrupt
28	12	UART02_INT	UART0,2 interrupt
29	13	UART13_INT	UART1,3 interrupt
30	14	SPI0_INT	SPI0 interrupt
31	15	QSPI0_INT	QSPI0 interrupt
32	16	Reserved	Reserved
33	17	UART57_INT	UART5,7 interrupt
34	18	I2C0_INT	I2C0 interrupt
35	19	I2C1_INT	I2C1 interrupt
36	20	BPWM0_INT	BPWM0 interrupt
37	21	BPWM1_INT	BPWM1 interrupt
38	22	USCI01	USCI0,1 interrupt
39	23	USBD_INT	USB device interrupt
40	24	Reserved	Reserved
41	25	ACMP01_INT	ACMP0 and ACMP1 interrupt
42	26	PDMA_INT	PDMA interrupt
43	27	UART46_INT	UART4,6 interrupt

44	28	PWRWU_INT	Clock controller interrupt for chip wake-up from power-down state
45	29	ADC_INT	ADC interrupt
46	30	CLKFAIL	Clock fail detected or IRC Auto Trim interrupt or SRAM parity check error interrupt
47	31	RTC_INT	RTC interrupt

Table 6.3-9 Interrupt Number Table

6.3.15.2 Vector Table

When an interrupt is accepted, the processor will automatically fetch the starting address of the interrupt service routine (ISR) from a vector table in memory. For Armv6-M, the vector table base address is fixed at 0x00000000. The vector table contains the initialization value for the stack pointer on reset, and the entry point addresses for all exception handlers. The vector number on previous page defines the order of entries in the vector table associated with exception handler entry as illustrated in previous section.

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

Table 7.2-10 Vector Figure Format

6.3.15.3 Operation Description

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

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

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

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

6.3.15.4 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+0x000	R/W	IRQ0 ~ IRQ31 Set-enable Control Register	0x0000_0000
NVIC_ICER0	NVIC_BA+0x080	R/W	IRQ0 ~ IRQ31 Clear-enable Control Register	0x0000_0000
NVIC_ISPR0	NVIC_BA+0x100	R/W	IRQ0 ~ IRQ31 Set-pending Control Register	0x0000_0000
NVIC_ICPR0	NVIC_BA+0x180	R/W	IRQ0 ~ IRQ31 Clear-pending Control Register	0x0000_0000
NVIC_IABR0	NVIC_BA+0x200	R/W	IRQ0 ~ IRQ31 Active Bit Register	0x0000_0000
NVIC_IPRn **n=0,1..7**	0xE000E400 +0x4*n	R/W	IRQ0 ~ IRQ31 Priority Control Register	0x0000_0000
STIR	0xE000EF00	R/W	Software Trigger Interrupt Registers	0x0000_0000

IRQ0 ~ IRQ31 Set-enable Control Register (NVIC_ISER0)

Register	Offset	R/W	Description	Reset Value
NVIC_ISER0	NVIC_BA+0x000	R/W	IRQ0 ~ IRQ31 Set-enable Control Register	0x0000_0000

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

Bits	Description	
[31:0]	**SETENA**	**Interrupt Set Enable Bit** The NVIC_ISER0 registers enable interrupts, and show which interrupts are enabled Write Operation: 0 = No effect. 1 = Interrupt Enabled. Read Operation: 0 = Interrupt Disabled. 1 = Interrupt Enabled.

IRQ0 ~ IRQ31 Clear-enable Control Register (NVIC_ICER0)

Register	Offset	R/W	Description	Reset Value
NVIC_ICER0	NVIC_BA+0x080	R/W	IRQ0 ~ IRQ31 Clear-enable Control Register	0x0000_0000

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

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

IRQ0 ~ IRQ31 Set-pending Control Register (NVIC_ISPR0)

Register	Offset	R/W	Description	Reset Value
NVIC_ISPR0	NVIC_BA+0x100	R/W	IRQ0 ~ IRQ31 Set-pending Control Register	0x0000_0000

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

Bits	Description	
[31:0]	**SETPEND**	**Interrupt Set-pending** The NVIC_ISPR0 registers force interrupts into the pending state, and show which interrupts are pending Write Operation: 0 = No effect. 1 = Changes interrupt state to pending. Read Operation: 0 = Interrupt is not pending. 1 = Interrupt is pending.

IRQ0 ~ IRQ31 Clear-pending Control Register (NVIC_ICPR0)

Register	Offset	R/W	Description	Reset Value
NVIC_ICPR0	NVIC_BA+0x180	R/W	IRQ0 ~ IRQ31 Clear-pending Control Register	0x0000_0000

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

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

IRQ0 ~ IRQ31 Active Bit Register (NVIC_IABR0)

Register	Offset	R/W	Description	Reset Value
NVIC_IABR0	NVIC_BA+0x200	R/W	IRQ0 ~ IRQ31 Active Bit Register	0x0000_0000

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

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

IRQ0 ~ IRQ31 Interrupt Priority Register (NVIC_IPRn)

Register	Offset	R/W	Description	Reset Value
NVIC_IPRn **n=0,1..7**	0xE000E400 +0x4*n	R/W	IRQ0 ~ IRQ31 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	
[31:30]	**PRI_4n_3**	**Priority of IRQ_4n+3** "0" denotes the highest priority and "3" denotes the lowest priority
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_4n_2**	**Priority of IRQ_4n+2** "0" denotes the highest priority and "3" denotes the lowest priority
[21:16]	**Reserved**	Reserved.
[15:14]	**PRI_4n_1**	**Priority of IRQ_4n+1** "0" denotes the highest priority and "3" denotes the lowest priority
[13:8]	**Reserved**	Reserved.
[7:6]	**PRI_4n_0**	**Priority of IRQ_4n+0** "0" denotes the highest priority and "3" denotes the lowest priority
[5:0]	**Reserved**	Reserved.

Software Trigger Interrupt Register (STIR)

Register	Offset	R/W	Description	Reset Value
STIR	0xE000EF00	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 0-31. For example, a value of 0x03 specifies interrupt IRQ3.

6.3.15.5 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 Pin 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 and IRC Auto Trim Interrupt NMI Source Enable (Write Protect)** 0 = Clock fail detected and IRC Auto Trim interrupt NMI source Disabled. 1 = Clock fail detected and IRC Auto Trim 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 = UART1 interrupt is deasserted. 1 = UART1 interrupt is asserted.
[13]	**EINT5**	**External Interrupt From PB.7 or PF.14 Pin Interrupt Flag (Read Only)** 0 = External Interrupt from PB.7 or PF.14 interrupt is deasserted. 1 = External Interrupt from PB.7 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 or IRC Auto Trim Interrupt Flag (Read Only)** 0 = Clock fail detected or IRC Auto Trim interrupt is deasserted. 1 = Clock fail detected or IRC Auto Trim 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.

6.3.16 System Control Register

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

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

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
VTOR	SCS_BA+0xD08	R/W	Vector Table Offset 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	PENDSVCLR	PENDSTSET	PENDSTCLR	Reserved
23	22	21	20	19	18	17	16
ISRPREEMPT	ISRPENDING	Reserved				VECTPENDING	
15	14	13	12	11	10	9	8
VECTPENDING				RETTOBASE	Reserved		
7	6	5	4	3	2	1	0
Reserved		VECTACTIVE					

Bits	Description	
[31]	**NMIPENDSET**	**NMI Set-pending Bit** Write Operation: 0 = No effect. 1 = Changes NMI exception state to pending. Read Operation: 0 = NMI exception is not pending. 1 = NMI exception is pending. **Note:** Because NMI is the highest-priority exception, normally the processor enters the NMI exception handler as soon as it detects a write of 1 to this bit. Entering the handler then clears this bit to 0. This means a read of this bit by the NMI exception handler returns 1 only if the NMI signal is reasserted while the processor is executing that handler.
[30:29]	**Reserved**	Reserved.
[28]	**PENDSVSET**	**PendSV Set-pending Bit** Write Operation: 0 = No effect. 1 = Changes PendSV exception state to pending. Read Operation: 0 = PendSV exception is not pending. 1 = PendSV exception is pending. **Note:** Writing 1 to this bit is the only way to set the PendSV exception state to pending.
[27]	**PENDSVCLR**	**PendSV Clear-pending Bit** Write Operation: 0 = No effect. 1 = Removes the pending state from the PendSV exception. **Note:** This is a write only bit. To clear the PENDSV bit, you must “write 0 to PENDSVSET and write 1 to PENDSVRTC_CAL” at the same time.

[26]	**PENDSTSET**	**SysTick Exception Set-pending Bit** Write Operation: 0 = No effect. 1 = Changes SysTick exception state to pending. Read Operation: 0 = SysTick exception is not pending. 1 = SysTick exception is pending.
[25]	**PENDSTCLR**	**SysTick Exception Clear-pending Bit** Write Operation: 0 = No effect. 1 = Removes the pending state from the SysTick exception. **Note:** This is a write only bit. To clear the PENDST bit, you must "write 0 to PENDSTSET and write 1 to PENDSTRTC_CAL" at the same time.
[24]	**Reserved**	Reserved.
[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** Indicate the Exception Number of the Highest Priority Pending Enabled Exception 0 = no pending exceptions. Nonzero = the exception number of the highest priority pending enabled exception. The value indicated by this field includes the effect of the BASEPRI and FAULTMASK registers, but not any effect of the PRIMASK register.
[11]	**RETTOBASE**	**Preempted Active Exceptions Indicator** Indicate whether There are Preempted Active Exceptions 0 = there are preempted active exceptions to execute. 1 = there are no active exceptions, or the currently-executing exception is the only active exception.
[10:6]	**Reserved**	Reserved.
[5:0]	**VECTACTIVE**	**Number of the Current Active Exception** 0 = Thread mode. Non-zero = The exception number of the currently active exception.

Vector Table Offset Register (VTOR)

Register	Offset	R/W	Description	Reset Value
VTOR	SCS_BA+0xD08	R/W	Vector Table Offset Register	0x0000_0000

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

Bits	Description	
[31:7]	**TBLOFF**	**Table Offset Bits** The vector table address for the selected Security state.
[6:0]	**Reserved**	Reserved.

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 6.3-10 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 = Enters sleep, or deep sleep, on return from an ISR to Thread mode. Setting this bit to 1 enables an interrupt driven application to avoid returning to an empty main application.
[0]	**Reserved**	Reserved.

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							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PRI_5							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PRI_4							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**PRI_6**	Priority of system handler 6, UsageFault
[15:8]	**PRI_5**	Priority of system handler 5, BusFault
[7:0]	**PRI_4**	Priority of system handler 4, MemManage

System Handler Priority Register 2 (SHPR2)

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

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

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

System Handler Priority Register 3 (SHPR3)

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

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

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

6.4 Flash Memory Controller (FMC)

6.4.1 Overview

This chip is equipped with 16/32/64/128/256/512 Kbytes on-chip embedded Flash (the chip with 512 Kbytes consists of two 256 Kbytes BANK0 and BANK1). A User Configuration block is provided for system initialization. A loader ROM (LDROM) is used for In-System-Programming (ISP) function. A security protection ROM (SPROM) can conceal user program. For M031xG/I and M032xG/I, a 4 Kbytes cache with zero wait cycle is implemented to improve the performance of code/data fetching. This chip also supports In-Application-Programming (IAP) function. User switches the code executing without the chip reset after the embedded Flash is updated. Refer to Table 6.4-1 for detailed differences between each M031/M032 series.

6.4.2 Features

- Supports dual-bank Flash macro for safe firmware upgrade.
- Supports OTA function.
- Supports APROM bank erase.
- Supports 16/32/64/128/256/512 Kbytes application ROM (APROM).
- Supports 512 bytes page size for 16/32/64/128 Kbytes Flash.
- Supports 2048 bytes page size for 256/512 Kbytes Flash.
- Supports 2/4/8 Kbytes loader ROM (LDROM).
- Supports configurable Data Flash size to share with APROM.
- Supports 512/2048 bytes security protection ROM (SPROM) to conceal user program.
- Supports 12 bytes User Configuration block to control system initialization.
- Supports 512/2048 bytes page erase for all embedded Flash.
- Supports CRC-32 checksum calculation 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.

Section	Sub-Section	M031xB/C/D/E M032xB/C/D/E	M031xG/I M032xG/I
6.4.4 Functional Description	6.4.4.3 Physical and Virtual Address Concept	-	-/●
	6.4.4.4 APROM Reboot Address Operation Model Selection	-	●
	6.4.4.14 Cache Memory Controller	-	●
	6.4.4.15 Embedded Flash Memory Programming 64-bit Programming and Multi-word Programming	-	●
	6.4.4.17 Flash All-One Verification	-	●

Section	Sub-Section	M031xB/C/D/E M032xB/C/D/E	M031xG/I M032xG/I
	ISP Control Register (FMC_ISPCTL) INTEN (FMC_ISPCTL[24])	-	•
6.4.6 Register Description	Flash Access Time Control Register (FMC_FTCTL) CACHEINV (FMC_FTCTL[9])	-	•
	Flash Access Time Control Register (FMC_FTCTL) FOM (FMC_FTCTL[6:4])	000 = Frequency is less than or equal to 48 MHz. 001 = Frequency is less than or equal to 24 MHz. Others = Reserved.	001 = Frequency is less than or equal to 12 MHz. 010 = Frequency is less than or equal to 36 MHz. 011 = Frequency is less than or equal to 60 MHz. Others = Frequency is less than or equal to 72 MHz.
	ISP Status Register FBS (FMC_ISPSTS[30])	-	•
	ISP Status Register (FMC_ISPSTS) INTFLAG (FMC_ISPSTS[8])	-	•
	ISP Status Register (FMC_ISPSTS) PGFF (FMC_ISPSTS[5])	-	•
	ISP Data 0 Register (FMC_MPDAT0)	-	•
	ISP Data 1 Register (FMC_MPDAT1)	-	•
	ISP Data 2 Register (FMC_MPDAT2)	-	•
	ISP Data 3 Register (FMC_MPDAT3)	-	•
	ISP Multi-word Program Address Register (FMC_MPADDR)	-	•
	ISP Multi-program Status Register (FMC_MPSTS)	-	•

Table 6.4-1 FMC Features Comparison Table at Different Chip

6.4.3 Block Diagram

The Flash memory controller (FMC) consists of AHB slave interface, Flash control registers, cache memory controller (for M031xG/I and M032xG/I only), Flash initialization controller, Flash operation control and embedded Flash memory. Figure 6.4-1 shows the block diagram of Flash memory controller.

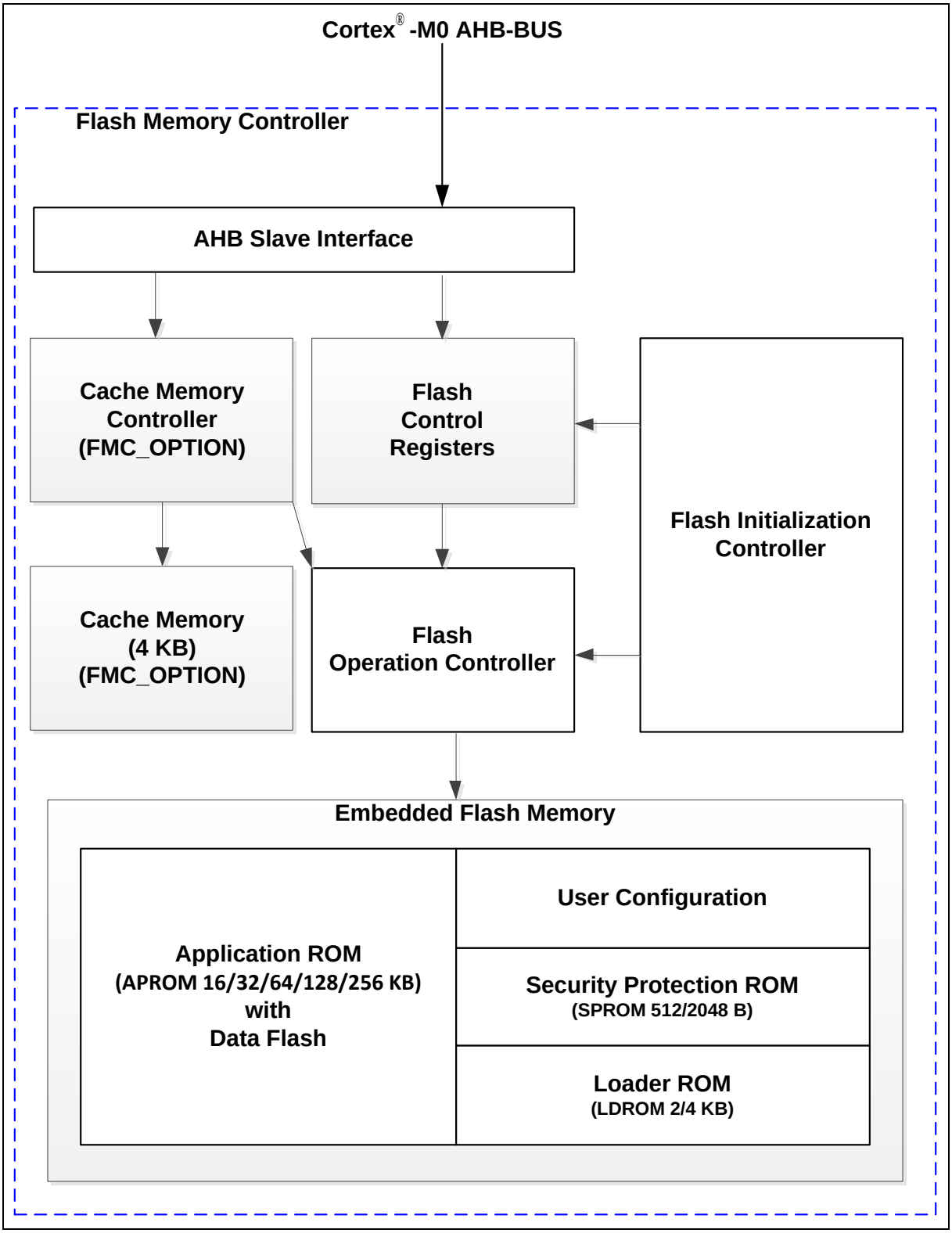


Figure 6.4-116/32/64/128/256 KB Flash Memory Control Block Diagram

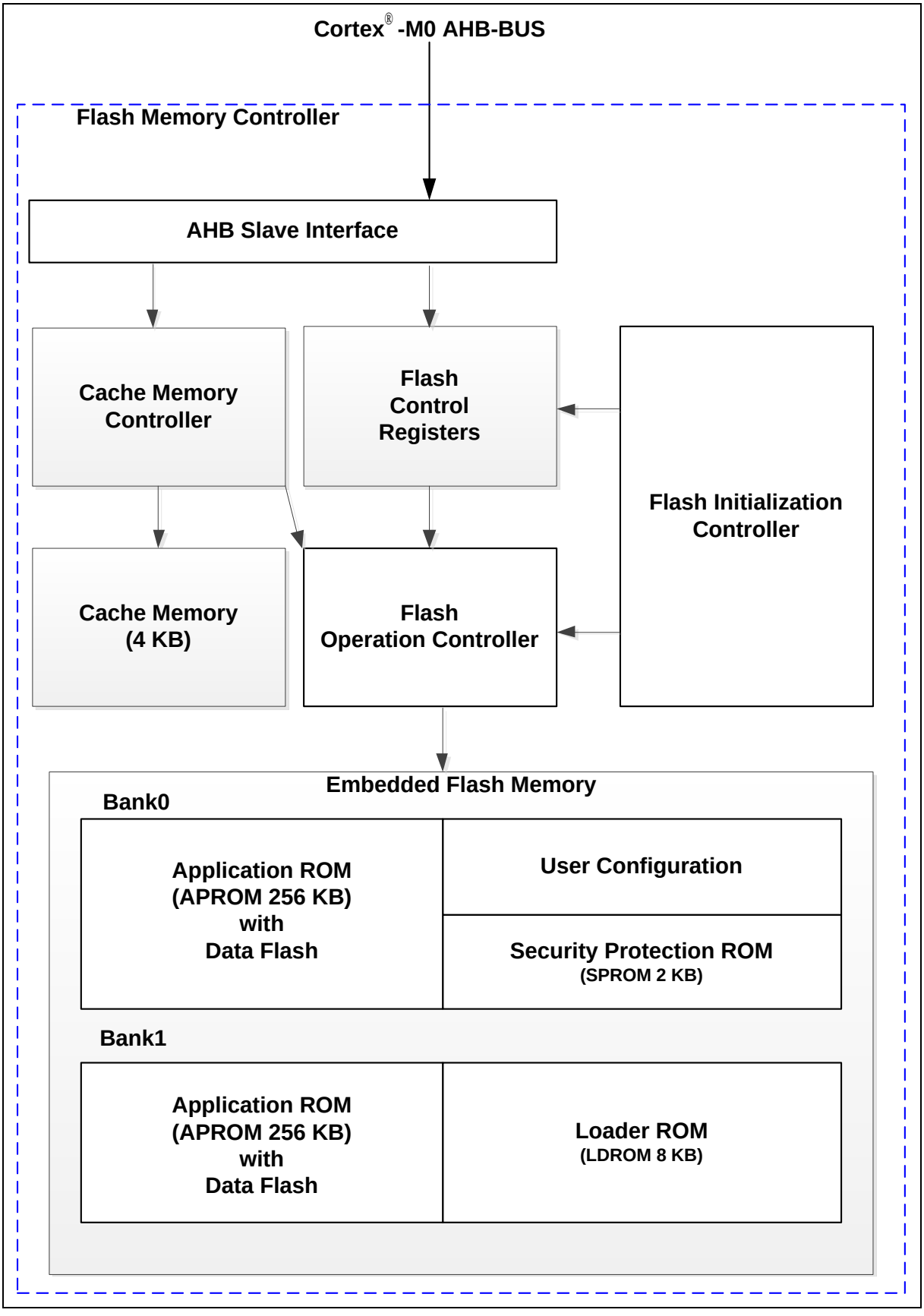

Cortex® -M0 AHB-BUS
Flash Memory Controller
AHB Slave Interface
Cache Memory Controller
Flash Control Registers
Flash Initialization Controller
Cache Memory (4 KB)
Flash Operation Controller
Embedded Flash Memory
Bank0
Application ROM (APROM 256 KB) with Data Flash
User Configuration
Security Protection ROM (SPROM 2 KB)
Bank1
Application ROM (APROM 256 KB) with Data Flash
Loader ROM (LDROM 8 KB)

Figure 6.4-2512 KB Flash Memory Control Block Diagram

AHB Slave Interface

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

Flash Control Registers

All of ISP control and status registers are in the Flash control registers. Refer to the Register Description section for the detailed register description.

Cache Memory Controller

A 4 Kbytes cache with zero wait cycle is implemented between Cortex®-M0 CPU and embedded Flash memory. This cache memory controller improves the performance of fetching code/data and reduces power consumption of the embedded Flash memory.

Flash Initialization Controller

When the 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 as system initialization stage.

Flash Operation Controller

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

Embedded Flash Memory

The embedded Flash memory is the main memory for user application code and parameters. It consists of the user configuration block, 2/4/8 Kbytes LDROM, 512/2048 bytes SPROM and 16/32/64/128/256/512 Kbytes APROM with Data Flash. The page erase Flash size is 512/2048 bytes, and program bit width is 32 bits.

6.4.4 Functional Description

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

6.4.4.1 Memory Organization

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

Bank	Flash Memory Block	Address Range
0	APROM with 256 KB	0x00_0000 ~ 0x03_ffff
	User Configuration	0x30_0000 ~ 0x30_000b
	SPROM	0x20_0000 ~ 0x20_07ff
1	APROM with 256 KB	0x04_0000 ~ 0x07_ffff
	LDROM	0x10_0000 ~ 0x10_1fff

Table 6.4-2 Dual Bank Block Address Range

6.4.4.2 LDROM, APROM and Data Flash

LDROM is designed for a loader to implement In-System-Programming (ISP) function by user. LDROM is a 2/4/8 KB embedded Flash memory, the Flash address range is from 0x0010_0000 to 0x0010_07FF/0x0010_0FFF/0x0010_1FFF. APROM is main memory for user applications. The APROM size is 16/32/64/128/256/512 KB. Data Flash is used to store application parameters (not instruction). Data Flash is shared with APROM and its size is configurable. The base address of Data Flash is determined by DFBA (CONFIG1[19:0]). All of embedded Flash memory is 512/2048 bytes page erased.

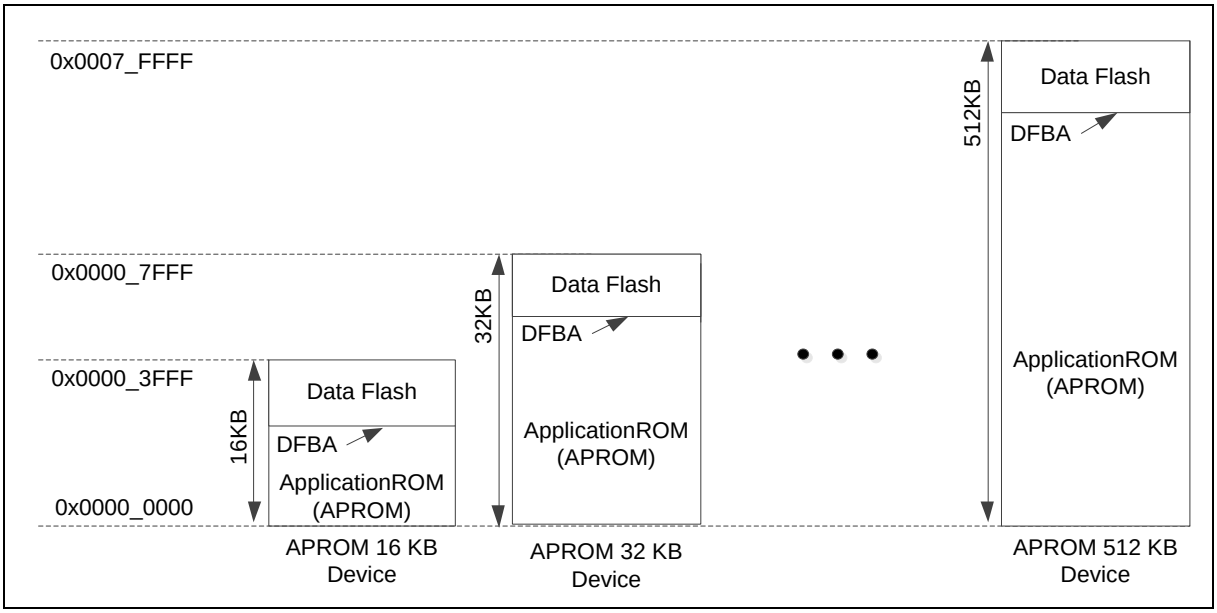


Figure 6.4-3 Data Flash Shared with APROM

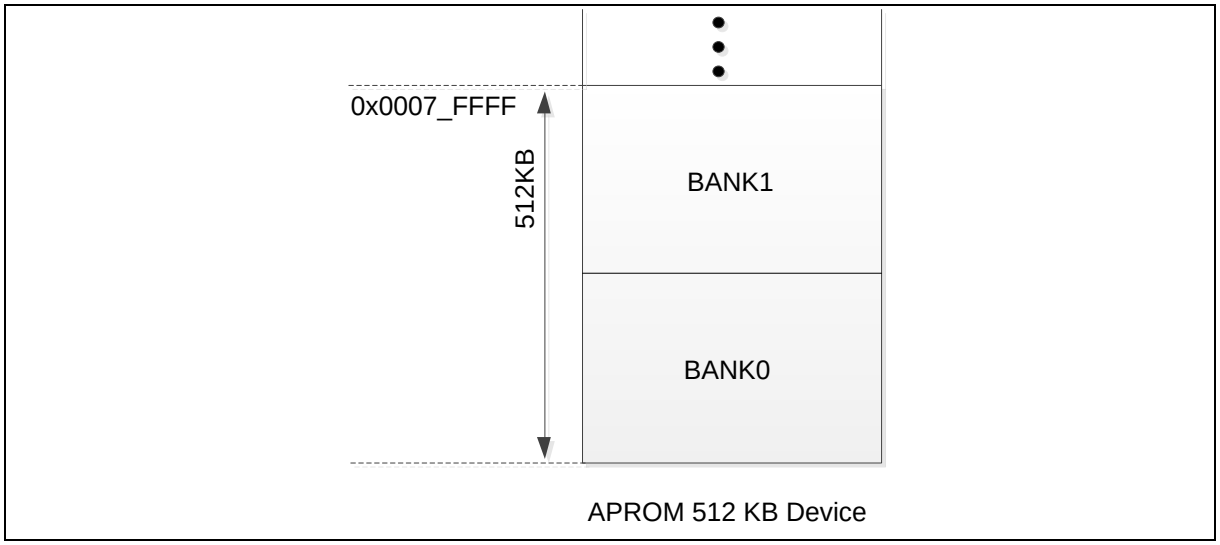


Figure 6.4-4 APROM with Dual Bank Examples (512 KB)

6.4.4.3 Physical and Virtual Address Concept

For OTA application, the memory space concept is a little different form FMC commands and CPU

viewpoint.

- From FMC command viewpoint, physical address concept was adopted. All of FMC commands regard APROM memory as physical address except “remapping” command. It means APROM Flash memory will be flat memory model no matter memory remapping state.
- From Data Flash viewpoint, virtual address concept was adopted.
- From CPU viewpoint, virtual address concept was adopted. APROM memory space is separated into 2 segments (banks). As runtime stage, the virtual address depends on what address operation model (OP0, OP1) is, shown in Figure 6.4-5.

6.4.4.4 APROM Reboot Address Operation Model Selection

To activate APROM reboot with different address operation model, use address operation model selection command 0x2C with FMC_ISPDAT “0x5AA5_5AA5” and FMC_ISPADDR ”0x0”(Address OP0) or ”0x1”(Address OP1) to select address operation model.

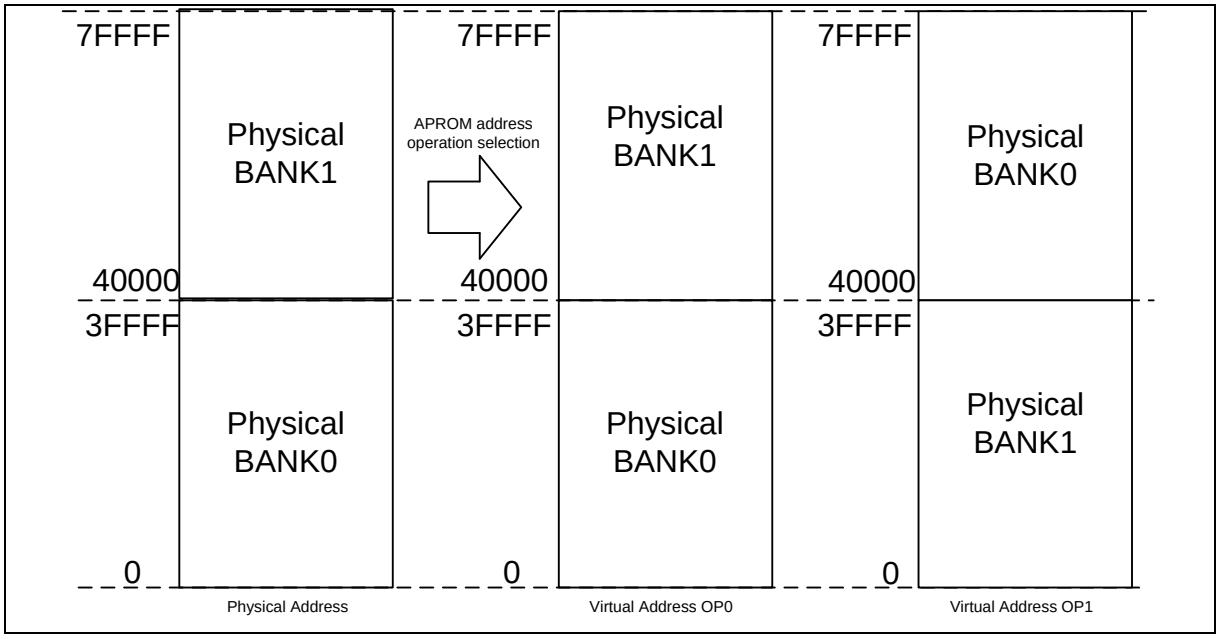


Figure 6.4-5 Address Operation Model

6.4.4.5 User Configuration Block

User Configuration block is internal programmable configuration area for boot options, such as Flash security lock, boot selection, 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 requirements. User Configuration block can be updated by ISP function and its address located at 0x0030_0000 with three 32 bits words (CONFIG0, CONFIG1 and CONFIG2). Any change on User Configuration block will take effect after system reboot

CONFIG0 (Address = 0x0030_0000)

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

Bits	Description	
[31]	**CWDTEN[2]**	**Watchdog Timer Hardware Enable Bit** When the 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 in normal operation mode. However, in Power-down mode, the LIRC may be able to be disabled by setting CWDTPDEN=1 and LIRCEN=0 (CLK_PWRCTL[3]). **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, WDT clock source only can be changed in this case. Others = WDT hardware enable function is active. WDT clock is always on.
[30]	**CWDTPDEN**	**Watchdog Clock Power-down Enable Bit** This bit should be used with CWDTEN. When WDT enabled by CWDTEN, User can use this bit to control WDT wakeup when system is in Power-down mode. If it is necessary to wakeup system by WDT, then we can set CWDTPDEN=0 to make sure WDT keep working at Power-down mode. If we don't want to wakeup system by WDT, we may just set CWDTPDEN=1 and LIRCEN=0 to let WDT suspend in power down. 0 = Watchdog Timer clock kept enabled when chip enters power-down. 1 = Watchdog Timer clock is controlled by LIRCEN (CLK_PWRCTL[3]) when the chip enters power-down. **Note:** This bit only works if CWDTEN[2:0] is set to 011.
[29:28]	**Reserved**	Reserved.
[27]	**CFGXT1**	**HXT Mode Selection** 0 = HXT works as external clock mode. PF.3 is configured as external clock input pin. 1 = HXT works as crystal mode. PF.2 and PF.3 are configured as external high speed crystal(HXT) pins. **Note:** When CFGXT1 = 0, PF.3 MFP should be set as GPIO mode. The DC characteristic of XT1_IN is the same as GPIO.
[26:22]	**Reserved**	Reserved.

[21]	**CBOV**	**Brown-out Voltage Selection** 0 = Brown-out voltage is 2.0V. 1 = Brown-out voltage is 2.5V.
[20]	**CBORST**	**Brown-out Reset Enable Bit** 0 = Brown-out reset Enabled after power on or active from reset pin. 1 = Brown-out reset Disabled after power on or active from reset pin.
[19]	**CBODEN**	**Brown-out Detector Enable Bit** 0= Brown-out detect Enabled after powered on. 1= Brown-out detect Disabled after powered on.
[18:13]	**Reserved**	Reserved.
[12]	**ICELOCK**	**ICE Lock Bit** This bit is only used to disable ICE function. User may use it with LOCK (CONFIG0[1]) bit to increase security level. 0 = ICE function Disabled. 1 = ICE function Enabled.
[11]	**Reserved**	Reserved.
[10]	**CIOINI**	**I/O Initial State Selection** 0 = All GPIO set as Quasi-bidirectional mode after chip powered on or active from reset pin. 1 = All GPIO set as input tri-state mode after powered on or active from reset pin.
[9]	**RSTEXT**	**Chip Reset Time Extend** 0 = Extend reset time to 26.6m s if chip release from power-on reset/LVR/BOD/RST pin reset happened. 1 = Extend reset time to 3.2ms if chip release from power-on reset/LVR/BOD/RST pin reset happened.
[8]	**RSTWSEL**	**RST Pin Width Selection** 0 = RST pin debounce width is 2us. 1 = RST pin debounce width is 32us.
[7:6]	**CBS**	**Chip Booting Selection** When CBS[0] = 0 with IAP mode, 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, the code in LDROM and APROM can be called by each other. CBS value is valid. 00 = Boot from LDROM with IAP mode. 01 = Boot from LDROM without IAP mode. 10 = Boot from APROM with IAP mode. 11 = Boot from APROM without IAP mode. **Note:** BS (FMC_ISPCTL[1]) is only be used to control boot switching when CBS[0] = 1. VECMAP (FMC_ISPSTS[23:9]) is only be used to remap 0x0~0x1ff when CBS[0] = 0.
[5]	**Reserved**	Reserved.
[4:3]	**CWDTEN[1:0]**	**Watchdog Timer Hardware Enable Bit** Please refer to CWDTEN[2] (CONFIG0[31]) for details.
[2]	**Reserved**	Reserved.

[1]	**LOCK**	**Security Lock Control** 0 = Flash memory content is locked. 1 = Flash memory content is unlocked if ALOCK (CONFIG2[7:0]) is also equal to 0x5A.
[0]	**DFEN**	**Data Flash Enable Bit** The Data Flash is shared with APROM, and the base address of Data Flash is decided by DFBA (CONFIG1[19:0]) when DFEN is 0. 0 = Data Flash Enabled. 1 = Data Flash Disabled.

CONFIG1 (Address = 0x0030_0004)

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

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19:0]	**DFBA**	**Data Flash Base Address** This register works only when DFEN (CONFIG0[0]) is set to 0. If DFEN (CONFIG0[0]) is set to 0, the Data Flash base address is defined by user. Since on-chip Flash erase unit is 512 or 2048 bytes, it is mandatory to keep bit 8-0 or bit 10-0 respectively as 0.

CONFIG2 (Address = 0x0030_0008)

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**ALOCK**	**Advance Security Lock Control** 0x5A = Flash memory content is unlocked if LOCK (CONFIG0[1]) is set to 1. Others = Flash memory content is locked. **Note:** ALOCK will be programmed as 0x5A after executing ISP page erase or ISP/ICP whole chip erase

6.4.4.6 Security Protection Memory (SPROM)

The security protection memory (SPROM) is used to store instructions for security application. The SPROM includes 512/2048 bytes at location address 0x20_0000 ~ 0x20_01FF/0x20_07FF and does not support “whole chip erase command”. Figure 6.4-6 shows that the last byte of SPROM (address: 0x0020_01FF) is used to identify the SPROM code is non-secured, debug secured or secured mode. Last byte of SPROM (address: 0x0020_07FF) is used for 256/512 KB Flash case.

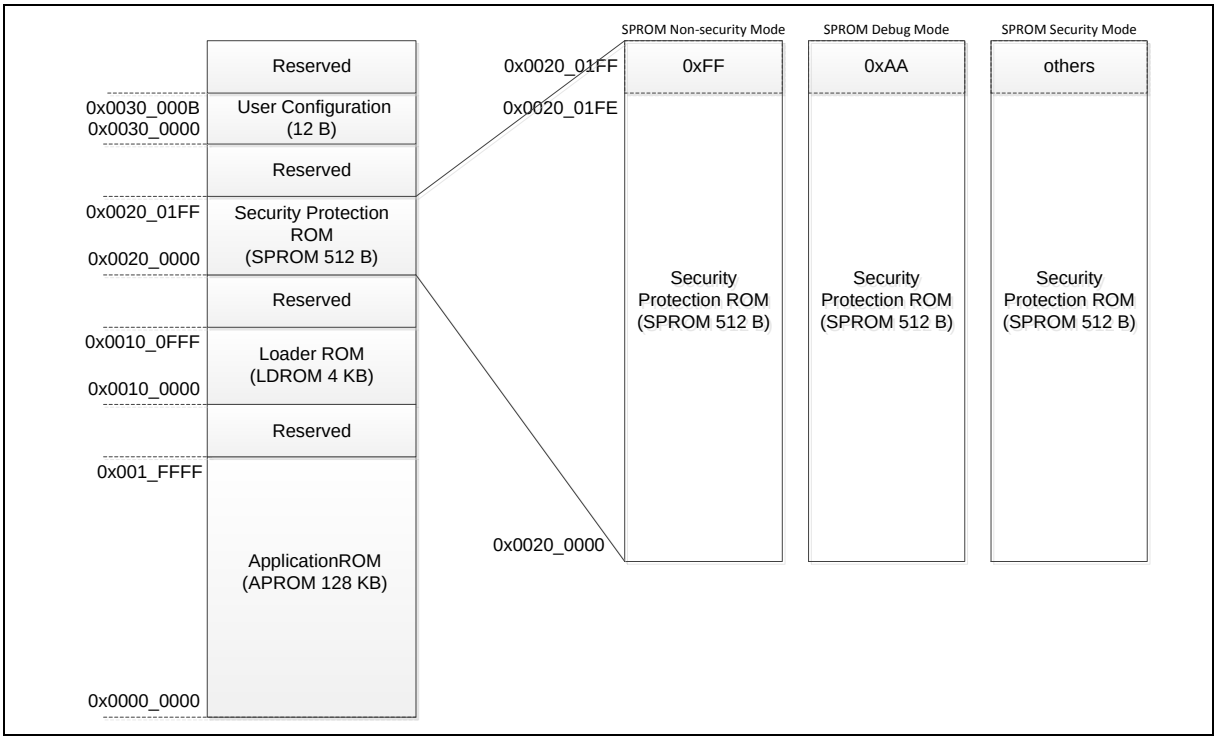


Figure 6.4-616/32/64/128 Kbytes Flash SPROM Security Mode

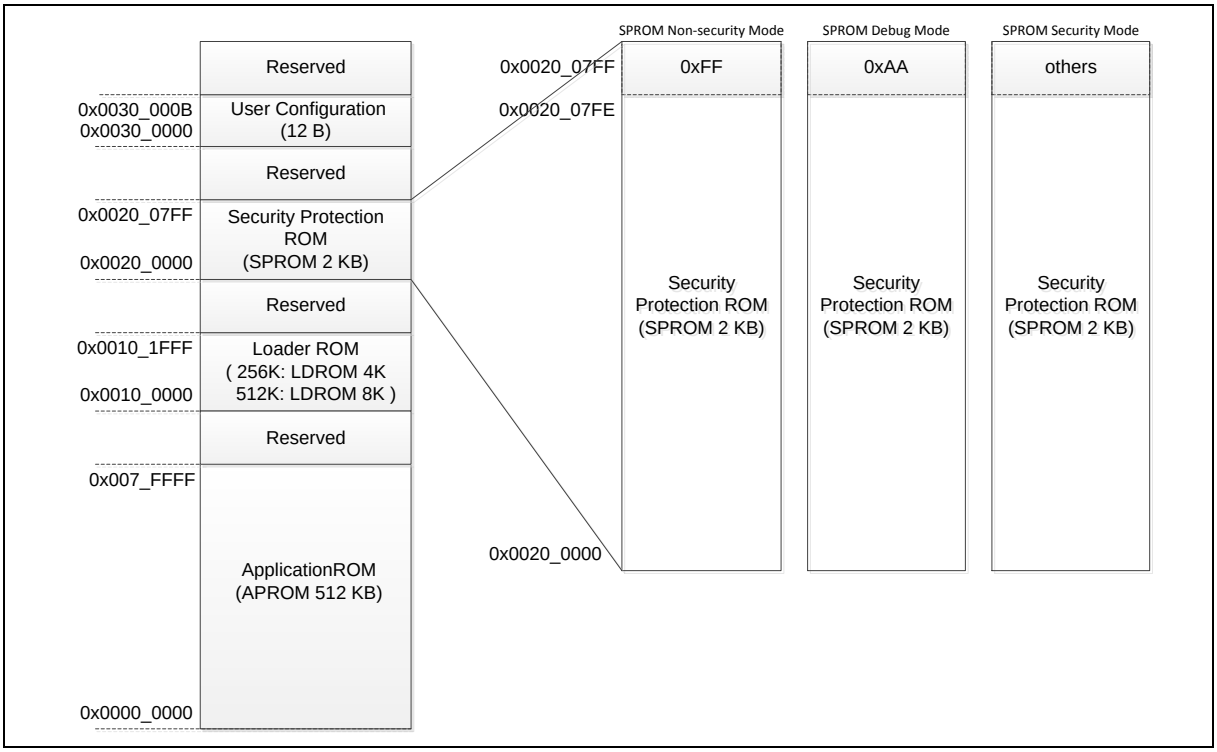


Figure 6.4-7256/512 Kbytes Flash SPROM Security Mode

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

erased and programmed by ISP command.

2. SPROM debug secured mode (the last byte is 0xAA). In order to debug easily, FMC controller accepts to execute program of SPROM when Cortex®-M0 ICE (In-Circuit-Emulator) port is connected. Other behaviors of SPROM are the same with SPROM secured mode.
3. SPROM secured mode (the last byte is not 0xFF or 0xAA). In order to conceal SPROM code in secured mode, CPU only can perform instruction fetch and get data from SPROM when CPU is run at SPROM area. Otherwise, CPU will get all zero (0x0000_0000) for data access. If user uses ICE tool to debug system. In order to protect SPROM, CPU fetches code from secured SPROM area will cause ICE disconnection. At this mode, SPROM doesn't support ISP program and read Flash command and only supports page erase command with 0x0055AA03 data.
4. The SCODE (FMC_ISPSTS[31]) is SPROM secured flag to indicate that SPROM keeps secured mode, debug secured mode or not. It is set to 1 at Flash initialization if the last byte of SPROM isn't 0xFF, and can be cleared after the SPROM page erase operation complete. In order to easily test SPROM secured mode at normal run, it also can be set to 1 by user if the last byte of SPROM is 0xFF.

6.4.4.7 Flash Memory Map

In the M031/M032 series, the Flash memory map is different from system memory map. The system memory map is used by CPU fetch code or data from FMC memory. The Flash memory map is used for ISP function to read, program or erase FMC memory. Figure 6.4-8 shows the Flash memory map.

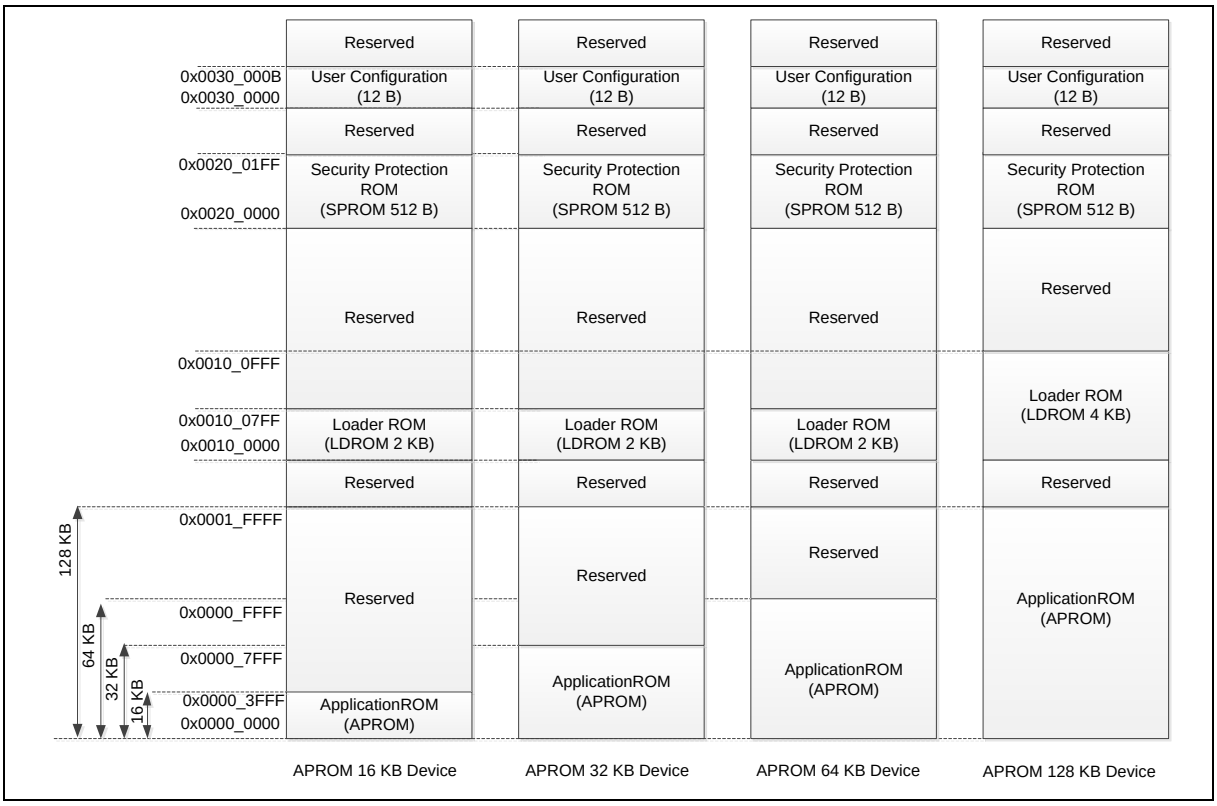


Figure 6.4-816/32/64/128 Kbytes Flash Memory Map

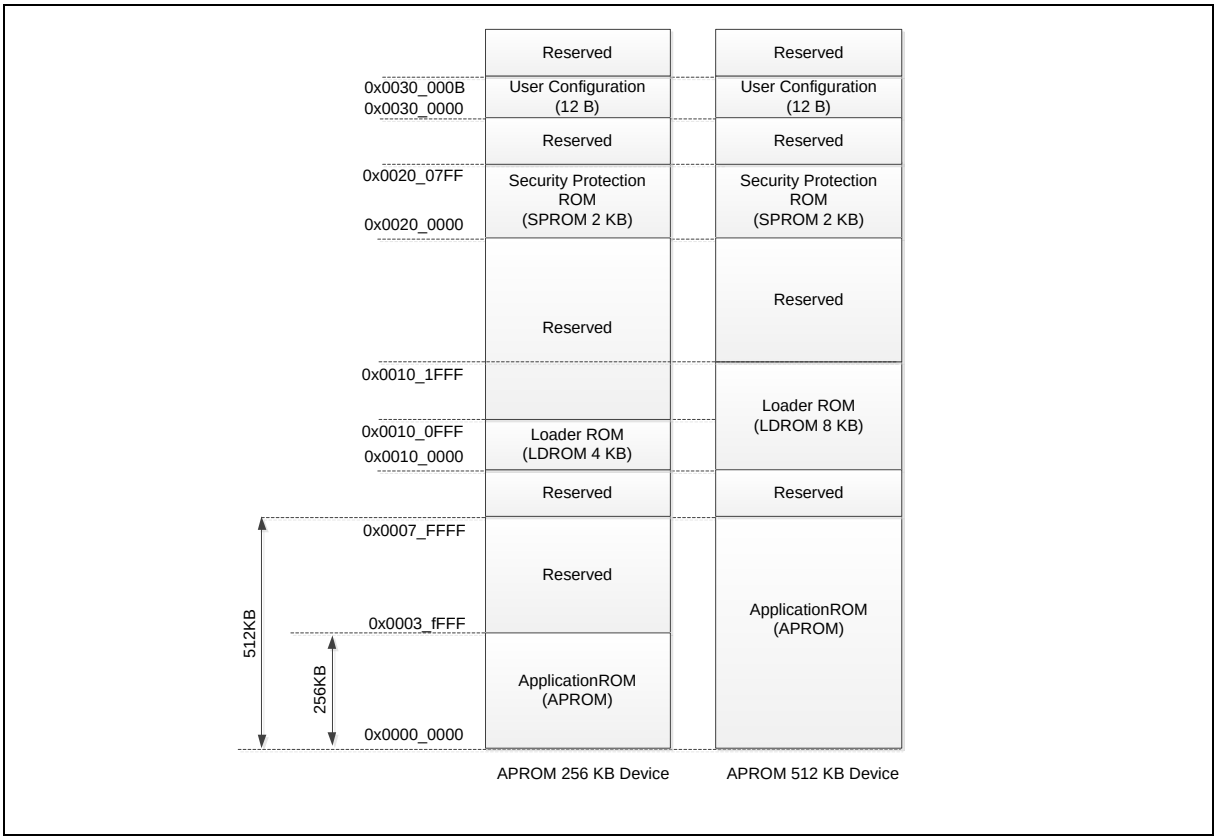


Figure 6.4-9256/512 Kbytes Flash Memory Map

6.4.4.8 System Memory Map with IAP Mode

The system memory map is used by CPU to fetch code or data from FMC memory. SPROM(0x0020_0000~0x0020_01FF/0x0020_07FF) and LDROM(0x0010_0000~0x0010_07FF/0x0010_0FFF/0x0010_1FFF) address map are the same as in the Flash memory map. The Data Flash is shared with APROM and the Data Flash base address is defined by CONFIG1. The content of CONFIG1 is loaded into DFBA (Data Flash Base Address Register) at the Flash initialization. The DFBA~APROM maximum size is the Data Flash region for Cortex®-M0 data access, and 0x0000_0200~(DFBA-1) is APROM region for Cortex®-M0 instruction access.

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

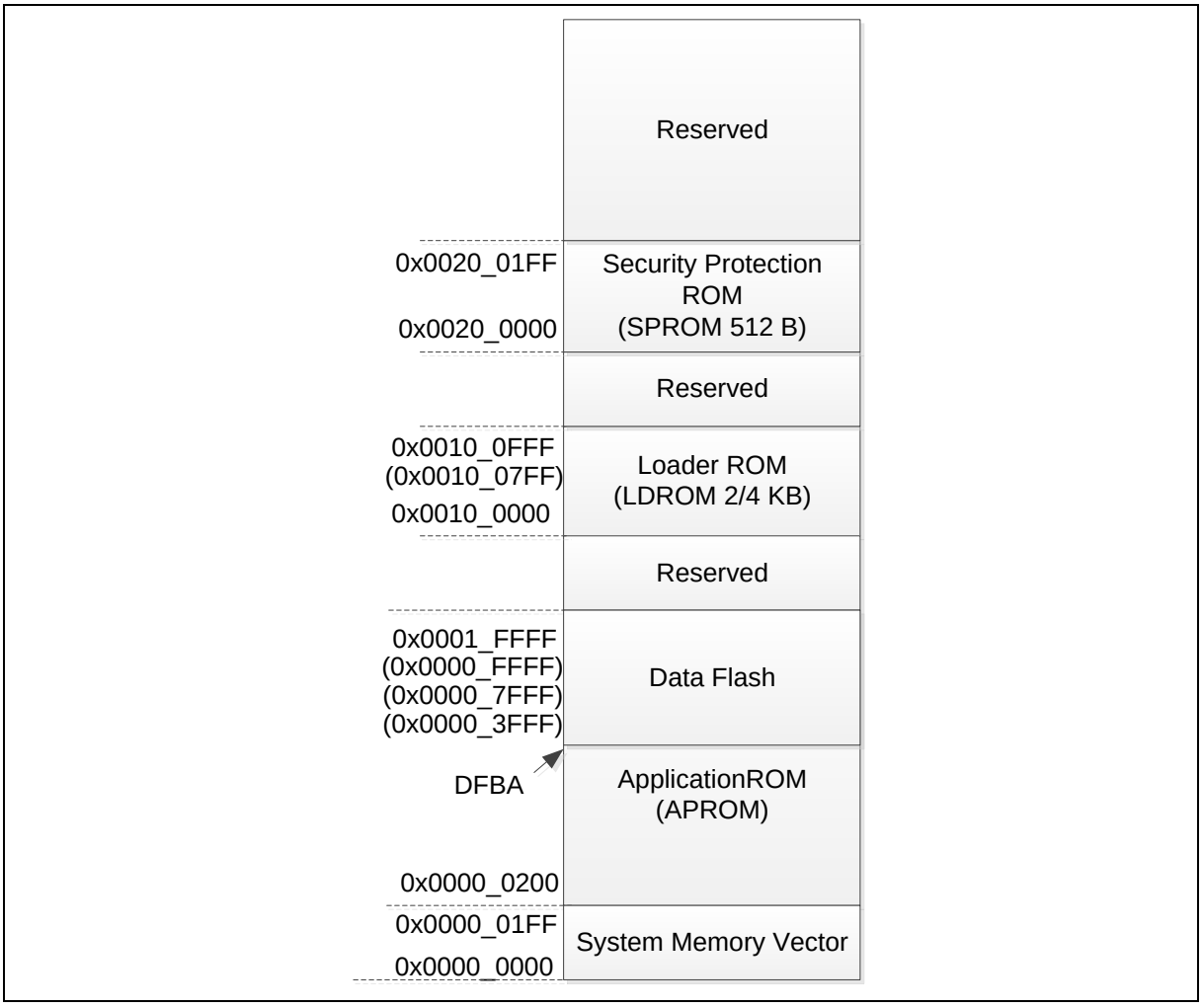


Figure 6.4-1016/32/64/128 Kbytes Flash System Memory Map with IAP Mode

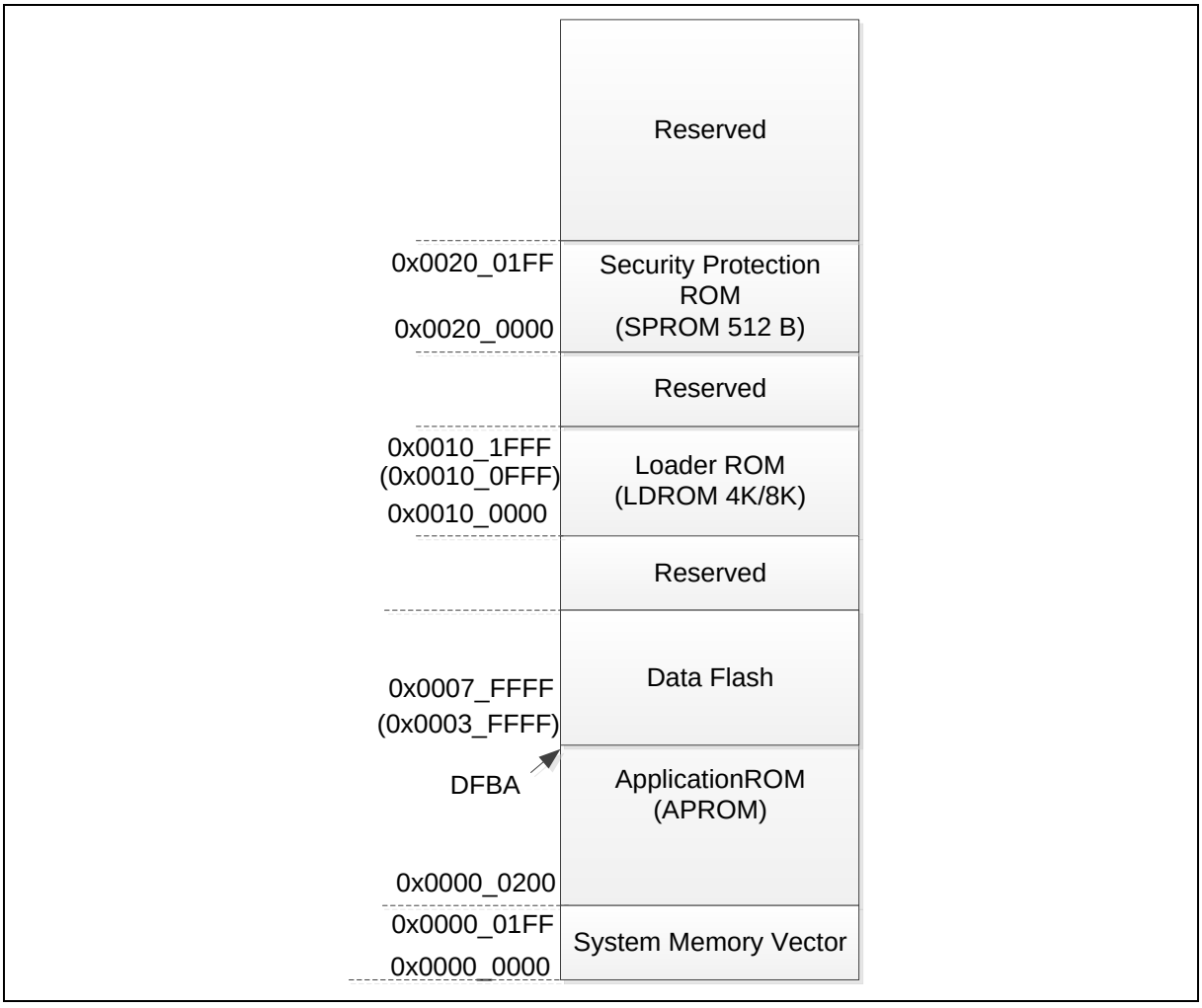


Figure 6.4-11 256/512 Kbytes Flash System Memory Map with IAP Mode

In LDROM with IAP mode, the default value of {VECMAP[11:0], 9'h000} is 0x100000 and first page of LDROM (0x0010_0000 ~ 0x0010_01FF) is mapping to the system memory vector for Cortex®-M0 instruction or data access.

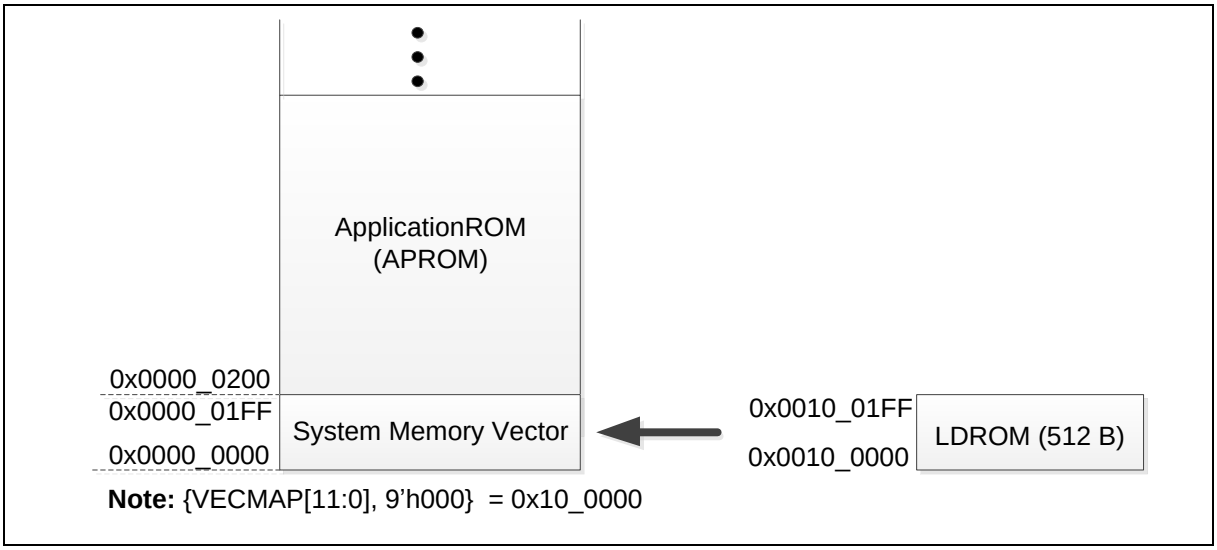


Figure 6.4-12 LDROM with IAP Mode

In APROM with IAP mode, the default value of {VECMAP[11:0], 9'h000} is 0x000000 and first page of APROM (0x0000_0000~0x0000_01FF) is mapping to the system memory vector for Cortex®-M0 instruction or data access.

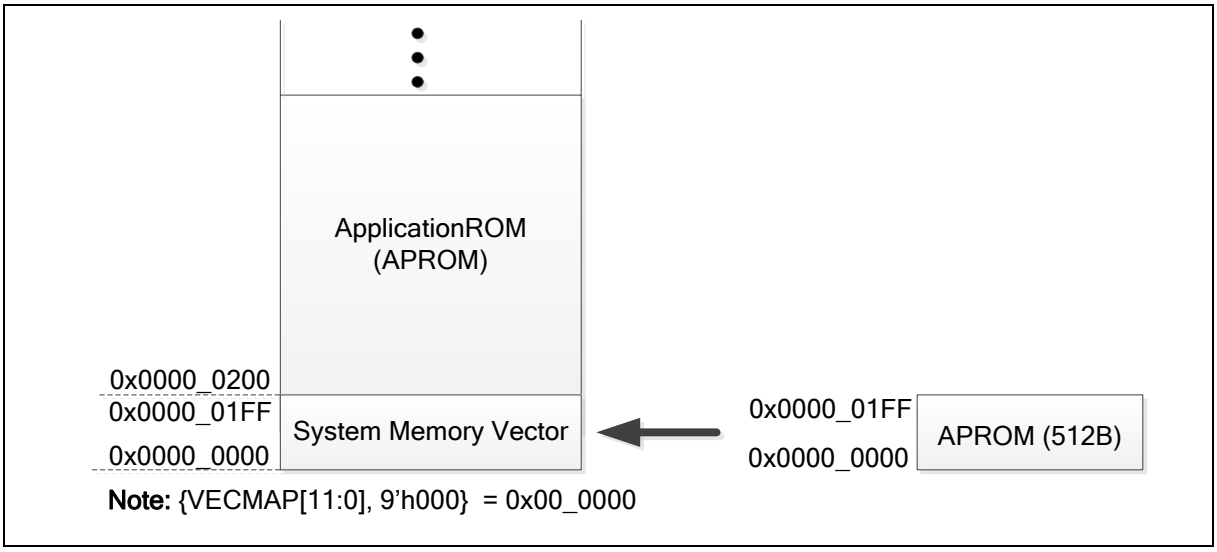


Figure 6.4-13 APROM with IAP Mode

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 Page Remap" command (0x2E). In VECMAP (FMC_ISPSTS[23:9]), shows the finial system memory vector mapping address.

6.4.4.9 System Memory Map without IAP Mode

In system memory map without IAP mode, CPU still can access SPROM(0x0020_0000~0x0020_01FF/0x0020_07FF), but the system memory vector mapping is not supported. There are two kinds of system memory map without IAP mode when chip booting: (1) LDROM without IAP, (2) APROM without IAP. In LDROM without IAP mode, LDROM base is mapping to 0x0000_0000. CPU program cannot run to access APROM. In APROM without IAP mode, APROM

base is mapping to 0x0000_0000. CPU program cannot run to access LDROM. The Data Flash is shared with APROM and the Data Flash base address is defined by CONFIG1. The content of CONFIG1 is loaded into DFBA (Data Flash Base Address Register) at the Flash initialization. The DFBA~0x0000_7FFF is the Data Flash region for Cortex®-M0 data access, and 0x0000_0000~(DFBA-1) is APROM region for Cortex®-M0 instruction access.

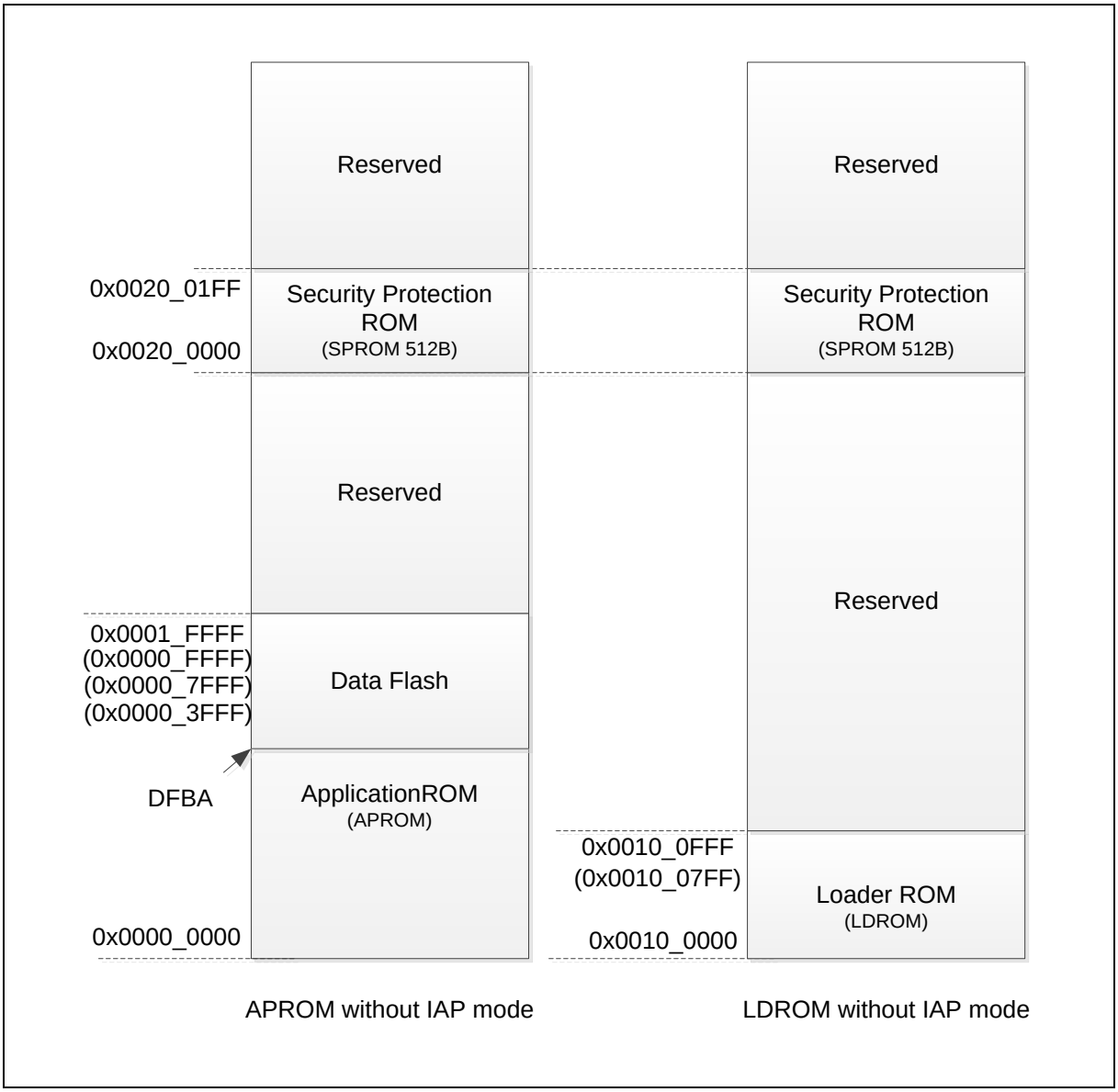


Figure 6.4-1416/32/64/128 Kbytes Flash System Memory Map without IAP Mode

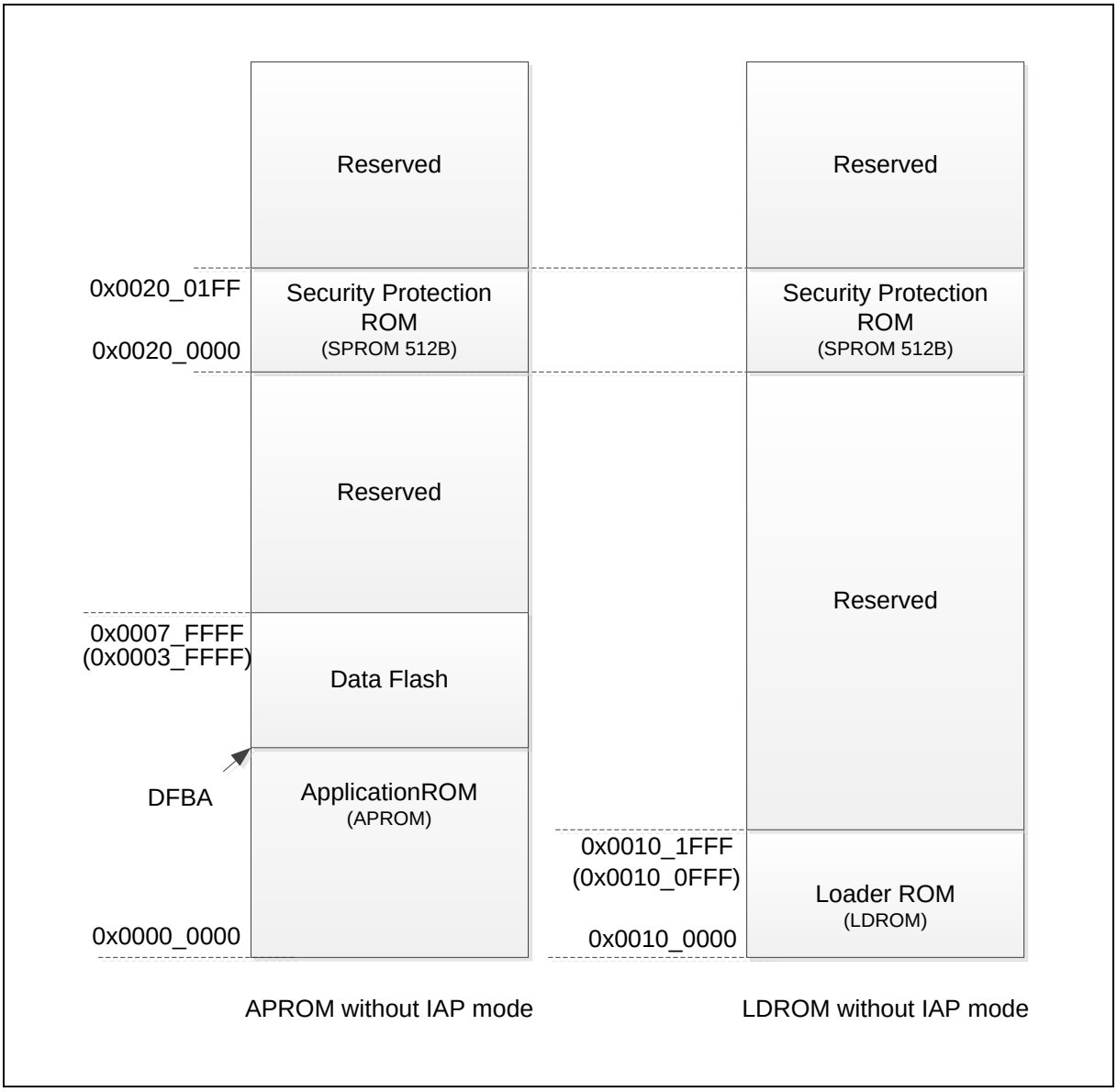


Figure 6.4-15 256/512 Kbytes Flash System Memory Map without IAP Mode

6.4.4.10 Boot Selection

The M031/M032 series provides four booting modes for application field. They are LDROM with IAP, LDROM without IAP, APROM with IAP, and APROM without IAP. The booting modes and system memory map are setting by CBS (CONFIG0[7:6]).

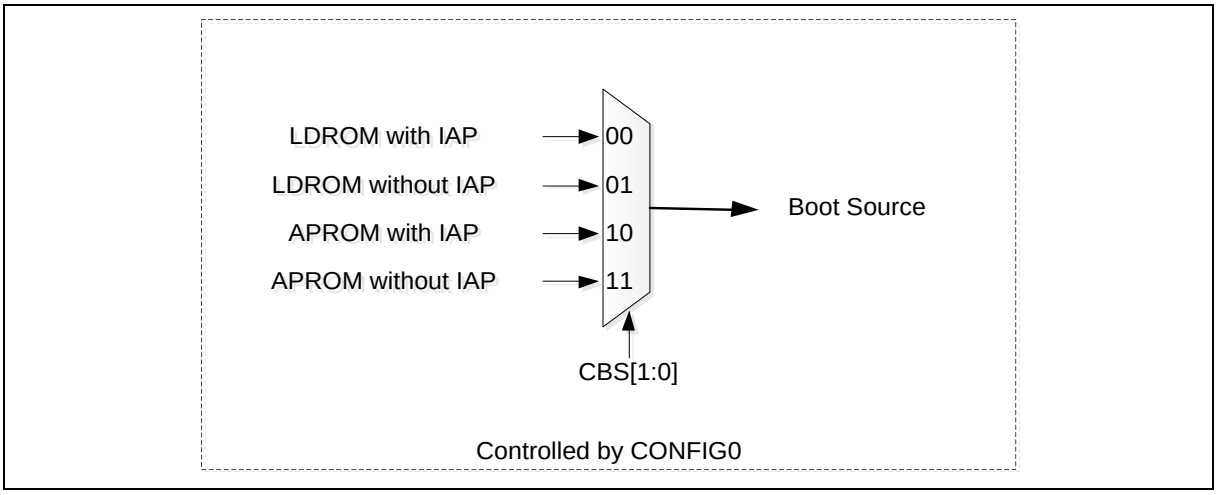


Figure 6.4-16 Boot Source Selection

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

Table 6.4-3 Vector Mapping Support

6.4.4.11 In-Application-Programming (IAP)

The M031/M032 series provides In-Application-Programming (IAP) function for user to switch the code executing between APROM, LDROM and SPROM. User can enable the IAP function by booting chip and setting the chip boot selection bits in CBS (CONFIG0[7:6]) as 10 or 00.

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

6.4.4.12 In-System-Programming (ISP)

The M031/M032 series supports In-System-Programming (ISP) function allowing the embedded Flash memory to be reprogrammed under software control. ISP is performed without removing the microcontroller from the system through the firmware and on-chip connectivity interface, such as UART, I^2C, and SPI.

The M031 ISP provides the following functions for embedded Flash memory.

- Supports Flash page erase function
- Supports Flash APROM bank erase function
- Supports Flash data program function
- Supports Flash data read function
- Supports company ID read function

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

ISP Commands

ISP Command	FMC_ISPCMD	FMC_ISPADDR	FMC_ISPDAT FMC_MPDAT0~ MPDAT3
FLASH Page Erase	0x22	Valid address of Flash memory origination. It must be 512/2048 bytes page alignment.	N/A
SPROM Page Erase	0x22	0x0020_0000	0x0055_AA03
FLASH APROM BANK Erase	0x23	Valid address of Flash APROM origination.	N/A
FLASH 32-bit Program	0x21	Valid address of Flash memory origination	FMC_ISPDAT :Programming Data
FLASH 64-bit Program	0x61	Valid address of Flash memory organization. It must be 64-bit alignment.	FMC_ISPDAT :N/A FMC_MPDAT0: LSB Programming Data FMC_MPDAT1: MSB Programming Data FMC_MPDAT2~FMC_MPDAT3: N/A
FLASH Multi-Word Program	0x27	Valid address of Flash memory organization. It must be 32-bit alignment.	FMC_ISPDAT :N/A FMC_MPDAT0: 1'st Programming Data FMC_MPDAT1: 2'nd Programming Data FMC_MPDAT2: 3'rd Programming Data FMC_MPDAT3: 4'th Programming Data
FLASH 32-bit Read	0x00	Valid address of Flash memory origination. It must be word alignment.	FMC_ISPDAT: Return Data
FLASH 64-bit Read	0x40	Valid address of Flash memory organization. It must be 64-bit alignment.	FMC_ISPDAT: LSB Return Data FMC_MPDAT0: LSB Return Data FMC_MPDAT1: MSB Return Data FMC_MPDAT2~FMC_MPDAT3: N/A
Read Company ID	0x0B	0x0000_0000	FMC_ISPDAT: 0x0000_00DA
APROM Address Operation Model Selection	0x2C	0x0000_0000 (address OP0) : BANK0 virtual address range 0x00000~0x3FFFF BANK1 virtual address range 0x40000~0x7FFFF 0x0000_0001 (address OP1) : BANK0 virtual address range 0x40000~0x7FFFF BANK1 virtual address range	0x5AA5_5AA5

		0x00000~0x3FFFF	
Read Checksum	0x0D	Keep address of “Run Checksum Calculation”	FMC_ISPDAT: Return Checksum
Run Checksum Calculation	0x2D	Valid start address of memory origination It must be 512 bytes page alignment	FMC_ISPDAT: Size It must be 512 bytes 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 512 bytes page alignment	FMC_ISPDAT: Size It must be 512 bytes alignment
Read Unique ID	0x04	0x0000_0000	FMC_ISPDAT: Unique ID Word 0
		0x0000_0004	FMC_ISPDAT: Unique ID Word 1
		0x0000_0008	FMC_ISPDAT: Unique ID Word 2
		0x0000_0070	FMC_ISPDAT[11:0]: Built-in VBG ADC conversion result
Vector Remap	0x2E	Valid address in APROM or LDROM. It must be 512 bytes alignment	N/A

Table 6.4-4 ISP Command List

ISP Procedure

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

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

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

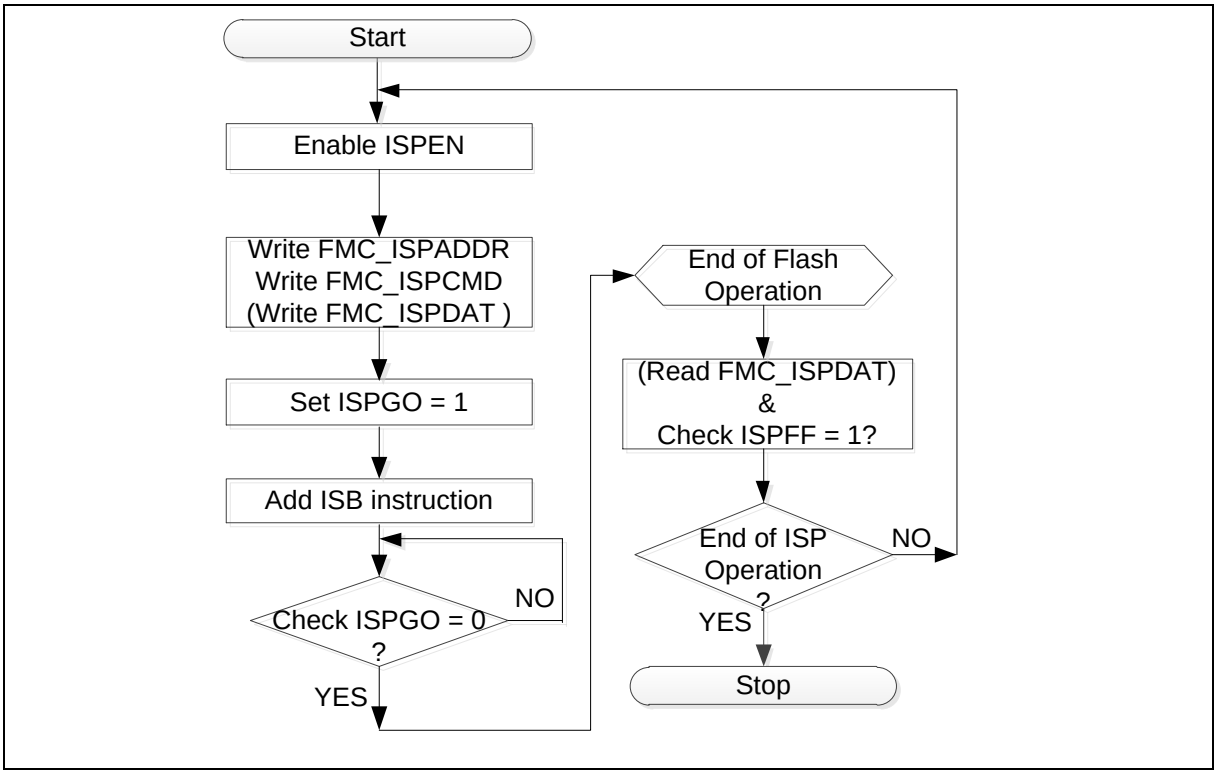


Figure 6.4-17 ISP Procedure Example

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

Several error conditions will be checked after ISP is completed. If an error condition occurs, ISP operation is not started and the ISP fail flag will be set instead. ISPFF(FMC_ISPSTS[6]) flag can only be cleared by software. 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, 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. User should add ISB (Instruction Synchronization Barrier) instruction next to the instruction in which ISPGO (FMC_ISPTRG[0]) bit is set 1 to ensure correct execution of the instructions following ISP operation.

6.4.4.13 VECMAP for Interrupt and Memory Programming

Accelerate interrupt by VECMAP

In IAP mode, VECMAP function could be used to map 512 bytes SRAM to vector map space. It means it is possible to store all exception vectors to SRAM. Then, if any exceptions assert, CPU can read exception handler from SRAM with zero wait state to speed up exception latency.

Because the vector map space is fixed to be 512 bytes, user must copy all 512 bytes to SRAM before remapping SRAM to vector map space. Otherwise, CPU may get wrong data from vector map space after remapping. Figure 6.4-18 shows an example to accelerating interrupt by VECMAP.

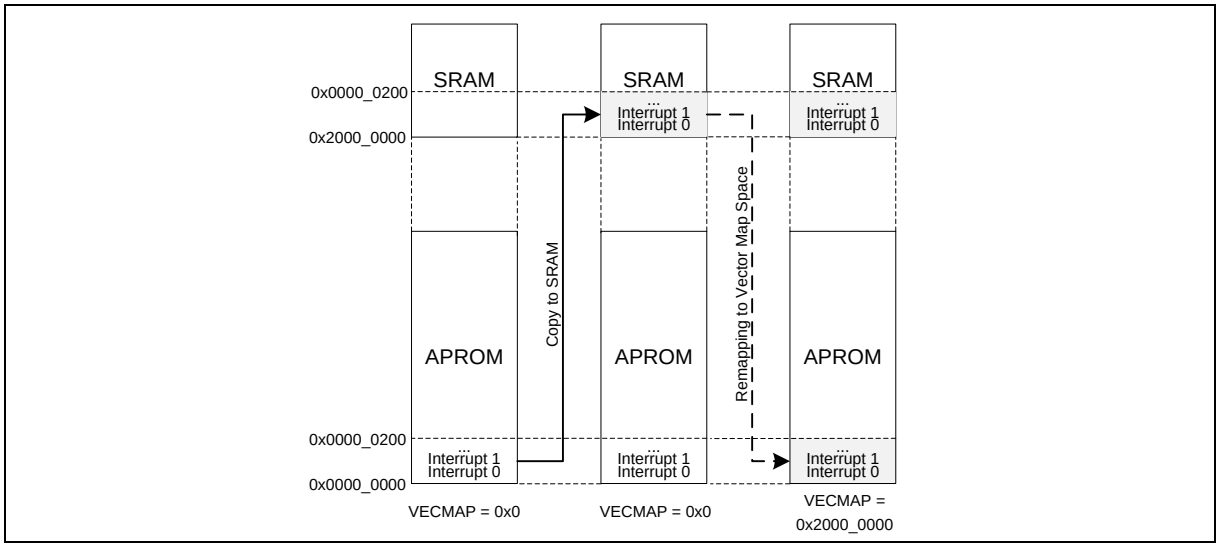


Figure 6.4-18 Example for Accelerating Interrupt by VECMAP

Avoid CPU halt when Flash programming

When Flash memory controller is busy, any CPU access to Flash memory will cause CPU halt for waiting Flash controller ready. If Flash controller is busy in page erasing, it may cause CPU halt for a long time to erase pages. To avoid this situation, user needs to avoid CPU access Flash memory when page erasing. The easiest way is to execute code in SRAM and use VECMAP to map all exceptions to SRAM. By executing code in SRAM, CPU will not access Flash to get instructions. By mapping all exceptions to SRAM, all interrupts will not need to get exception handler from Flash memory.

6.4.4.14 Cache Memory Controller

The cache memory controller will store each 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, 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 sucessfully. In such scenarios, it's necessary to manually send invalidation command (by writing CACHEINV in FMC_FTCTL) afterwards.

6.4.4.15 Embedded Flash Memory Programming

This chip provides 32-bit, 64-bit and multi-word Flash memory programming function to speed up Flash updated procedure. Table 6.4-5 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 CMD 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_MPSTS	ISP Multi-Program status	N/A	N/A	●
FMC_MPADDR	ISP Multi-Program Address	N/A	N/A	●

Table 6.4-5 FMC Control Registers for Flash Programming

64-bit Programming

The NuMicro® M031/M032 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 6.4-19 / Figure 6.4-20 shows ISP 32-bit / 64-bit programing procedure.

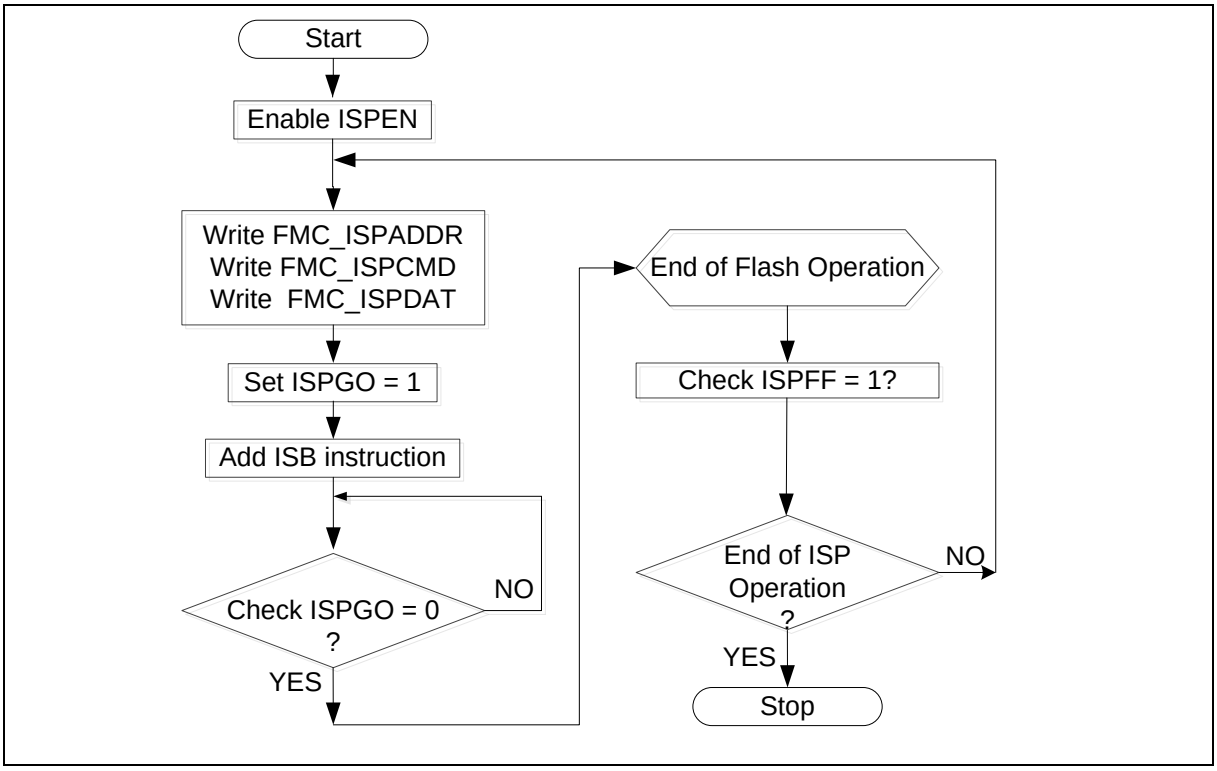


Figure 6.4-19 ISP 32-bit Programming Procedure

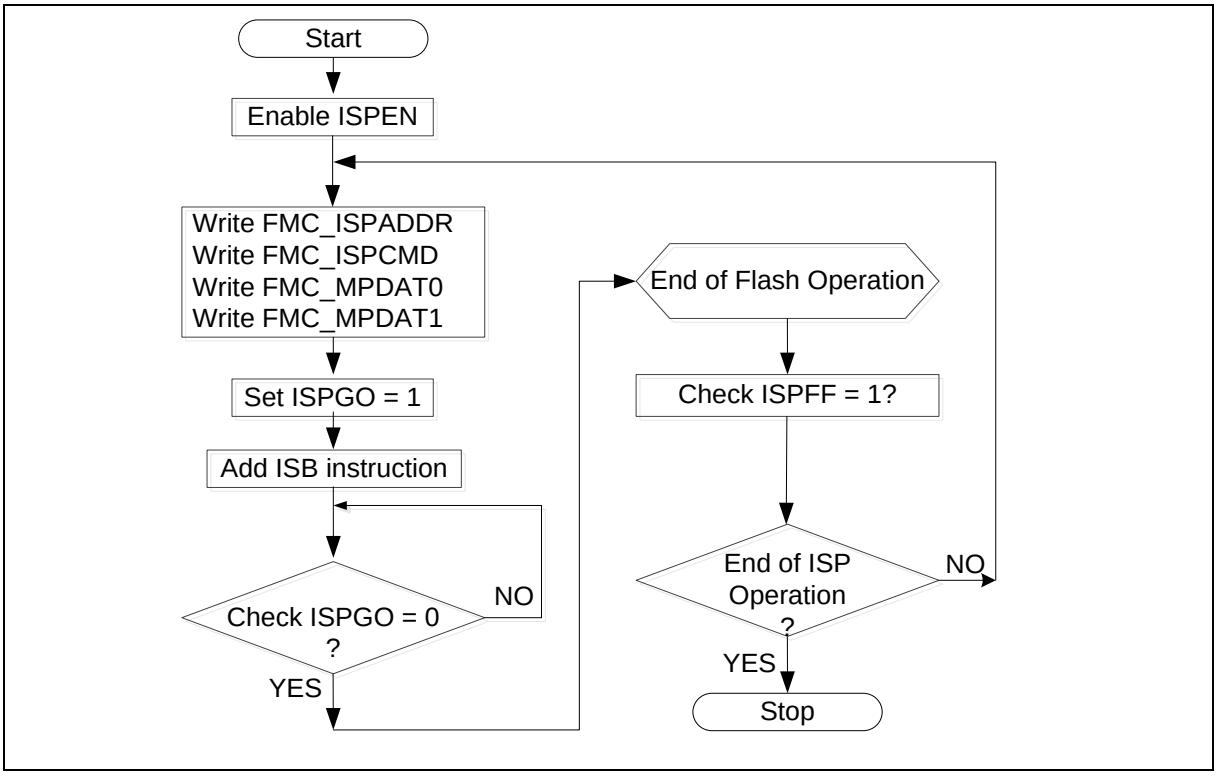


Figure 6.4-20 ISP 64-bit Programming Procedure

Multi-word Programming

The M031/M032 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.

In multi-word programming operation, CPU has to monitor the empty status of the programming buffer and prepare the next data for programming continuity. The multi-program firmware should not be located in APROM, LDROM or SPROM, because CPU instruction fetch cannot be hold during ISP processing. In other words, the multi-word programming code should executed in SRAM. The multi-word programming code also needs to make sure all exceptions will not access Flash memory when ISP processing. By the way, multi-word programming needs HCLK run up to 48 MHz.

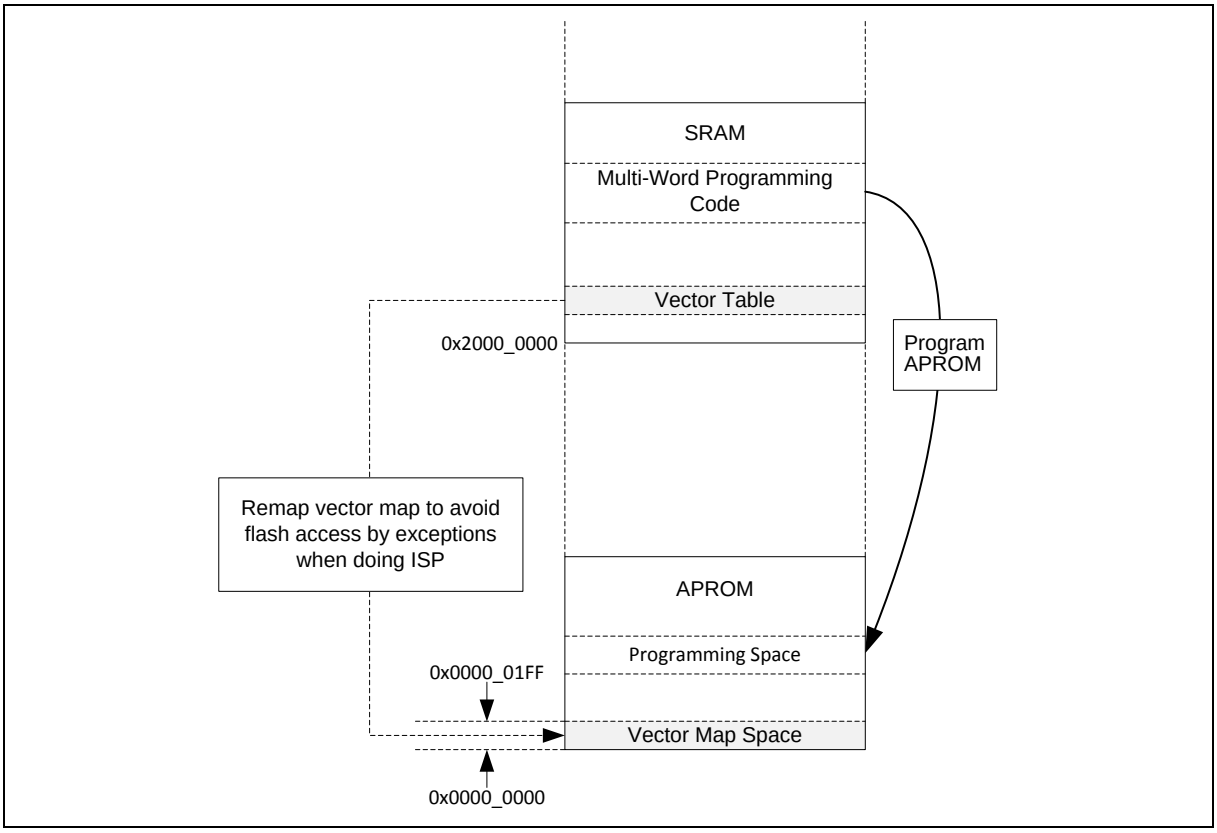


Figure 6.4-21 Firmware in SRAM for Multi-word Programming

The multi-word programming flow is shown as bellow. The starting ISP address (FMC_ISPADDR) has to be 8-byte align. FMC_MPDAT0 is the data word of the offset 0x0, FMC_MPDAT1 is the second word (offset 0x4), FMC_MPDAT2 is the third word (offset 0x8), and FMC_MPDAT3 is forth word (offset 0xC). If the starting ISP address FMC_ISPADDR [3] is 0, the 1st data word should put on FMC_MPDAT0, and 2nd word is FMC_MPDAT1, 3rd word is FMC_MPDAT2, and 4th word is FMC_MPDAT3. If the starting ISP address FMC_ISPADDR [3] is 1, the 1st data word should put on FMC_MPDAT2, and 2nd word is FMC_MPDAT3, 3rd word is FMC_MPDAT0, and 4th word is FMC_MPDAT1. The maximum programming size is 256 bytes. While FMC controller performs multi-word programming operation, CPU needs to monitor the buffer status D3~D0(FMC_MPSTS[7:4]) and MPBUSY (FMC_MPSTS[0]) to wait the buffer empty ((D1,D0)=00, or (D3,D2)=00), and then CPU needs to update the next programming data (FMC_MPDAT0, FMC_MPDAT1, FMC_MPDAT2 and FMC_MPDAT3) in time. Otherwise, FMC controller will exit multi-word programming operation (MPBUSY (FMC_MPSTS[0]) = 0). If CPU cannot update the data in time (MPBUSY (FMC_MPSTS[0]) =0), CPU needs restart a new multi-word programming procedure to continue, FMC_MPADDR provides the last program address information. At the end of operation, CPU has to check ISPFF (FMC_MPSTS[2]) to confirm the multi-word operation successful complete.

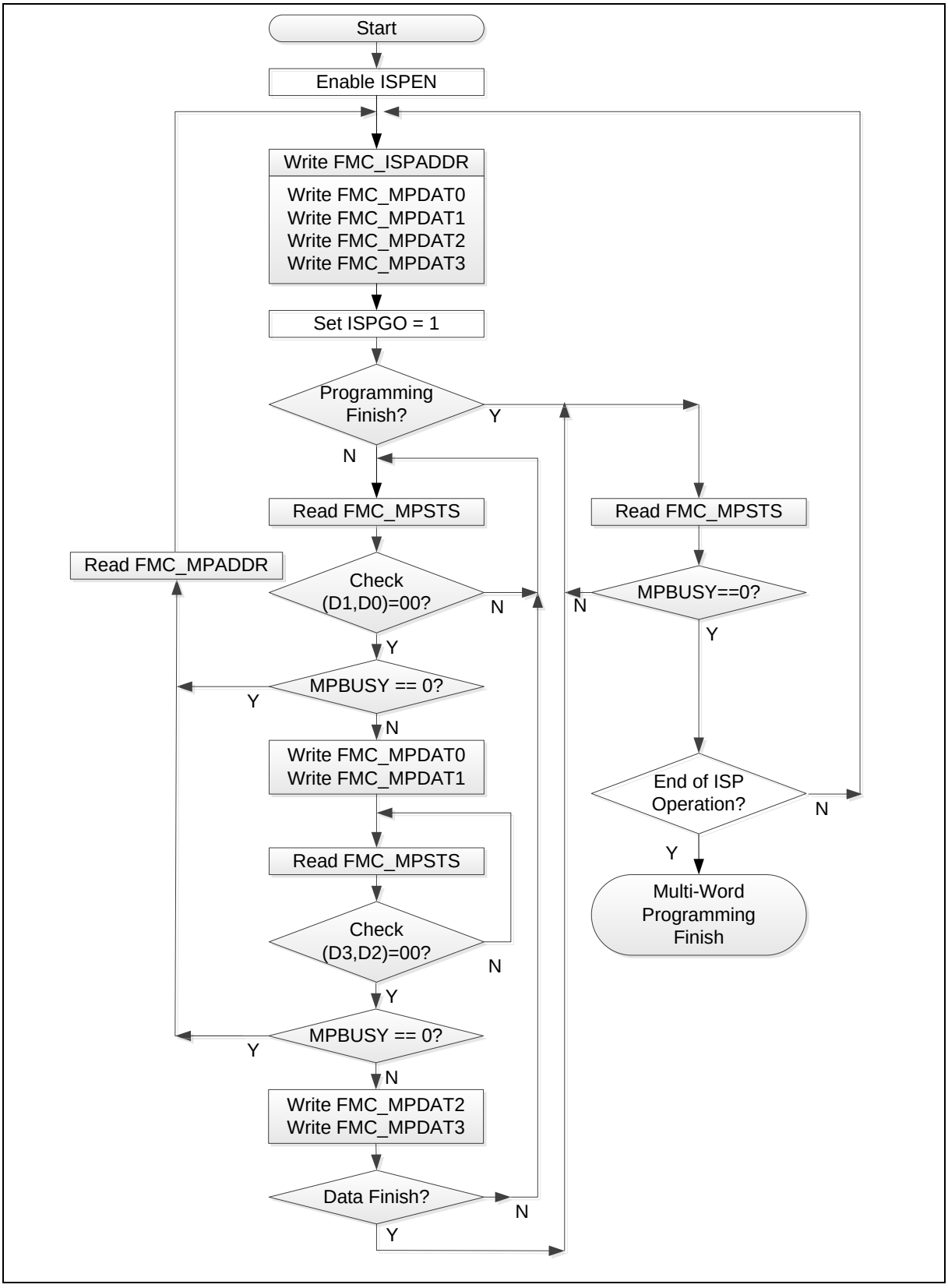


Figure 6.4-22 Multi-word Programming Flow

6.4.4.16 CRC32 Checksum Calculation

The M031/M032 series supports the Cyclic Redundancy Check (CRC-32) checksum calculation function to help user quickly check the memory content includes APROM, LDROM and SPROM. The CRC-32 polynomial is as below.

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

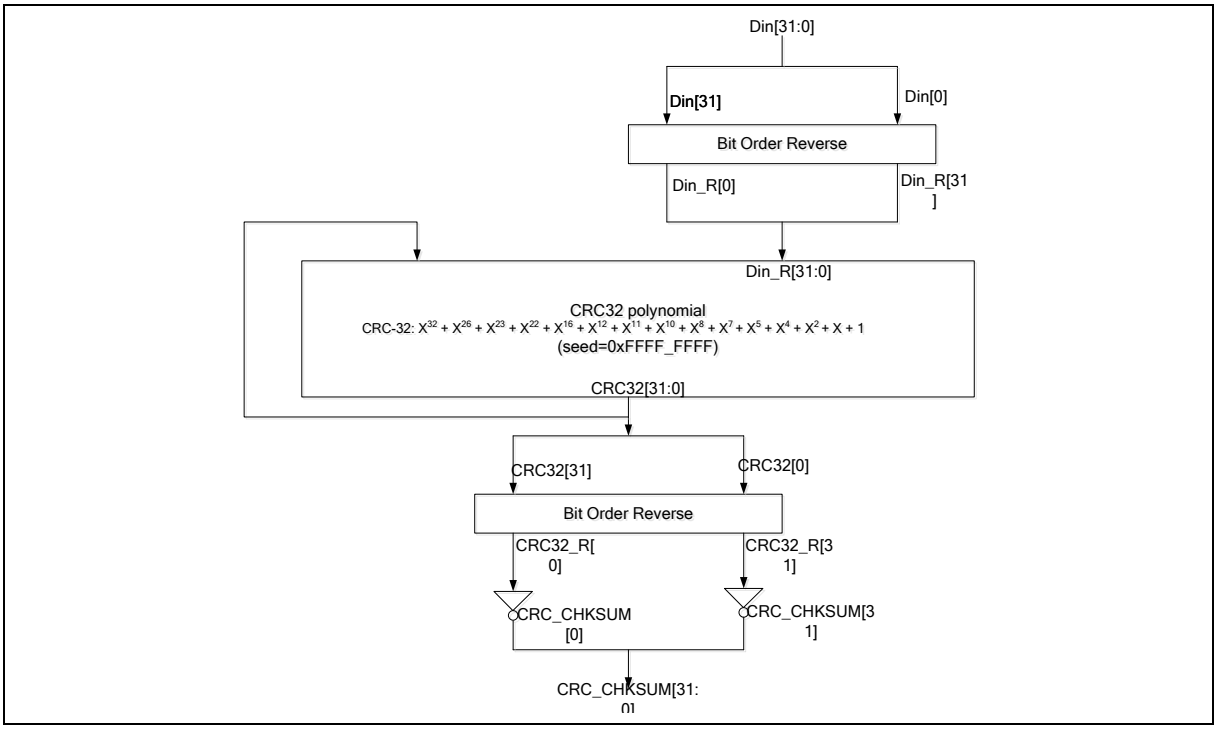


Figure 6.4-23 CRC-32 Checksum Calculation

The following three steps complete the CRC-32 checksum calculation.

1. Perform ISP “Run Memory Checksum” operation: user has to set the memory starting address (FMC_ISPADDR) and size (FMC_ISPDAT) to calculate. Both address and size have to be 512 bytes alignment, the size must be multiples of 512 bytes and the starting address includes APROM, LDROM and SPROM.
2. Perform ISP “Read Memory Checksum” operation: the FMC_ISPADDR should be kept the same as step 1.
3. Read FMC_ISPDAT to get checksum: The checksum is read from FMC_ISPDAT. If the checksum is 0x0000_0000, It must be one of two conditions (1) Checksum calculation is in-progress, (2) Address and size is over device limitation.

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

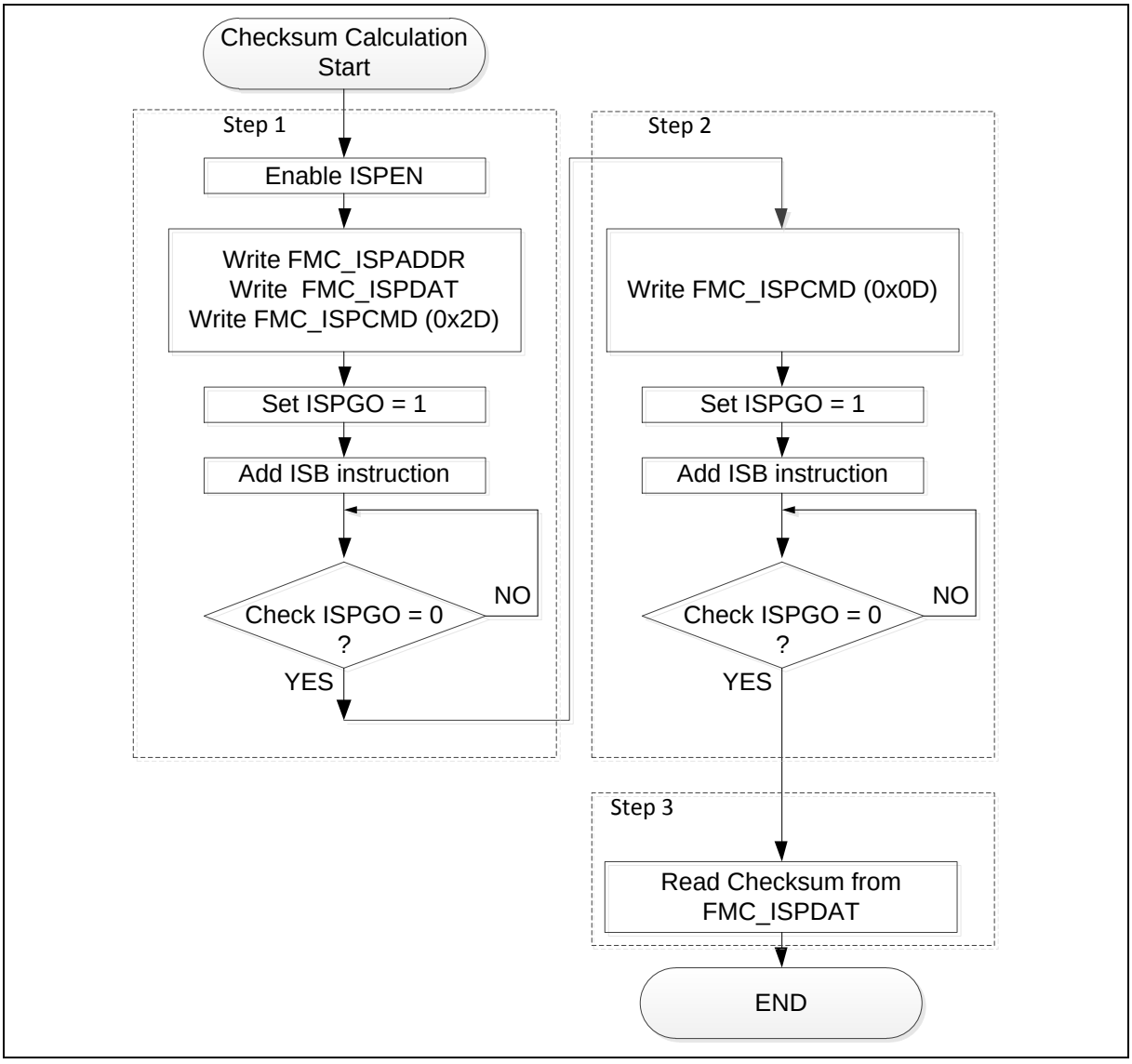


Figure 6.4-24 CRC-32 Checksum Calculation Flow

6.4.4.17 Flash All-One Verification

This chip 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.

The following two steps can be used to complete this Flash all-one verification.

1. Perform ISP “Run All One” command and check if the command is finished until FMC_ISPSTS[0] goes to 0.
2. Perform ISP “Read All One” command and check if the command is finished until FMC_ISPSTS[0] goes to 0. Following check FMC_ISPDAT content, if the value keeps 0, repeat step 2. If the specified Flash region all Flash are blank, the FMC_ISPDAT will report 0xA11FFFFF. Otherwise, FMC_ISPDAT will report 0xA1100000.

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 512 bytes alignment, the size should be ⩾ 512 bytes and the starting address includes APROM and LDROM.

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

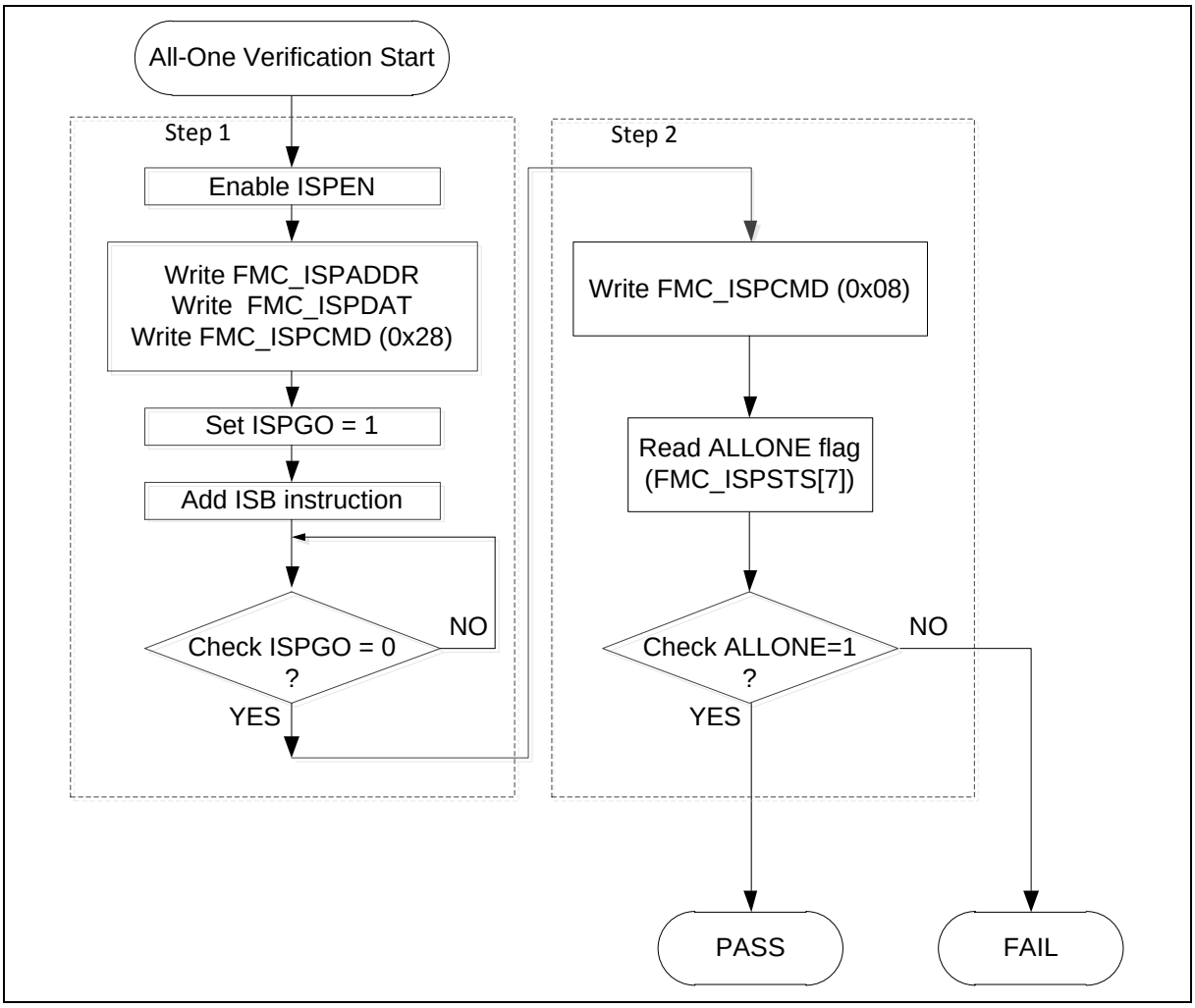


Figure 6.4-25 All-One Verification Flow

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_000X
FMC_ISPADDR	FMC_BA+0x04	R/W	ISP Address Register	0x0000_0000
FMC_ISPDAT	FMC_BA+0x08	R/W	ISP Data Register	0x0000_0000
FMC_ISPCMD	FMC_BA+0x0C	R/W	ISP Command Register	0x0000_0000
FMC_ISPTRG	FMC_BA+0x10	R/W	ISP Trigger Control Register	0x0000_0000
FMC_DFBA	FMC_BA+0x14	R	Data Flash Base Address	0xXXXX_XXXX
FMC_FTCTL	FMC_BA+0x18	R/W	Flash Access Time Control Register	0x0000_0000
FMC_ISPSTS	FMC_BA+0x40	R/W	ISP Status Register	0xX0X0_000X
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

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_000X

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

Bits	Description	
[31:15]	**Reserved**	Reserved.
[24]	**INTEN**	**ISP Interrupt Enabled Bit (Write Protect)** 0 = ISP INT Disabled. 1 = ISP INT Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.Before use INT,user needs to clear the INTFLAG(FMC_ISPSTS[8]) make sure INT happen at correct time.
[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. 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. SPROM is erased/programmed if SPUEN is set to 0 SPROM is programmed at SPROM secured mode. Page Erase command at LOCK mode with ICE connection Erase or Program command at brown-out detected Destination address is illegal, such as over an available range. Invalid ISP commands **Note:** This bit is write-protected. Refer to the SYS_REGLCTL register.
[5]	**LDUEN**	**LDROM Update Enable Bit (Write Protect)** LDROM update enable bit. 0 = LDROM cannot be updated. 1 = LDROM can be updated. **Note:** This bit is write-protected. Refer to the SYS_REGLCTL register.

[4]	**CFGUEN**	**CONFIG Update Enable 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]	**SPUEN**	**SPROM Update Enable Bit (Write Protect)** 0 = SPROM cannot be updated. 1 = SPROM can be updated. **Note:** This bit is write-protected. Refer to the SYS_REGLCTL register.
[1]	**BS**	**Boot Selection (Write Protect)** Set/clear this bit to select next booting from LDROM/APROM, respectively. This bit also functions as chip booting status flag, which can be used to check where chip booted from. This bit is initiated with the inversed value of CBS[1] (CONFIG0[7]) after any reset is happened except CPU reset (RSTS_CPU is 1) or system reset (RSTS_SYS) is happened 0 = Booting from APROM. 1 = Booting from LDROM. **Note:** This bit is write-protected. Refer to the SYS_REGLCTL register.
[0]	**ISPEN**	**ISP Enable Bit (Write Protect)** I0 = 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 M031/M032 series is equipped with embedded Flash. ISPADDR[1:0] must be kept 00 for ISP 32-bit operation. For CRC32 Checksum Calculation command, this field is the Flash starting address for checksum calculation, 512 bytes alignment is necessary for checksum calculation. 16/32/64/128 Kbytes Flash: ISPADR[8:0] must be kept all 0 for Vector Page Re-map Command. 256/512 Kbytes Flash: ISPADR[11:0] must be kept all 0 for Vector Page Re-map Command.

ISP Data Register (FMC_ISPDAT)

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

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

Bits	Description	
[31:0]	**ISPDAT**	**ISP Data** Write data to this register before ISP program operation. Read data from this register after ISP read operation. For run CRC32 Checksum Calculation command, ISPDAT is the memory size (byte) and 512 bytes alignment. For ISP Read Checksum command, ISPDAT is the checksum result. If ISPDAT = 0x0000_0000, it means that (1) the checksum calculation is in progress, or (2) the memory range for checksum calculation is incorrect.

ISP Command (FMC_ISPCMD)

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

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

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6:0]	**CMD**	**ISP CMD** ISP command table is shown below: 0x00= Flash Read. 0x04= Read Unique ID. 0x08= Read All One. 0x0B= Read Company ID. 0x0C= Read Device ID. 0x0D= Read CRC32 Checksum. 0x21= Flash 32-bit Program. 0x22= Flash Page Erase. 0x23= Flash APROM Bank Erase. 0x27= Flash Multi-Word Program. 0x28= Run All One. 0x2C= APROM Address Operation Model Selection. 0x2D= Run CRC32 Checksum Calculation. 0x2E= Vector Remap. 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.

Data Flash Base Address Register (FMC_DFBA)

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

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

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

Flash Access Time Control Register (FMC_FTCTL)

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

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

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**CACHEINV**	**Flash Cache Invalidation (Write Protect)** 0 = Flash Cache Invalidation finished (default). 1 = Flash Cache Invalidation. **Note 1:** Write 1 to start cache invalidation. The value will be changed to 0 once the process is finished. **Note 2:** This bit is write-protected. Refer to the SYS_REGLCTL register. **Note 3:** Only supported in 256/512 Kbytes Flash.
[8:7]	**Reserved**	Reserved.
[6:4]	**FOM**	**Frequency Optimization Mode (Write Protect)** The M031/M032 series support adjustable Flash access timing to optimize the Flash access cycles in different system working frequency. For 16/32/64/128 Kbytes Flash: 000 = Frequency is less than or equal to 48 MHz. 001 = Frequency is less than or equal to 24 MHz. Others = Reserved. For 256/512 Kbytes Flash: 000 = Frequency is less than or equal to 72 MHz. 001 = Frequency is less than or equal to 12 MHz. 010 = Frequency is less than or equal to 36 MHz. 011 = Frequency is less than or equal to 60 MHz. Others = Reserved. **Note:** This bit is write-protected. Refer to the SYS_REGLCTL register.
[3:0]	**Reserved**	Reserved.

ISP Status Register (FMC_ISPSTS)

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

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

Bits	Description	
[31]	**SCODE**	**Security Code Active Flag** This bit is set to 1 by hardware when detecting SPROM secured code is active at Flash initialization, or software writes 1 to this bit to make secured code active; this bit is only cleared by SPROM page erase operation. 0 = SPROM secured code is inactive. 1 = SPROM secured code is active.
[30]	**FBS**	**Flash Bank Select Indicator** This bit indicates which APROM address model is selected to boot. 0 = Address model OP0 is selected to boot. 1 = Address model OP1 is selected to boot. **Note:** Only supported in 256/512 Kbytes Flash.
[29:9]	**VECMAP**	**Vector Page Mapping Address (Read Only)** All access to 0x0000_0000~0x0000_01FF is remapped to the Flash memory or SRAM address {VECMAP[20:0], 9'h000} ~ {VECMAP[20:0], 9'h1FF}, except SPROM. VECMAP [20:19] = 00 system vector address is mapped to Flash memory. VECMAP [20:19] = 10 system vector address is mapped to SRAM memory. VECMAP [18:12] should be 0.
[8]	**INTFLAG**	**ISP Command Finish Interrupt Flag** 0 = ISP Not Finished. 1 = ISP done or ISPFF set. **Note:** Only supported in 256/512 Kbytes Flash.
[7]	**ALLONE**	**Flash All-one Verification Flag** This bit is set by hardware if all of Flash bits are 1, and cleared if Flash bits are not all 1 after "Run Flash All-One Verification" complete; this bit can also be cleared 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. SPROM is erased/programmed if SPUEN is set to 0 SPROM is programmed at SPROM secured mode. Page Erase command at LOCK mode with ICE connection Erase or Program command at brown-out detected Destination address is illegal, such as over an available range. Invalid ISP commands
[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 successful. 1 = Flash Program is failed. Program data is different with data in the Flash memory
[4:3]	**Reserved**	Reserved.
[2:1]	**CBS**	**Boot Selection of CONFIG (Read Only)** This bit is initiated with the CBS (CONFIG0[7:6]) after any reset is happened except CPU reset (RSTS_CPU is 1) or system reset (RSTS_SYS) is happened. 00 = LDROM with IAP mode. 01 = LDROM without IAP mode. 10 = APROM with IAP mode. 11 = APROM without IAP mode.
[0]	**ISPBUSY**	**ISP BUSY (Read Only)** 0 = ISP operation is finished. 1 = ISP operation is busy.

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

ISP Multi-word Program Address Register (FMC_MPADDR)

Register	Offset	R/W	Description	Reset Value
FMC_MPADDR	FMC_BA+0xC4	R	ISP Multi-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.

6.5 General Purpose I/O (GPIO)

6.5.1 Overview

This chip has up to 111 General Purpose I/O pins to be shared with other function pins depending on the chip configuration. These 111 pins are arranged in 5 ports named as PA, PB, PC, PD, PE, PF, PG and PH. PA and PB has 16 pins on port. PC has 15 pins on port. PD and PE has 16 pins on port. PF has 14 pins on port. PG has 10 pins on port. PH has 8 pins on port. Each of the 111 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]).

6.5.2 Features

- Four I/O modes:
  - Quasi-bidirectional mode
  - Push-Pull Output mode
  - Open-Drain Output mode
  - Input only with high impendence mode
- I/O pin can be configured as interrupt source with edge/level setting
- Input schmitt trigger function
- 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
- I/O pin internal pull-up resistor enabled only in Quasi-bidirectional I/O mode
- Enabling the pin interrupt function will also enable the wake-up function

6.5.3 Block Diagram

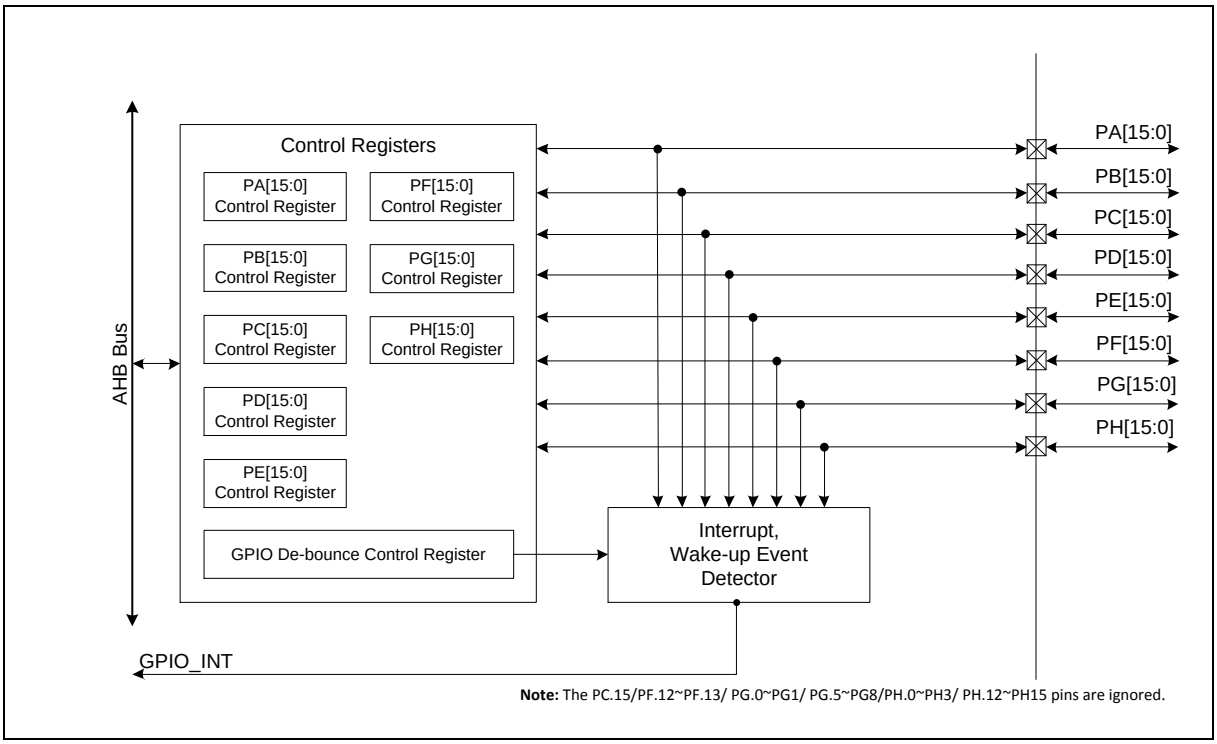


Figure 6.5-1 GPIO Controller Block Diagram

Note: The PC.15/PF.12~PF.13/ PG.0~PG1/ PG.5~PG8/PH.0~PH3/ PH.12~PH15 pins are not avaliable.

6.5.4 Basic Configuration

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

6.5.6 Functional Description

6.5.6.1 Input Mode

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

6.5.6.2 Push-pull Output Mode

Figure 6.5-2 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.

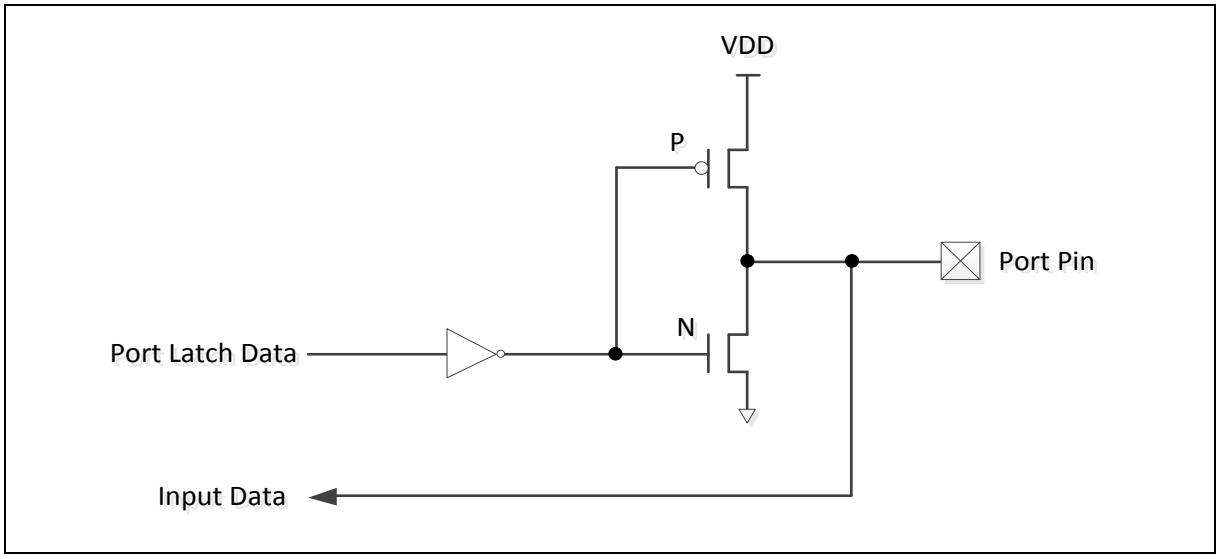


Figure 6.5-2 Push-Pull Output

6.5.6.3 Open-drain Mode

Figure 6.5-2 shows the diagram of Open-drain Mode. Set MODEn (Px_MODE[2n+1:2n]) to 10 the Px.n pin is in Open-drain mode and the digital output function of I/O pin supports only sink current capability, an external pull-up register is needed for driving high state. If the bit value in the corresponding DOUT (Px_DOUT[n]) bit is 0, the pin drive a low output on the pin. If the bit value in the corresponding DOUT (Px_DOUT[n]) bit is 1, the pin output drives high that is controlled by external pull high resistor.

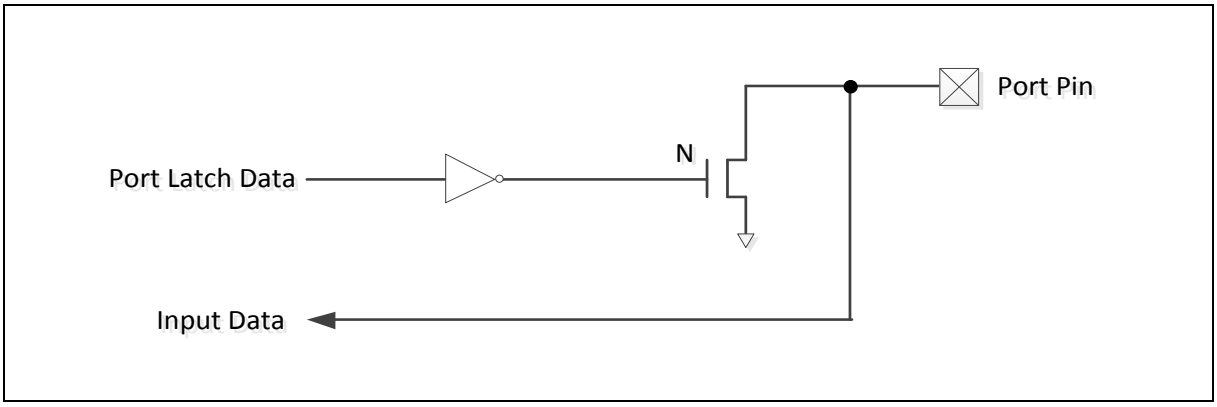


Figure 6.5-3 Open-Drain Output

6.5.6.4 Quasi-bidirectional Mode

Figure 6.5-4 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. 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.

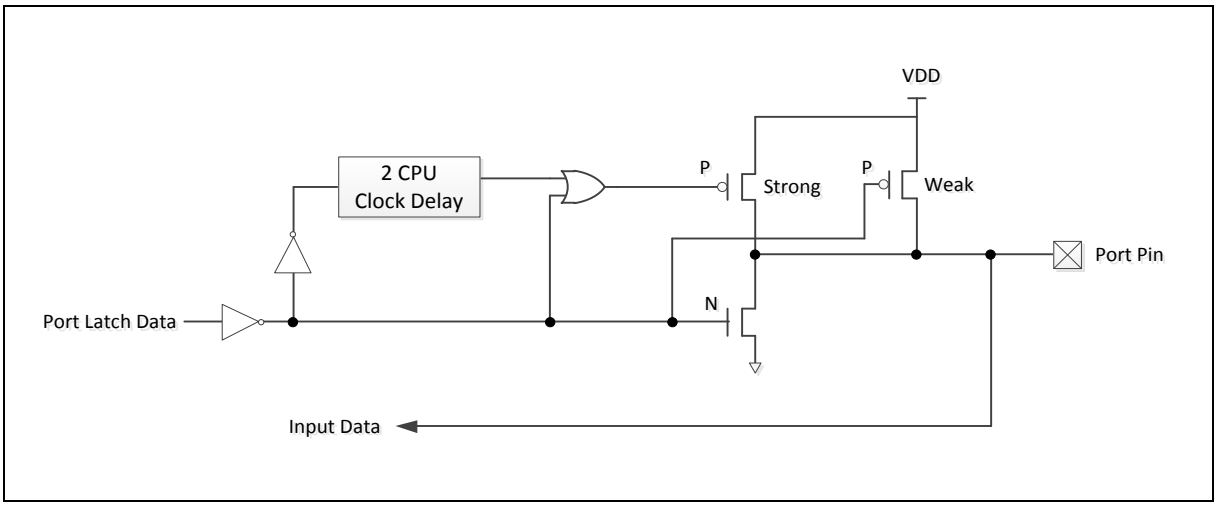


Figure 6.5-4 Quasi-Bidirectional I/O Mode

6.5.6.5 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.

6.5.6.6 GPIO De-bounce Function

GPIO de-bounce function can be used to sample interrupt input for each GPIO pin and prevent unexpected interrupt happened which caused by noise. GPIO de-bounce function only support edge detection trigger type. 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 (38.4 kHz) by setting DBCLKSRC (Px_DBCTL[4]) register. And DBCLKSEL (Px_DBCTL[3:0]) register can control sampling cycle period.

Figure 6.5-5 shows GPIO rising edge trigger interrupt. The interval of time between the two valid sample signal is determined by DBCLKSRC (Px_DBCTL[4]) and DBCLKSEL (Px_DBCTL[3:0]). Each valid data from GPIO pin need to be sample twice. For rising edge setting, if pin status is low before setting DBEN (Px_DBEN), interrupt will happen when generating a pin high valid data. But, if pin status is high before setting DBEN (Px_DBEN), interrupt will happen when generating a pin low valid data first, and then generating a pin high valid data. For falling edge trigger, Figure 6.5-6 shows the situation is opposite to rising edge trigger.

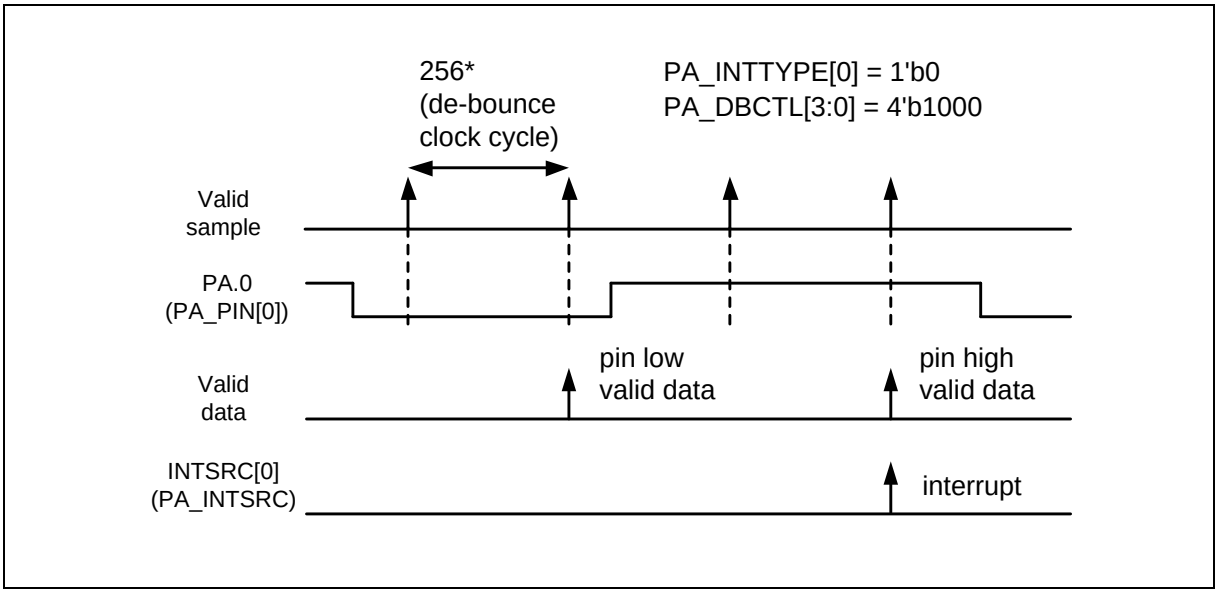


Figure 6.5-5 GPIO Rising Edge Trigger Interrupt

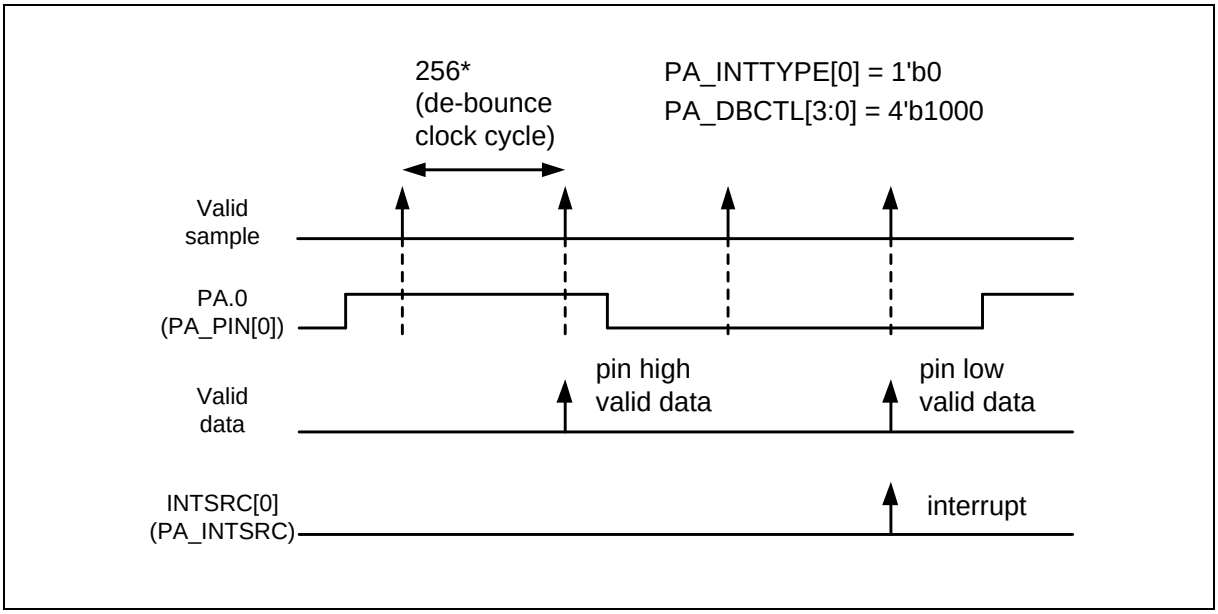


Figure 6.5-6 GPIO Falling Edge Trigger Interrupt

GPIO de-bounce function are also supported in Power-down mode. Table 6.5-1 shows the de-bounce function support situation in different system status. The de-bounce function can support in Power-down mode by setting DBENn(Px_DBEN[n]) and DBCLKSRC(Px_DBCTL[4]) to 1. DBCLKSEL (Px_DBCTL[3:0]) can be set to control the GPIO de-bounce time to wake up system.

System Status	DBEN	DBCLKSRC	Description
Normal Mode / Idle Mode	0	0	No de-bounce function
		1	No de-bounce function
	1	0	De-bounce function using HCLK

		1	De-bounce function using LIRC (38.4 kHz)
Power Down Mode	0	0	No de-bounce function
		1	No de-bounce function
	1	0	No de-bounce function
		1	De-bounce function using LIRC (38.4 kHz)

Table 6.5-1 De-Bounce Function Setting Table

6.5.6.7 GPIO Digital Input Path Disable Control

User can disable GPIO digital input path by setting DINOFF (Px_DINOFF[n+16]). When GPIO digital input path is disabled, the digital input pin value PIN (Px_PIN[n]) is tied to low. By the way, the GPIO digital input path is force disabled by hardware and DINOFF control is useless when I/O function configure as ADC/ACMP/ext. XTL

6.5.7 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
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
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

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
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
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_CFFF
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
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
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
GPIO_DBCTL	GPIO_BA+0x440	R/W	Interrupt De-bounce Control Register	0x0000_0020
PAn_PDIO n=0,1..15	GPIO_BA+0x800+(0x04 * n)	R/W	GPIO PA.n Pin Data Input/Output Register	0x0000_000X
PBn_PDIO n=0,1..15	GPIO_BA+0x840+(0x04 * n)	R/W	GPIO PB.n Pin Data Input/Output Register	0x0000_000X
PCn_PDIO n=0,1..14	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..11,14,15	GPIO_BA+0x940+(0x04 * n)	R/W	GPIO PF.n Pin Data Input/Output Register	0x0000_000X
PGn_PDIO n=2,3,4,9,10..15	GPIO_BA+0x980+(0x04 * n)	R/W	GPIO PG.n Pin Data Input/Output Register	0x0000_000X
PHn_PDIO n=4,5..11	GPIO_BA+0x9C0+(0x04 * n)	R/W	GPIO PH.n Pin Data Input/Output Register	0x0000_000X

6.5.8 Register Description

Port A-H 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

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-H 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/PF.12~13/PG.0~1,5~8/PH.0~3,12~15 pin is ignored.

Port A-H 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

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-H 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/PF.12~13/PG.0~1,5~8/PH.0~3,12~15 pin is ignored.
[15:0]	**Reserved**	Reserved.

Port A-H 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_CFFF
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

31	30	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-H 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/PF.12~13/PG.0~1,5~8/PH.0~3,12~15 pin is ignored.

Port A-H 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

31	30	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-H 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/PF.12~13/PG.0~1,5~8/PH.0~3,12~15 pin is ignored.

Port A-H 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

31	30	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-H 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/PF.12~13/PG.0~1,5~8/PH.0~3,12~15 pin is ignored.

Port A-H 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

31	30	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-H 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/PF.12~13/PG.0~1,5~8/PH.0~3,12~15 pin is ignored.

Port A-H 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

31	30	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-H 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/PF.12~13/PG.0~1,5~8/PH.0~3,12~15 pin is ignored.

Port A-H 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

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-H 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/PF.12~13/PG.0~1,5~8/PH.0~3,12~15 pin is ignored.
[n] n=0,1..15	**FLIEN[n]**	**Port A-H 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/PF.12~13/PG.0~1,5~8/PH.0~3,12~15 pin is ignored.

Port A-H 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

31	30	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-H 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/PF.12~13/PG.0~1,5~8/PH.0~3,12~15 pin is ignored.

Interrupt De-bounce Control Register (GPIO_DBCTL)

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

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

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5]	**ICLKON**	**Interrupt Clock on Mode** 0 = Edge detection circuit is active only if I/O pin corresponding RHIEN (Px_INTEN[n+16])/FLIEN (Px_INTEN[n]) bit is set to 1. 1 = All I/O pins edge detection circuit is always active after reset. **Note:** It is recommended to disable this bit to save system power if no special application concern.
[4]	**DBCLKSRC**	**De-bounce Counter Clock Source Selection** 0 = De-bounce counter clock source is the HCLK. 1 = De-bounce counter clock source is the 38.4 kHz 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.

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..14	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..11,14,15	GPIO_BA+0x940+(0x04 * n)	R/W	GPIO PF.n Pin Data Input/Output Register	0x0000_000X
PGn_PDIO n=2,3,4,9..15	GPIO_BA+0x980+(0x04 * n)	R/W	GPIO PG.n Pin Data Input/Output Register	0x0000_000X
PHn_PDIO n=4,5..11	GPIO_BA+0x9C0+(0x04 * n)	R/W	GPIO PH.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 (Px_DOUT[0]), reading PA0_PDIO will return the value of PIN (PA_PIN[0]). **Note 1:** The writing operation will not be affected by register DATMSK (Px_DATMSK[n]). **Note 2:** The PC.15/PF.12~13/PG.0~1,5~8/PH.0~3,12~15 pin is ignored.

6.6 PDMA Controller (PDMA)

6.6.1 Overview

The peripheral direct memory access (PDMA) controller is used to provide high-speed data transfer. The PDMA controller can transfer data from one address to another without CPU intervention. This has the benefit of reducing the workload of CPU and keeps CPU resources free for other applications. The PDMA controller has a total of 9 channels and each channel can perform transfer between memory and peripherals or between memory and memory.

6.6.2 Features

- Supports 9 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 I^2C, SPI/I^2S, UART, USCI, ADC, PWM , QSPI and TIMER request
- Supports Scatter-Gather mode to perform sophisticated transfer through the use of the descriptor link list table
- Supports single and burst transfer type
- Supports time-out function on channel 0 and channel1

6.6.3 Block Diagram

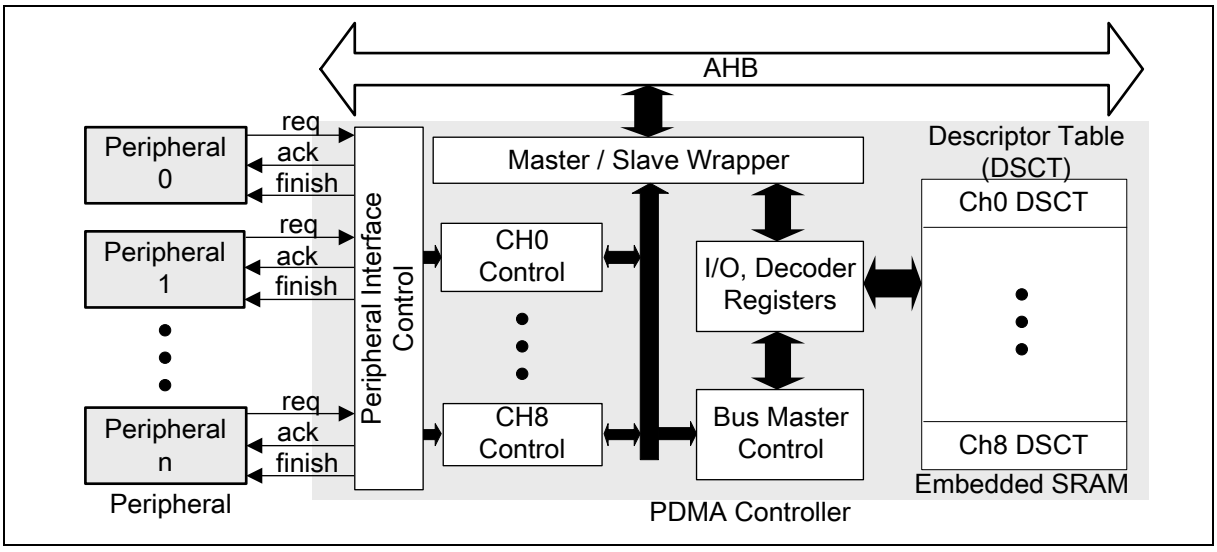


Figure 6.6-1 PDMA Controller Block Diagram

6.6.4 Basic Configuration

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

6.6.5 Functional Description

The PDMA controller transfers data from one address to another without CPU intervention. The PDMA controller supports 9 independent channels and serves only one channel at one time, as the result, PDMA controller supports two level channel priorities: fixed and round-robin priority, PDMA controller serves channel in order from highest to lowest priority channel. The PDMA controller supports two operation modes: Basic mode and Scatter-gather mode. Basic mode is used to perform one descriptor table transfer. Scatter-gather mode has more entries for each PDMA channel, and thus the PDMA controller supports sophisticated transfer through the entries. The descriptor table entry data structure contains many transfer information including the transfer source address, transfer destination address, transfer count, burst size, transfer type and operation mode. Figure 6.6-2 shows the diagram of descriptor table (DSCT) data structure.

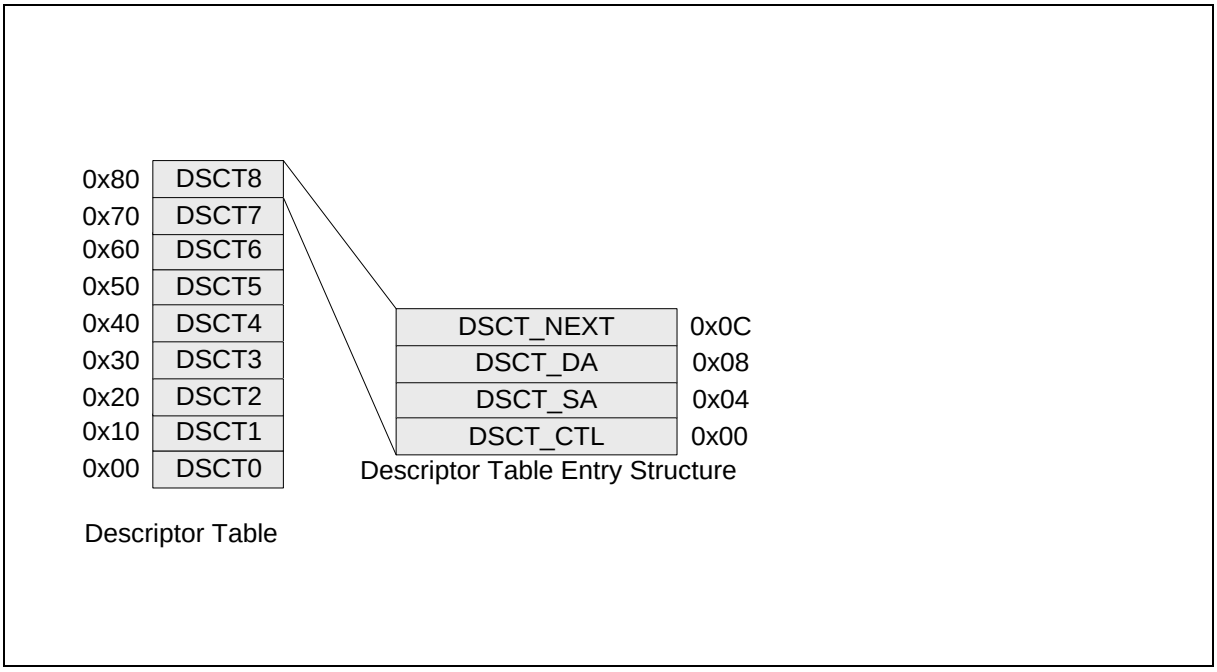


Figure 6.6-2 Descriptor Table Entry Structure

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).

6.6.5.1 Channel Priority

The PDMA controller supports two level channel priorities including fixed and round-robin priority. The fixed priority channel has higher priority than round-robin priority channel. If multiple channels are set as fixed or round-robin priority, the higher channel will have higher priority. The priority order is listed in Table 6.6-1.

PDMA_PRISET	Channel Number	Priority Setting	Arbitration Priority In Descending Order
1	8	Channel8, Fixed Priority	Highest
1	7	Channel7, Fixed Priority	---

---	---	---	---
1	0	Channel0, Fixed Priority	---
0	8	Channel8, Round-Robin Priority	---
0	7	Channel7, Round-Robin Priority	---
---	---	---	---
0	0	Channel0, Round-Robin Priority	Lowest

Table 6.6-1 Channel Priority Table

6.6.5.2 PDMA Operation Mode

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

Basic Mode

Basic mode is used to perform one descriptor table transfer mode. This mode can be used to transfer data between memory and memory, peripherals and memory or peripherals and peripherals, but if user want to transfer data between peripherals and peripherals, one thing must be sured is 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], n denotes PDMA channel), the default setting is in idle state (OPMODE (PDMA_DSCTn_CTL[1:0]) = 0x0) and recommend user configure the descriptor table in idle state. If operation mode is not in idle state, user re-configure channel setting may make some operation error.

User must fill the transfer count TXCNT (PDMA_DSCTn_CTL[31:16]) register and select transfer width TXWIDTH (PDMA_DSCTn_CTL[13:12]), destination address increment size DAINC (PDMA_DSCTn_CTL[11:10]), source address increment size SAINC (PDMA_DSCTn_CTL[9:8]), burst size BURSIZE (PDMA_DSCTn_CTL[6:4]) and transfer type TXTYPE (PDMA_DSCTn_CTL[2]), then the PDMA controller will perform transfer operation in transfer state after receiving request signal. Finishing this task will generate an interrupt to CPU if corresponding PDMA interrupt bit INTENn (PDMA_INTEN[8:0]) is enabled and the operation mode will be updated to idle state as shown in Figure 6.6-3. If software configures the operation mode to idle state, the PDMA controller will not perform any transfer and then clear this operation request. Finishing this task will also generate an interrupt to CPU if corresponding PDMA interrupt bit is enabled.

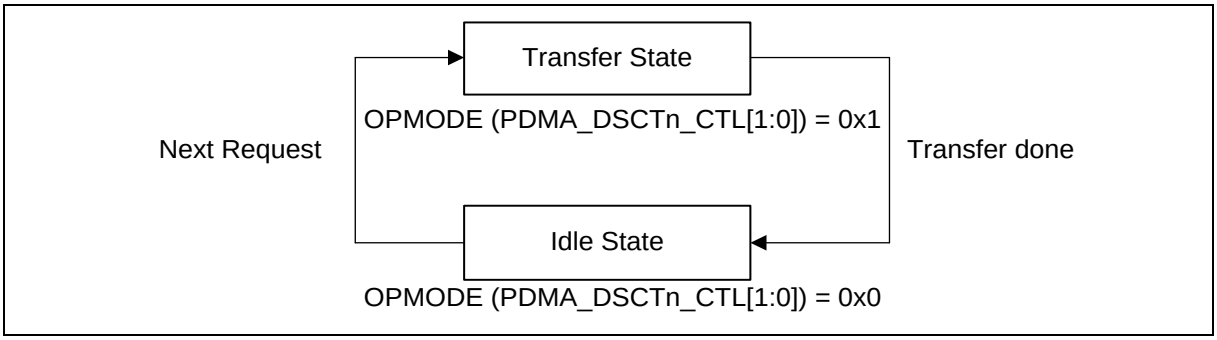


Figure 6.6-3 Basic Mode Finite State Machine

Scatter-Gather Mode

Scatter-Gather mode is a complex mode and can perform sophisticated transfer through the use of the description link list table as shown in Figure 6.6-4. Through operation mode user can perform peripheral wrapper-around, 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[8:0]) flag will be asserted and if this bit is “1” TDIFn will not be active).

If channel 8 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 16bits without last two bits [1:0] valid in PDMA_DSCTn_NEXT), and then the next DSCT entry start address is 0x2000_0100.

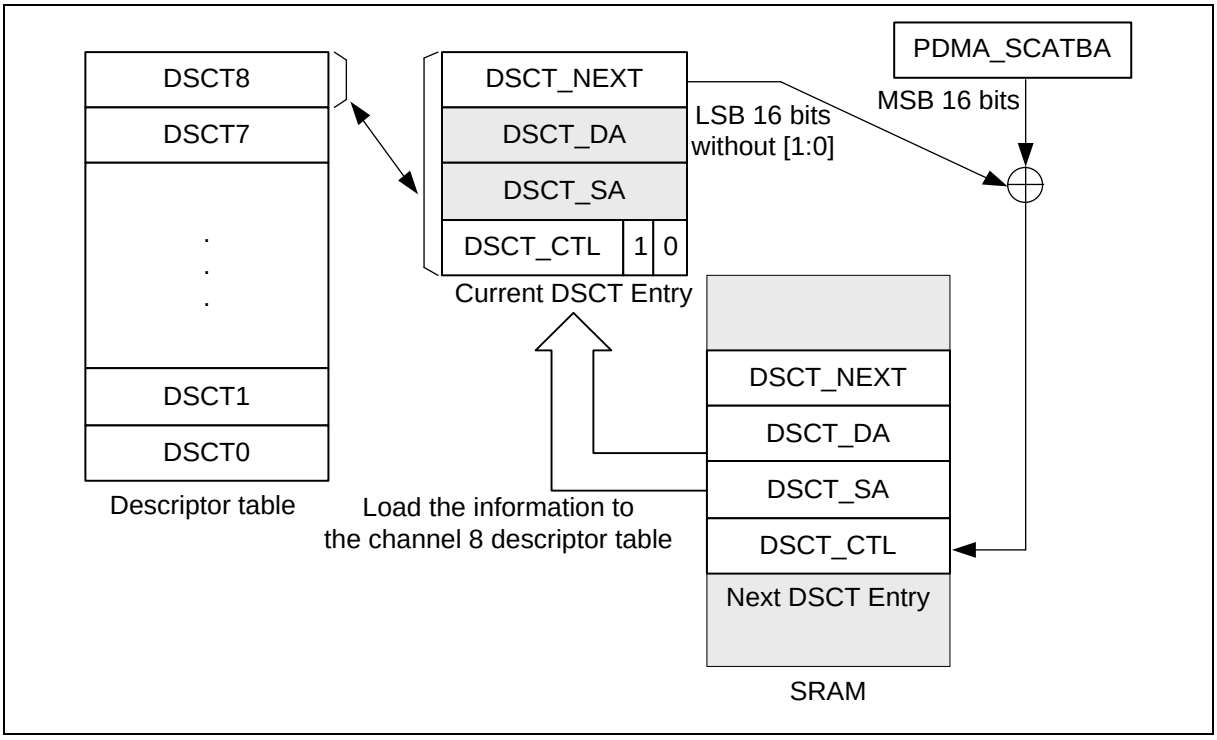


Figure 6.6-4 Descriptor Table Link List Structure

The above link list table operation is DSCT state in Scatter-Gather Mode as shown in Figure 6.6-5. When loading the information is finished, it will go to transfer state and start transfer by this information automatically. However, if the next PDMA information is also 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.

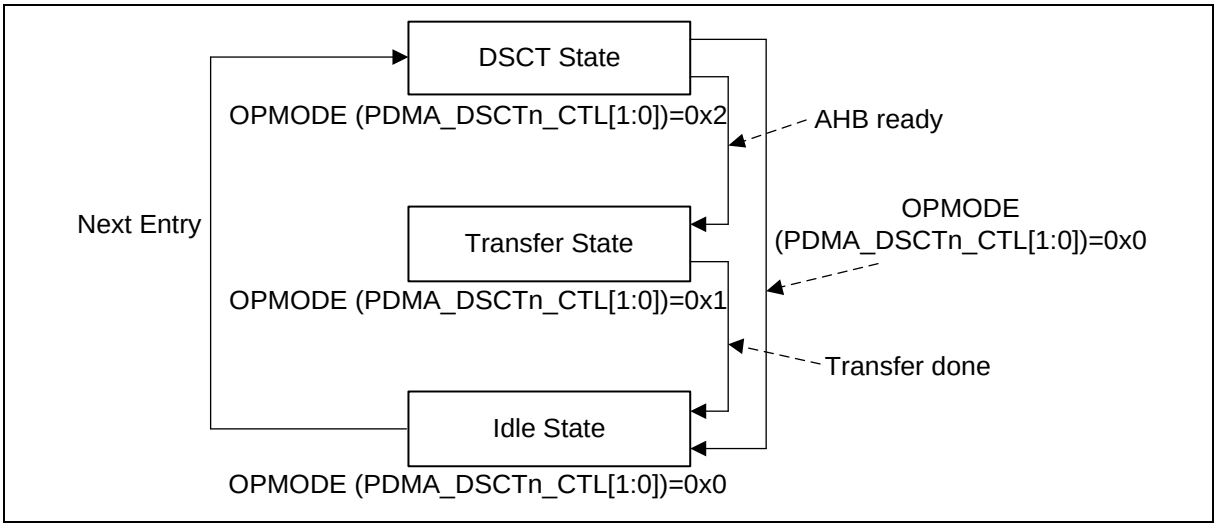


Figure 6.6-5 Scatter-Gather Mode Finite State Machine

6.6.5.3 Transfer Type

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

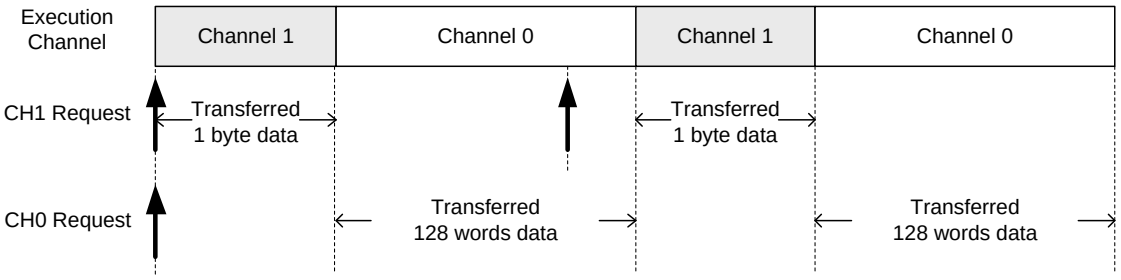
When the PDMA controller is operated in single transfer type, each transfer data needs one request signal for one transfer, after transferred data, TXCNT (PDMA_DSCTn_CTL[31:16]) will decrease 1. Transfer will be finished after the TXCNT (PDMA_DSCTn_CTL[31:16]) decreases to 0. In this mode, the BURSIZE (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.

Figure 6.6-6 shows an example about single and burst transfer type in basic mode. In this example, channel 1 uses single transfer type and TXCNT (PDMA_DSCTn_CTL[31:16]) = 127. Channel 0 uses burst transfer type, BURSIZE (PDMA_DSCTn_CTL[6:4]) = 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 6.6-6 Example of Single Transfer Type and Burst Transfer Type in Basic Mode

6.6.5.4 Channel Time-out

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

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

The 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 6.6-7 shows an example about time-out counter operation. The operation sequence is described below:

1. The channel 0 time-out counter is not counting when time-out function is enabled by setting TOUTEN0(PDMA_TOUTEN[0]) bit to 1.
2. The 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. The 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. The time-out counter is reset to 0 when time-out function is disabled.

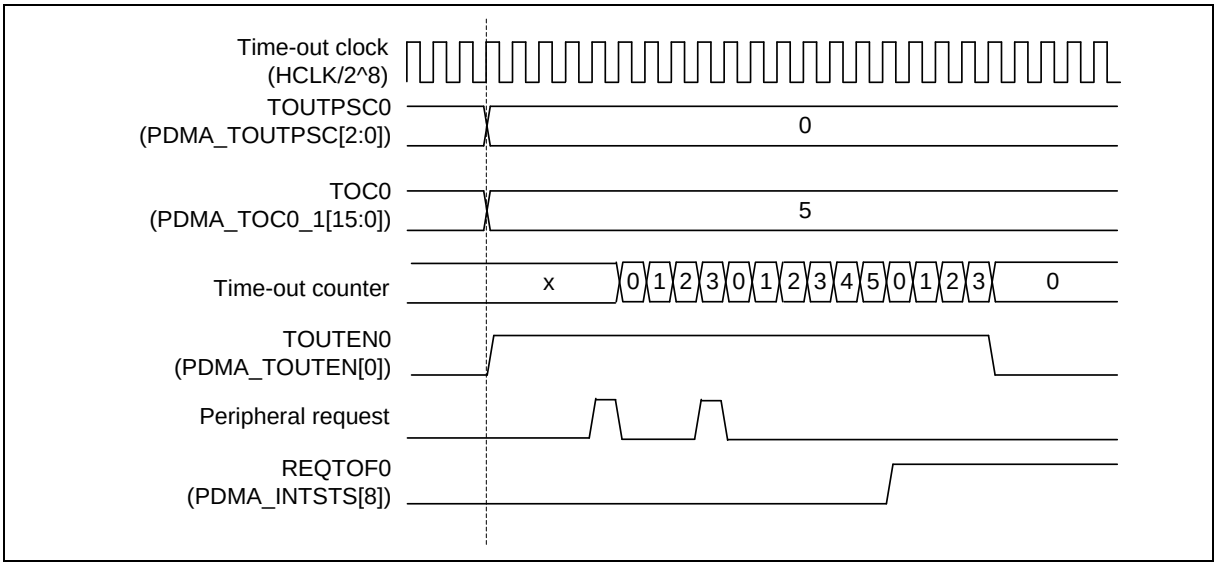


Figure 6.6-7 Example of PDMA Channel 0 Time-out Counter Operation

6.6.6 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..8**	PDMA_BA+0x10*n	R/W	Descriptor Table Control Register of PDMA Channel n	0xXXXX_XXXX
PDMA_DSCTn_SA **n = 0,1..8**	PDMA_BA+0x0004+0x10*n	R/W	Source Address Register of PDMA Channel n	0xXXXX_XXXX
PDMA_DSCTn_DA **n = 0,1..8**	PDMA_BA+0x0008+0x10*n	R/W	Destination Address Register of PDMA Channel n	0xXXXX_XXXX
PDMA_DSCTn_NEXT **n = 0,1..8**	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..8**	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_7	PDMA_BA + 0x484	R/W	PDMA Request Source Select Register 1	0x0000_0000

PDMA_REQSEL8	PDMA_BA + 0x488	R/W	PDMA Request Source Select Register 2	0x0000_0000

6.6.7 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 16384, every transfer may be byte, half-word or word that is dependent on TXWIDTH field. **Note:** When PDMA finish each transfer data, this field will be decrease immediately.
[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 depend 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).

Bits	Description	
		Others = Increment and size depend 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 not generates TDIFn(PDMA_TDSTS[8:0]) 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 curren 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. **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							CHEN8
7	6	5	4	3	2	1	0
CHEN7	CHEN6	CHEN5	CHEN4	CHEN3	CHEN2	CHEN1	CHEN0

Bits	Description	
[31:9]	**Reserved**	Reserved.
[n] n=0,1..8	**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							PAUSE8
7	6	5	4	3	2	1	0
PAUSE7	PAUSE6	PAUSE5	PAUSE4	PAUSE3	PAUSE2	PAUSE1	PAUSE0

Bits	Description	
[31:9]	**Reserved**	Reserved.
[n] n=0,1..8	**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..8) and clear request active flag(PDMA_TRGSTS[n:0], n=0,1..8). 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							SWREQ8
7	6	5	4	3	2	1	0
SWREQ7	SWREQ6	SWREQ5	SWREQ4	SWREQ3	SWREQ2	SWREQ1	SWREQ0

Bits	Description	
[31:9]	Reserved	Reserved.
[n] n=0,1..8	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							REQSTS8
7	6	5	4	3	2	1	0
REQSTS7	REQSTS6	REQSTS5	REQSTS4	REQSTS3	REQSTS2	REQSTS1	REQSTS0

Bits	Description	
[31:9]	**Reserved**	Reserved.
[n] n=0,1..8	**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							**FPRISET8**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FPRISET7	**FPRISET6**	**FPRISET5**	**FPRISET4**	**FPRISET3**	**FPRISET2**	**FPRISET1**	**FPRISET0**

Bits	Description	
[31:9]	**Reserved**	Reserved.
[n] n=0,1..8	**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 only set to fixed priority, clear fixed priority use PDMA_PRICLR register.

PDMA Fix Priority Clear Register (PDMA_PRICLR)

Register	Offset	R/W	Description	Reset Value
PDMA_PRICLR	PDMA_BA + 0x414	W	PDMA Fixed Priority Clear Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							FPRICLR8
7	6	5	4	3	2	1	0
FPRICLR7	FPRICLR6	FPRICLR5	FPRICLR4	FPRICLR3	FPRICLR2	FPRICLR1	FPRICLR0

Bits	Description	
[31:9]	**Reserved**	Reserved.
[n] n=0,1..8	**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							**INTEN8**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
INTEN7	**INTEN6**	**INTEN5**	**INTEN4**	**INTEN3**	**INTEN2**	**INTEN1**	**INTEN0**

Bits	Description	
[31:9]	**Reserved**	Reserved.
[n] n=0,1..8	**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 clear 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 clear 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.

Bits	Description	
		1 = AHB bus ERROR response received.

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

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

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**ABTIF8**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ABTIF7	**ABTIF6**	**ABTIF5**	**ABTIF4**	**ABTIF3**	**ABTIF2**	**ABTIF1**	**ABTIF0**

Bits	Description	
[31:9]	**Reserved**	Reserved.
[n] n=0,1..8	**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.

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							TDIF8
7	6	5	4	3	2	1	0
TDIF7	TDIF6	TDIF5	TDIF4	TDIF3	TDIF2	TDIF1	TDIF0

Bits	Description	
[31:9]	**Reserved**	Reserved.
[n] n=0,1..8	**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							ALIGN8
7	6	5	4	3	2	1	0
ALIGN7	ALIGN6	ALIGN5	ALIGN4	ALIGN3	ALIGN2	ALIGN1	ALIGN0

Bits	Description	
[31:9]	**Reserved**	Reserved.
[n] n=0,1..8	**ALIGNn**	**Transfer Alignment Flag** This bit indicates whether source and destination address both follow transfer width setting, user can write 1 to clear these bits. 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.

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							TXACTF8
7	6	5	4	3	2	1	0
TXACTF7	TXACTF6	TXACTF5	TXACTF4	TXACTF3	TXACTF2	TXACTF1	TXACTF0

Bits	Description	
[31:9]	**Reserved**	Reserved.
[n] n=0,1..8	**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. The example of time-out period can refer TOC0 bit description.
[15:0]	**TOC0**	**Time-out Counter for Channel 0** This controls the period of time-out function for channel 0. The calculation unit is based on TOUTPSC0 (PDMA_TOUTPSC[2:0]) clock. Time-out period = (Period of time-out clock) * (16-bit TOCn), n = 0,1.

PDMA Channel Reset 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							**CH8RST**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CH7RST	**CH6RST**	**CH5RST**	**CH4RST**	**CH3RST**	**CH2RST**	**CH1RST**	**CH0RST**

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8:0]	**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 = reserved.** **4 = Channel connects to UART0_TX.** **5 = Channel connects to UART0_RX.** **6 = Channel connects to UART1_TX.**

Bits	Description	
		7 = Channel connects to UART1_RX. **8 = Channel connects to UART2_TX.** **9 = Channel connects to UART2_RX.** **10 = Channel connects to USCI0_TX.** **11 = Channel connects to USCI0_RX.** **12 = Channel connects to USCI1_TX.** **13 = Channel connects to USCI1_RX.** **14 = Reserved.** **15 = Reserved.** **16 = Channel connects to QSPI0_TX.** **17 = Channel connects to QSPI0_RX.** **18 = Channel connects to SPI0_TX.** **19 = Channel connects to SPI0_RX.** **20 = Channel connects to ADC_RX.** **21 = Channel connects to PWM0_P1_RX.** **22 = Channel connects to PWM0_P2_RX.** **23 = Channel connects to PWM0_P3_RX.** **24 = Channel connects to PWM1_P1_RX.** **25 = Channel connects to PWM1_P2_RX.** **26 = Channel connects to PWM1_P3_RX.** **27 = Reserved.** **28 = Channel connects to I2C0_TX.** **29 = Channel connects to I2C0_RX.** **30 = Channel connects to I2C1_TX.** **31 = Channel connects to I2C1_RX.** **32 = Channel connects to TMR0.** **33 = Channel connects to TMR1.** **34 = Channel connects to TMR2.** **35 = Channel connects to TMR3.** **36 = Channel connects to UART3_TX.** **37 = Channel connects to UART3_RX.** **38 = Channel connects to UART4_TX.** **39 = Channel connects to UART4_RX.** **40 = Channel connects to UART5_TX.** **41 = Channel connects to UART5_RX.** **42 = Channel connects to UART6_TX.** **43 = Channel connects to UART6_RX.** **44 = Channel connects to UART7_TX.** **45 = Channel connects to UART7_RX.** **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_7)

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

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

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29:24]	**REQSRC7**	**Channel 7 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 7. User can configure the peripheral setting by REQSRC7. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[23:22]	**Reserved**	Reserved.
[21:16]	**REQSRC6**	**Channel 6 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 6. User can configure the peripheral setting by REQSRC6. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[15:14]	**Reserved**	Reserved.
[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.

PDMA Request Source Select Register 2 (PDMA_REQSEL8)

Register	Offset	R/W	Description	Reset Value
PDMA_REQSEL8	PDMA_BA + 0x488	R/W	PDMA Request Source Select Register 2	0x0000_0000

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

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**REQSRC8**	**Channel 8 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 8. User can configure the peripheral setting by REQSRC8. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.

6.7 Timer Controller (TMR)

6.7.1 Overview

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

6.7.2 Features

6.7.2.1 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
- Supports event counting source from internal USB SOF signal
- 24-bit capture value is readable through CAPDAT (TIMERx_CAP[23:0])
- Supports external capture pin event for interval measurement
- Supports external capture pin event to reset 24-bit up counter
- Supports internal capture triggered while internal ACMP output signal and LIRC transition
- Supports chip wake-up from Idle/Power-down mode if a timer interrupt signal is generated
- Support Timer0 ~ Timer3 time-out interrupt signal or capture interrupt signal to trigger PWM, ADC, PDMA, BPWM function
- Supports Inter-Timer trigger mode

6.7.3 Block Diagram

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

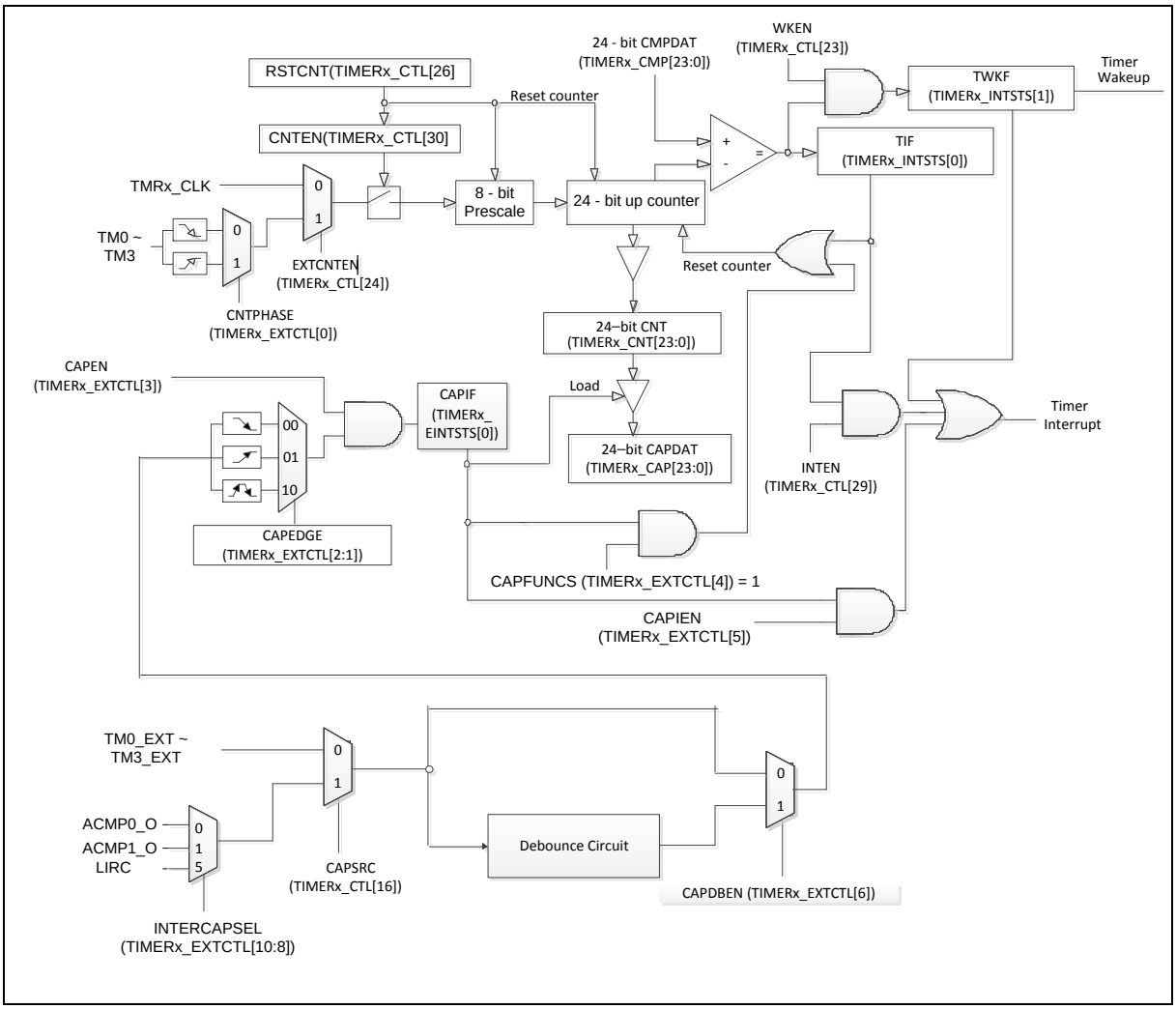


Figure 6.7-1 Timer Controller Block Diagram

6.7.4 Basic Configuration

- Clock Source Configuration
  - The clock source of Timer0 ~ Timer3 in timer mode can be enabled in TMRxCKEN (CLK_APBCLK0[5:2]).
  - Select the source of Timer0 ~ Timer3 on 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.

The Timer0 ~ Timer3 clock control is shown in Figure 6.7-2.

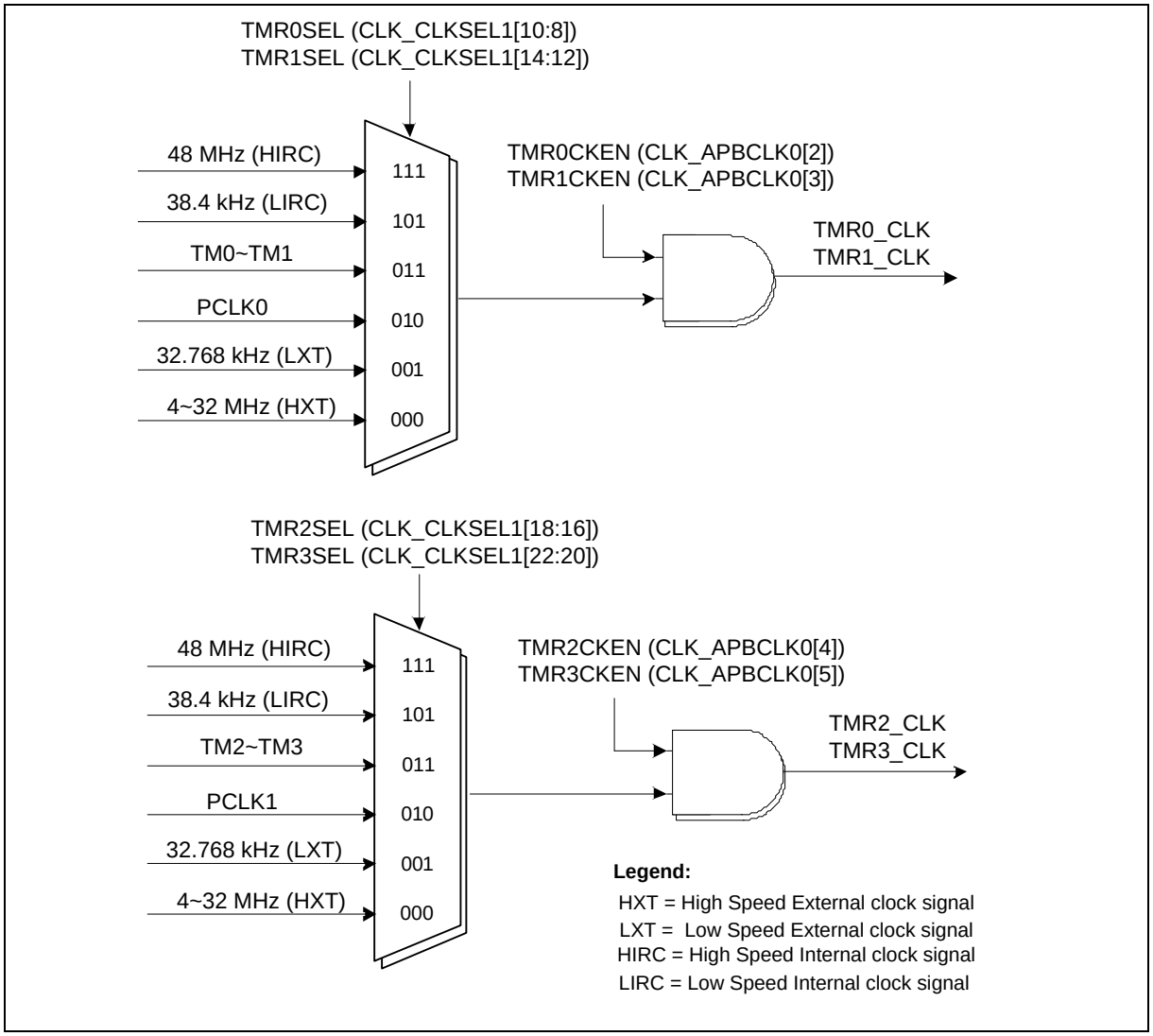


Figure 6.7-2 Clock Source of Timer Controller

- Timer0 ~ Timer3 MFP configurations
  - The MFP table for Timer0 ~ Timer3 is shown as Table 6.7-1.

Signal Name	Pin Name	MFP
TM0	PB.5, PC.7	MFP14

	PG.2	MFP13
TM1	PB.4, PC.6	MFP14
	PC.14, PG.3	MFP13
TM2	PA.7, PB.3, PD.0	MFP14
	PF.15, PG.4	MFP13
TM3	PA.6, PB.2, PD.15, PF.14	MFP14
	PF.11	MFP13
TM0_EXT	PA.11, PB.15	MFP13
TM1_EXT	PA.10, PB.14	MFP13
TM2_EXT	PA.9, PB.13	MFP13
TM3_EXT	PA.8, PB.12	MFP13

Table 6.7-1 Timer0 ~ Timer3 MFP table

6.7.5 Functional Description

6.7.5.1 Timer Interrupt Flag

The timer controller supports the followong interrupt flags; one is TIF (TIMERx_INTSTS[0]) and it is set while timer counter value CNT (TIMERx_CNT[23:0]) matches the timer compared value CMPDAT (TIMERx_CMP[23:0]). The other situation that CAPIF (TIMERx_EINTSTS[0]) is set means when the transition on the TMx_EXT pin, LIRC, or ACMP is associated with CAPEDGE (TIMERx_EXTCTL[2:1]) setting. The TWKF (TIMERx_INTSTS[1]) bit indicates the interrupt wake-up flag status of timer. User can set CAPSRC (TIMERx_CTL[16]) and INTERCAPSEL (TIMERx_EXTCTL[10:8]) to select capture source. The TWKF (TIMERx_INTSTS[1]) bit indicates the interrupt wake-up flag status of timer. Set WKEN (TIMERx_CTL[23]) to 1 to use wake-up function.

6.7.5.2 Timer Counting Mode

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

6.7.5.3 One–Shot Mode

If the timer controller is configured in 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 set, the timer interrupt signal is generated and sent to NVIC to inform CPU.

User can set ICEDEBUG (TIMERx_CTL[31]) to 1 to disable ICE debug mode acknowledgement that affects TIMER counting when CPU is halted.

6.7.5.4 Periodic Mode

If the timer controller is configured in 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 set, the timer interrupt signal is generated and sent to NVIC to inform CPU. In this mode, the timer controller operates counting and compares with CMPDAT value periodically until the CNTEN bit is cleared by user.

6.7.5.5 Toggle-Output Mode

If the timer controller is configured in 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[22]) 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.

6.7.5.6 Continuous Counting Mode

If the timer controller is configured in 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 set, the timer interrupt signal is generated and sent to NVIC to inform CPU. User can change different CMPDAT value immediately without disabling timer counting and restarting timer counting in this mode.

For example, the CMPDAT value is set as 80, first. The TIF will be set to 1 when the CNT value is equal to 80, the timer counter is kept counting and the CNT value will not go 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 be set to 1 again when the CNT value reaches 200. At last, user programs CMPDAT as 500 and clears TIF, and the TIF will be set to 1 again when the CNT value reaches 500.

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

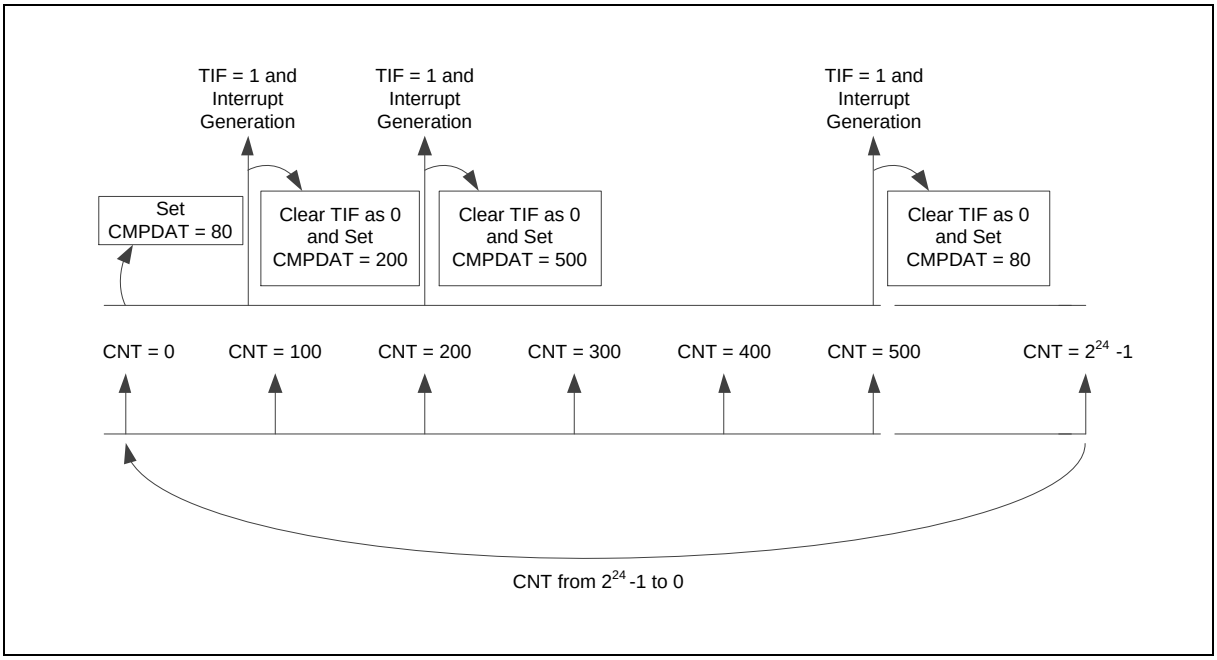


Figure 6.7-3 Continuous Counting Mode

6.7.5.7 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 pin.

If ECNTSSEL (TIMERx_EXTCTL[16]) is 1, the event counter source will generate by USB device detect the start-of-frame (SOF) packet. Please refer USB device specifications.

6.7.5.8 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, LIRC, or ACMP. 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[16]) 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[2:1]).

In event capture mode, if user did 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.

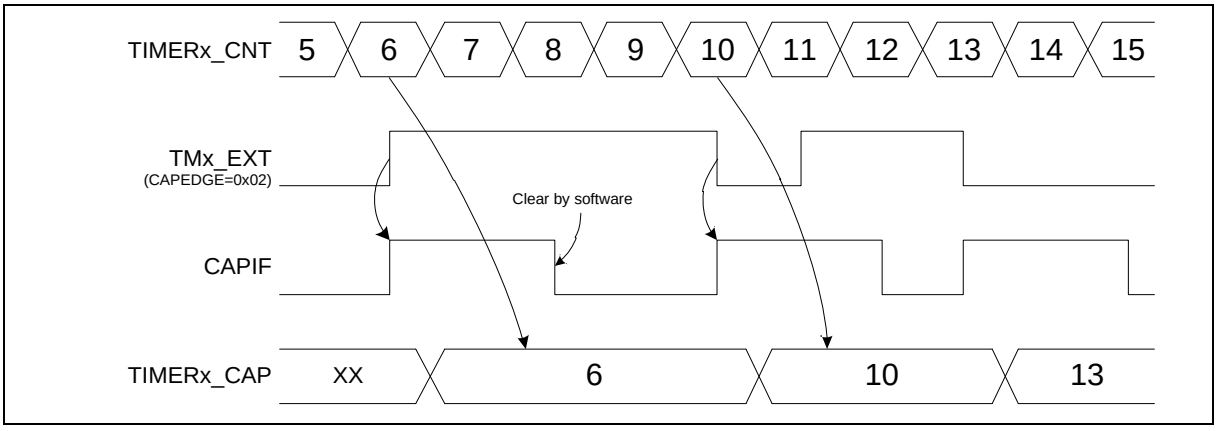


Figure 6.7-4 External Capture Mode

If CAPSRC (TIMERx_CTL[16]) 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 is 000, ACMP1 if INTERCAPSEL is 001, or LIRC if INTERCAPSEL is 101.

6.7.5.9 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 as 1. User must set CAPSRC and INTERCAPSEL to select TMx_EXT transition, internal ACMPx output signal, or LIRC to trigger reset counter value.

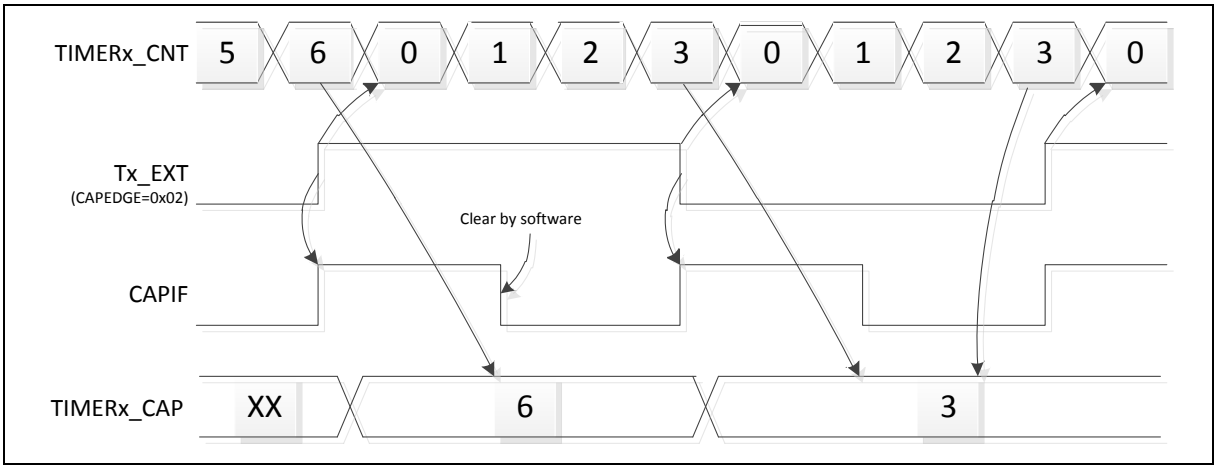


Figure 6.7-5 Reset Counter Mode

6.7.5.10 Timer Trigger Function

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

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

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

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

When the TRGBPWM (TIMERx_CTL[9]) is set, if the timer interrupt signal is generated, the timer controller will trigger BPWM.

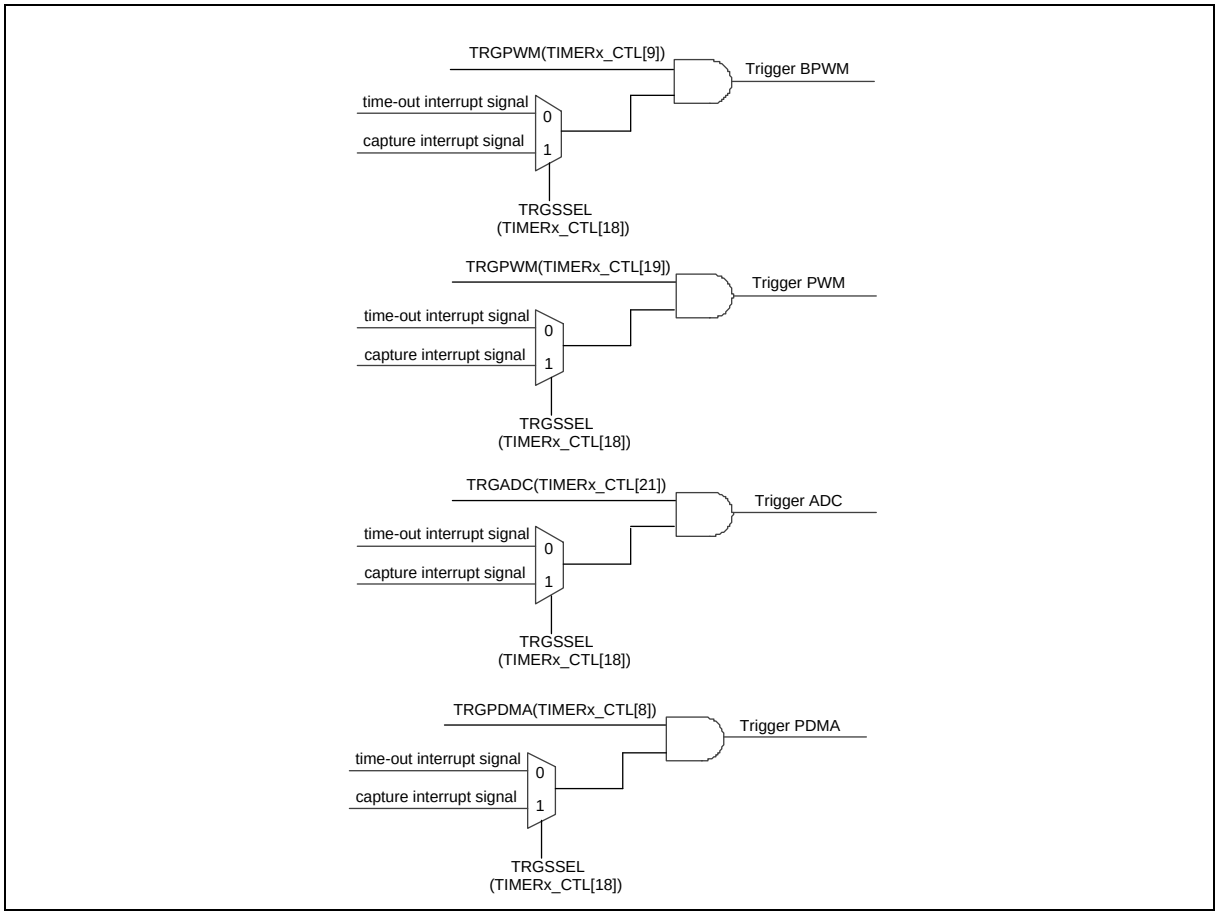


Figure 6.7-6 Internal Timer Trigger

6.7.5.11 Inter-Timer Trigger Capture Mode

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

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

Start Trigger

While INTRGEN (TIMERx_CTL[10]) in Timer0/2 is set, the Timer0/2 will make a rising-edge transition of INTR_TMR_TRG while Timer0/2 24-bit counter value (CNT) is counting from 0x0 to 0x1 and Timer1/3 counter will start counting immediately and automatically.

Stop Trigger

When Timer0/2 CNT reaches the Timer0/2 CMPDAT value, the Timer0/2 will make a falling-edge transition of INTR_TMR_TRG. Then Timer0/2 counter mode function will be disabled and INTRGEN (TIMERx_CTL[10]) 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]).

The inter-timer trigger mode can be used to measure the period of external event (TMx) more precisely. Figure 6.7-7 shows the sample flow of Inter-Timer Trigger Capture Mode for Timer0 as event counting mode and Timer1 as trigger-counting capture mode.

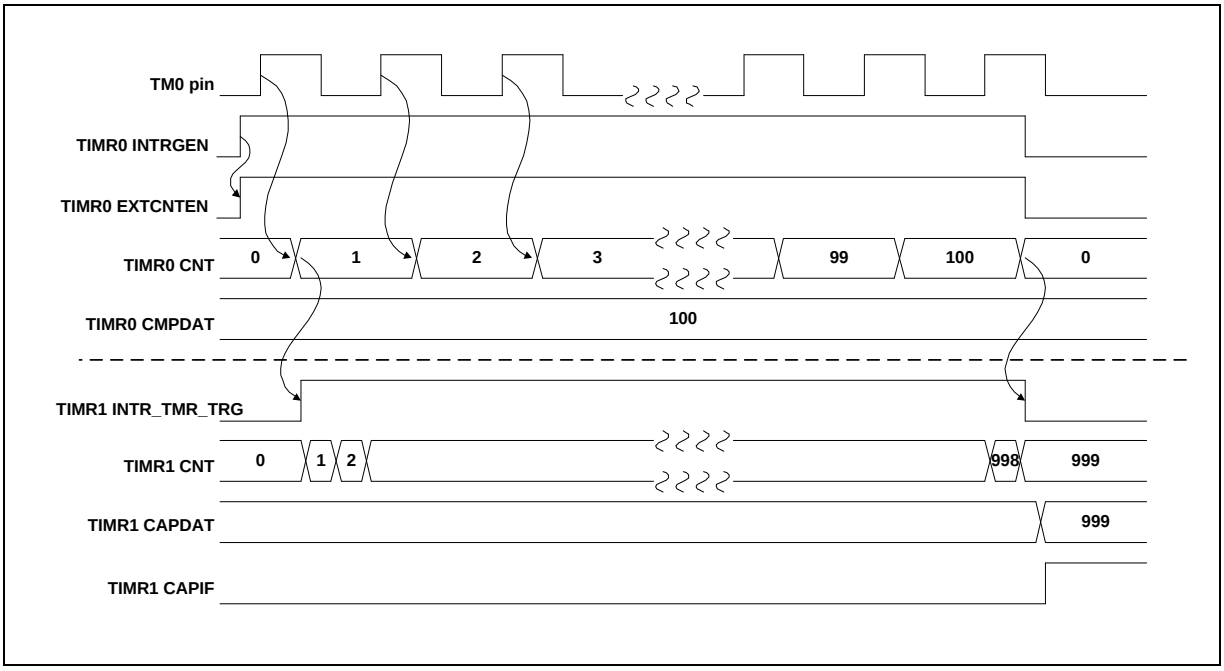


Figure 6.7-7 Inter-Timer Trigger Capture Timing

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

6.7.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
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	Timer0 Data Register	0x0000_0000
TIMER0_CAP	TMR01_BA+0x10	R	Timer0 Capture Data Register	0x0000_0000
TIMER0_EXTCTL	TMR01_BA+0x14	R/W	Timer0 External Control Register	0x0000_0000
TIMER0_EINTSTS	TMR01_BA+0x18	R/W	Timer0 External Interrupt Status Register	0x0000_0000
TIMER1_CTL	TMR01_BA+0x20	R/W	Timer1 Control Register	0x0000_0005
TIMER1_CMP	TMR01_BA+0x24	R/W	Timer1 Comparator Register	0x0000_0000
TIMER1_INTSTS	TMR01_BA+0x28	R/W	Timer1 Interrupt Status Register	0x0000_0000
TIMER1_CNT	TMR01_BA+0x2C	R	Timer1 Data Register	0x0000_0000
TIMER1_CAP	TMR01_BA+0x30	R	Timer1 Capture Data Register	0x0000_0000
TIMER1_EXTCTL	TMR01_BA+0x34	R/W	Timer1 External Control Register	0x0000_0000
TIMER1_EINTSTS	TMR01_BA+0x38	R/W	Timer1 External Interrupt Status Register	0x0000_0000
TIMER2_CTL	TMR23_BA+0x00	R/W	Timer2 Control 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	Timer2 Data Register	0x0000_0000
TIMER2_CAP	TMR23_BA+0x10	R	Timer2 Capture Data Register	0x0000_0000
TIMER2_EXTCTL	TMR23_BA+0x14	R/W	Timer2 External Control Register	0x0000_0000
TIMER2_EINTSTS	TMR23_BA+0x18	R/W	Timer2 External Interrupt Status Register	0x0000_0000
TIMER3_CTL	TMR23_BA+0x20	R/W	Timer3 Control Register	0x0000_0005
TIMER3_CMP	TMR23_BA+0x24	R/W	Timer3 Comparator Register	0x0000_0000
TIMER3_INTSTS	TMR23_BA+0x28	R/W	Timer3 Interrupt Status Register	0x0000_0000
TIMER3_CNT	TMR23_BA+0x2C	R	Timer3 Data Register	0x0000_0000
TIMER3_CAP	TMR23_BA+0x30	R	Timer3 Capture Data Register	0x0000_0000

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

6.7.7 Register Description

Timer Control Register (TIMERx_CTL)

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

31	30	29	28	27	26	25	24
ICEDEBUG	**CNTEN**	**INTEN**	**OPMODE**		**RSTCNT**	**ACTSTS**	**EXTCNTEN**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
WKEN	**TGLPINSEL**	**TRGADC**	**Reserved**	**TRGPWM**	**TRGSSEL**	**Reserved**	**CAPSRC**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**INTRGEN**	**TRGBPWM**	**TRGPDMA**
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]	**RSTCNT**	**Timer Counter Reset Bit** Setting this bit will reset the 24-bit up counter value CNT (TIMERx_CNT[23:0]) and also force CNTEN (TIMERx_CTL[30]) to 0 if ACTSTS (TIMERx_CTL[25]) is 1. 0 = No effect. 1 = Reset internal 8-bit prescale counter, 24-bit up counter value and CNTEN bit. **Note:** This bit will be auto cleared.
[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 1:** When timer is used as an event counter, this bit should be set to 1 and select PCLKx (x=0~1) as timer clock source. **Note 2:** When TMR0/TMR2 INTRGEN is set to 1, this bit is forced to 1. When INTRGEN is 1 and TMR1/TMR3 CAPIF (TIMERx_EINTSTS[0]) is 1, this bit is forced to 0.
[23]	**WKEN**	**Wake-up Function Enable Bit** If this bit is set to 1, while timer interrupt flag TIF (TIMERx_INTSTS[0]) is 1 and INTEN (TIMERx_CTL[29]) is enabled, the timer interrupt signal will generate a wake-up trigger event to CPU. 0 = Wake-up function Disabled if timer interrupt signal generated. 1 = Wake-up function Enabled if timer interrupt signal generated.
[22]	**TGLPINSEL**	**Toggle-output Pin Select** 0 = Toggle mode output to Tx (Timer Event Counter Pin). 1 = Toggle mode output to Tx_EXT (Timer External Capture Pin).
[21]	**TRGADC**	**Trigger ADC Enable Bit** If this bit is set to 1, timer time-out interrupt or capture interrupt can trigger ADC. 0 = Timer interrupt trigger ADC Disabled. 1 = Timer interrupt trigger ADC Enabled. **Note:** If TRGSSEL (TIMERx_CTL[18]) = 0, time-out interrupt signal will trigger ADC. If TRGSSEL (TIMERx_CTL[18]) = 1, capture interrupt signal will trigger ADC.
[20]	**Reserved**	Reserved.
[19]	**TRGPWM**	**Trigger PWM Enable Bit** If this bit is set to 1, timer time-out interrupt or capture interrupt can trigger PWM. 0 = Timer interrupt trigger PWM Disabled. 1 = Timer interrupt trigger PWM Enabled. **Note:** If TRGSSEL (TIMERx_CTL[18]) = 0, time-out interrupt signal will trigger PWM. If TRGSSEL (TIMERx_CTL[18]) = 1, capture interrupt signal will trigger PWM.
[18]	**TRGSSEL**	**Trigger Source Select Bit** This bit is used to select trigger source is from Timer time-out interrupt signal or capture interrupt signal.

		0 = Timer time-out interrupt signal is used to trigger PWM, BPWM, ADC and PDMA. 1 = Capture interrupt signal is used to trigger PWM, BPWM, ADC and PDMA.
[17]	**Reserved**	Reserved.
[16]	**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 or LIRC. User can set INTERCAPSEL (TIMERx_EXTCTL[10:8]) to decide which internal ACMP output signal or LIRC as timer capture source.
[15:11]	**Reserved**	Reserved.
[10]	**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 mode Disabled. 1 = Inter-Timer Trigger mode Enabled. **Note:** For Timer1/3, this bit is ignored and the read back value is always 0.
[9]	**TRGBPWM**	**Trigger BPWM Enable Bit** If this bit is set to 1, timer time-out interrupt or capture interrupt can trigger BPWM. 0 = Timer interrupt trigger BPWM Disabled. 1 = Timer interrupt trigger BPWM Enabled. **Note:** If TRGSSEL (TIMERx_CTL[18]) = 0, time-out interrupt signal will trigger BPWM. If TRGSSEL (TIMERx_CTL[18]) = 1, capture interrupt signal will trigger BPWM.
[8]	**TRGPDMA**	**Trigger PDMA Enable Bit** If this bit is set to 1, timer time-out interrupt or capture interrupt can trigger PDMA. 0 = Timer interrupt trigger PDMA Disabled. 1 = Timer interrupt trigger PDMA Enabled. **Note:** If TRGSSEL (TIMERx_CTL[18]) = 0, time-out interrupt signal will trigger PDMA. If TRGSSEL (TIMERx_CTL[18]) = 1, capture interrupt signal will trigger PDMA.
[7:0]	**PSC**	**Prescale Counter** Timer input clock or event source is divided by (PSC+1) before it is fed to the timer up counter. If this field is 0 (PSC = 0), then there is no scaling. **Note:** Updating 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+0x24	R/W	Timer1 Comparator Register	0x0000_0000
TIMER2_CMP	TMR23_BA+0x04	R/W	Timer2 Comparator Register	0x0000_0000
TIMER3_CMP	TMR23_BA+0x24	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+0x28	R/W	Timer1 Interrupt Status Register	0x0000_0000
TIMER2_INTSTS	TMR23_BA+0x08	R/W	Timer2 Interrupt Status Register	0x0000_0000
TIMER3_INTSTS	TMR23_BA+0x28	R/W	Timer3 Interrupt Status Register	0x0000_0000

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

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

Timer Data Register (TIMERx_CNT)

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

31	30	29	28	27	26	25	24
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	
[31:24]	**Reserved**	Reserved.
[23:0]	**CNT**	**Timer Data Register** 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.

Timer Capture Data Register (TIMERx_CAP)

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

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, CAPFUNCS (TIMERx_EXTCTL[4]) bit is 0, and a transition on TMx_EXT pin matched the CAPEDGE (TIMERx_EXTCTL[2:1]) setting, CAPIF (TIMERx_EINTSTS[0]) will set to 1 and the current timer counter value CNT (TIMERx_CNT[23:0]) will be auto-loaded into this CAPDAT field.

Timer External Control Register (TIMERx_EXTCTL)

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

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

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**ECNTSSEL**	**Event Counter Source Selection to Trigger Event Counter Function** 0 = Event Counter input source is from TMx (x= 0~3) pin. 1 = Event Counter input source is from USB internal SOF output signal.
[15:11]	**Reserved**	Reserved.
[10:8]	**INTERCAPSEL**	**Internal Capture Source Selection to Trigger Capture Function** 000 = Capture Function source is from internal ACMP0 output signal. 001 = Capture Function source is from internal ACMP1 output signal. 101 = Capture Function source is from LIRC. Others = Reserved. **Note:** these bits only available when CAPSRC (TIMERx_CTL[16]) 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, LIRC, or ACMP detection Interrupt Disabled. 1 = TMx_EXT (x= 0~3) pin, LIRC, or ACMP detection Interrupt Enabled. **Note:** CAPIEN is used to enable timer external interrupt. If CAPIEN enabled, timer will rise an interrupt when CAPIF (TIMERx_EINTSTS[0]) is 1. For example, while CAPIEN = 1, CAPEN = 1, and CAPEDGE = 00, a 1 to 0 transition on the Tx_EXT (x= 0~3) pin, or ACMP 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:** TMR1/TMR3 CAPEN will be forced to 1 when TMR0/TMR2 INTRGEN is enabled.
[2:1]	**CAPEDGE**	**Timer External Capture Pin Edge Detect** 00 = A Falling edge on Tx_EXT (x= 0~3) pin, LIRC or ACMPx (x=0~1) will be detected. 01 = A Rising edge on Tx_EXT (x= 0~3) pin, LIRC or ACMPx (x=0~1) will be detected. 10 = Either Rising or Falling edge on Tx_EXT (x= 0~3) pin, LIRC or ACMPx (x=0~1) will be detected. 11 = 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+0x38	R/W	Timer1 External Interrupt Status Register	0x0000_0000
TIMER2_EINTSTS	TMR23_BA+0x18	R/W	Timer2 External Interrupt Status Register	0x0000_0000
TIMER3_EINTSTS	TMR23_BA+0x38	R/W	Timer3 External Interrupt Status Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**CAPIF**	**Timer External Capture Interrupt Flag** This bit indicates the timer external capture interrupt flag status. 0 = TMx_EXT (x= 0~3) pin interrupt did not occur. 1 = TMx_EXT (x= 0~3) pin 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 Tx_EXT (x= 0~3) pin 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.

6.8 Watchdog Timer (WDT)

6.8.1 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.

6.8.2 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 416us ~ 27.3 s if WDT_CLK = 38.4 kHz (LIRC).
- System kept in reset state for a period of (1 / WDT_CLK) * 63
- Supports selectable WDT reset delay period, including 1026、130、18 or 3 WDT_CLK reset delay period
- Supports to force WDT enabled after chip powered on or reset by setting CWDTEN[2:0] in Config0 register
- Supports WDT time-out wake-up function only if WDT clock source is selected as LIRC or LXT.

6.8.3 Block Diagram

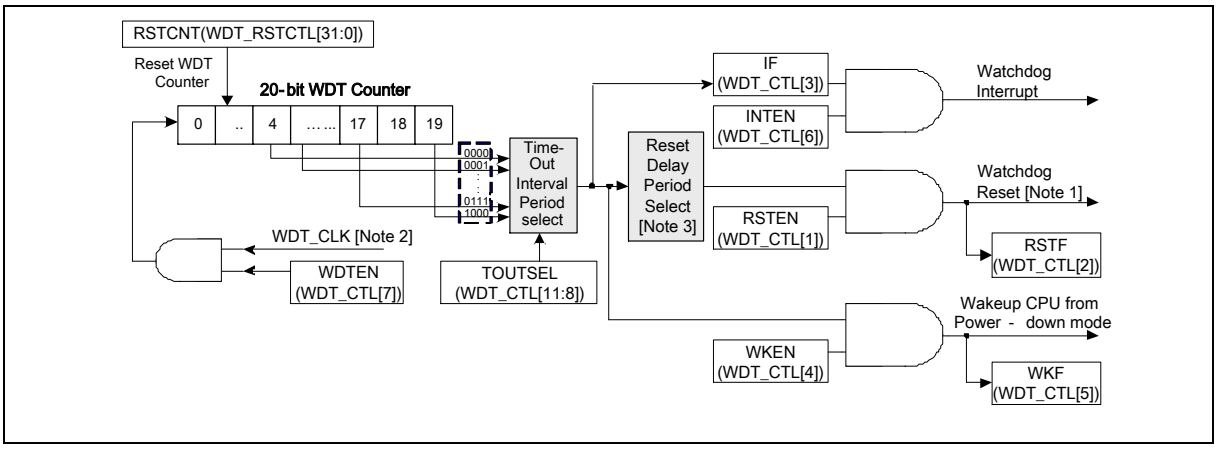


Figure 6.8-1 Watchdog Timer Block Diagram

Note1: WDT resets CPU and lasts 63 WDT_CLK.

Note2: Chip can be woken up by WDT time-out interrupt signal generated only if WDT clock source is selected to LIRC or LXT.

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

6.8.4 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])
  - WDT clock source can be changed only if CWDTEN[2:0] is 111.

The WDT clock control is shown in Figure 6.8-2.

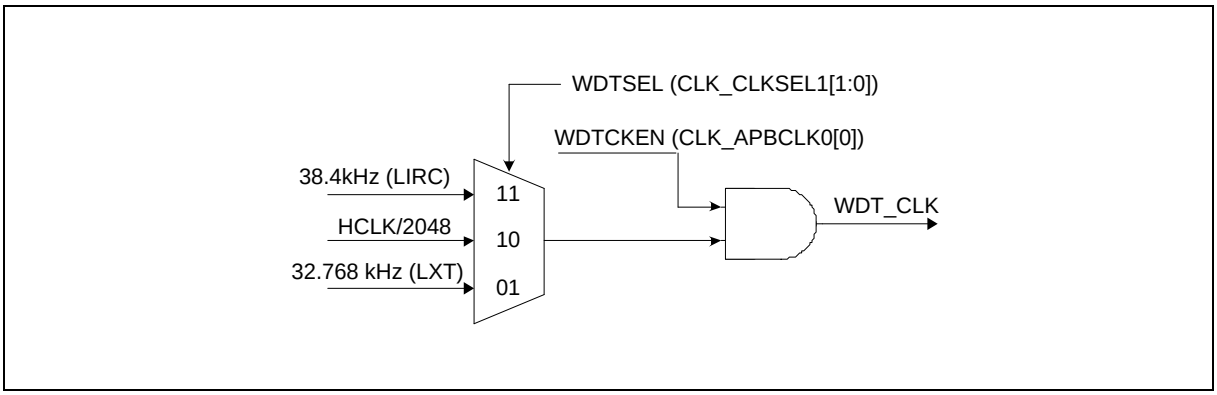


Figure 6.8-2 Watchdog Timer Clock Control

6.8.5 Functional Description

The WDT includes an 20-bit free running up counter with programmable time-out intervals. Table 6.8-1 shows the WDT time-out interval period selection and Figure 6.8-3 shows the WDT time-out interval and reset period timing.

6.8.5.1 WDT Time-out Interrupt

Setting WDTEN (WDT_CTL[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 (WDT_CTL[11:8]). When the WDT up counter reaches the TOUTSEL (WDT_CTL[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.

6.8.5.2 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 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 reset state immediately. Refer to Figure 6.8-3. 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}
000	$2^4 * T_{WDT}$	(3/18/130/1026) * T_{WDT}
001	$2^6 * T_{WDT}$	(3/18/130/1026) * T_{WDT}
010	$2^8 * T_{WDT}$	(3/18/130/1026) * T_{WDT}
011	$2^{10} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
100	$2^{12} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
101	$2^{14} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
110	$2^{16} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
111	$2^{18} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
1000	$2^{20} * T_{WDT}$	(3/18/130/1026) * T_{WDT}

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

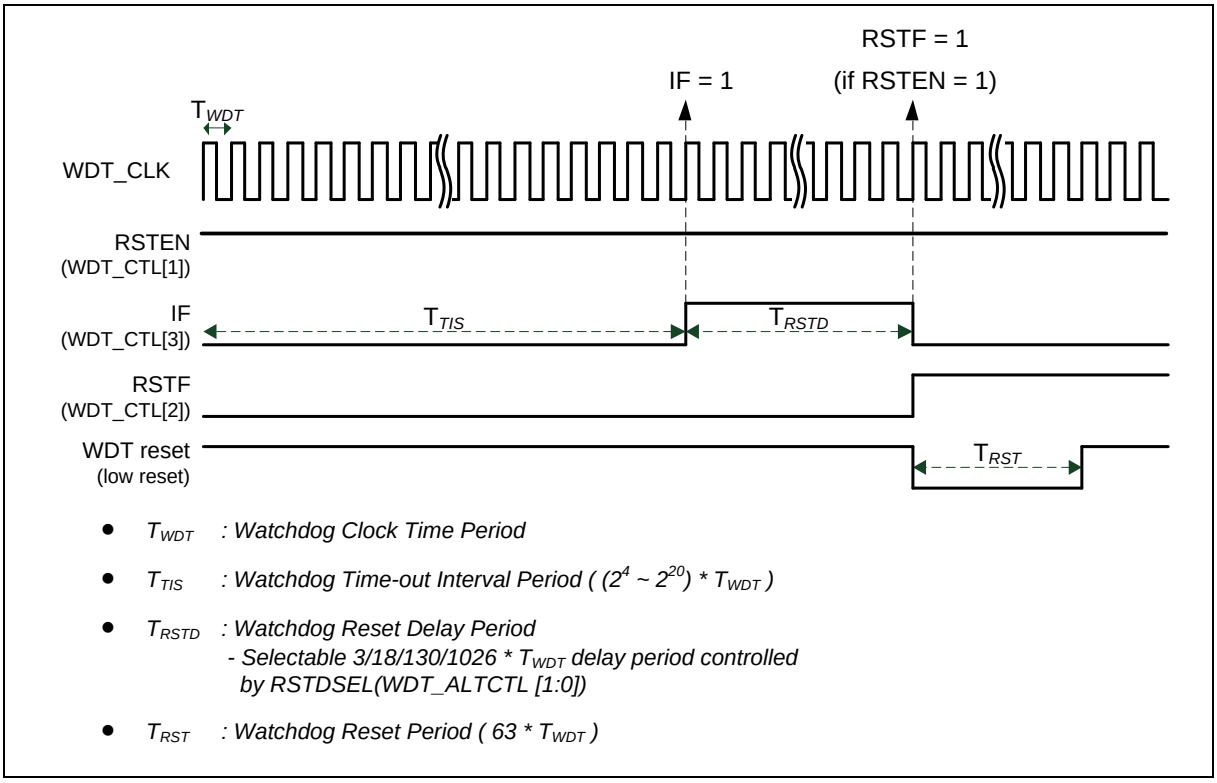


Figure 6.8-3 Watchdog Timer Time-out Interval and Reset Period Timing

6.8.5.3 WDT Wake-up

If WDT clock source is selected to LIRC 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.

6.8.5.4 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.

6.8.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
WDT Base Address: **WDT_BA = 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

6.8.7 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**	**Reserved**

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 four 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]	**Reserved**	Reserved.

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 writing 0x00005aa5 to RSTCNT (WDT_RSTCNT[31:0]) 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**.**

6.9 Window Watchdog Timer (WWDT)

6.9.1 Overview

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

6.9.2 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

6.9.3 Block Diagram

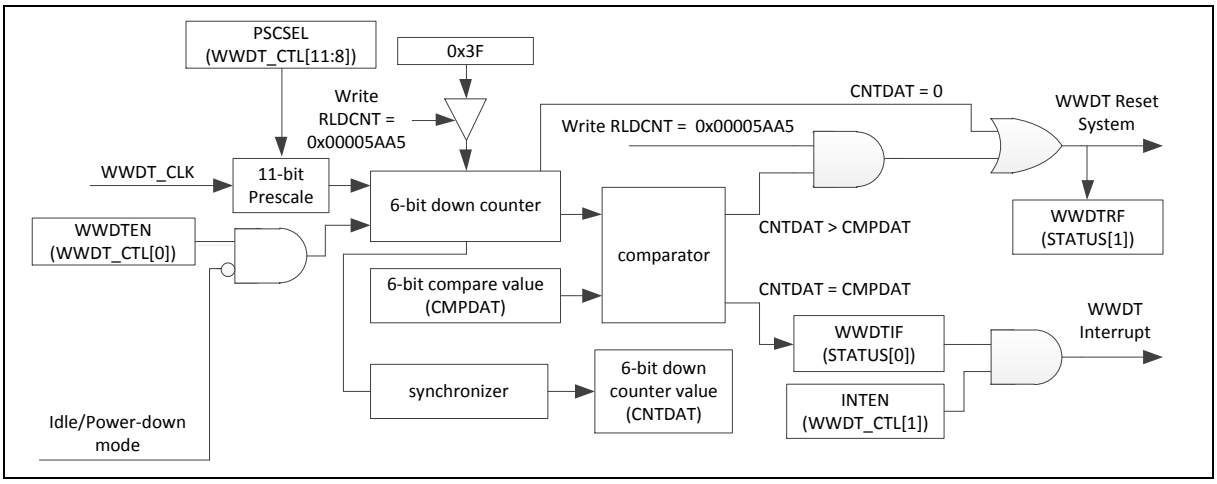


Figure 6.9-1 WWDT Block Diagram

6.9.4 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 6.9-2.

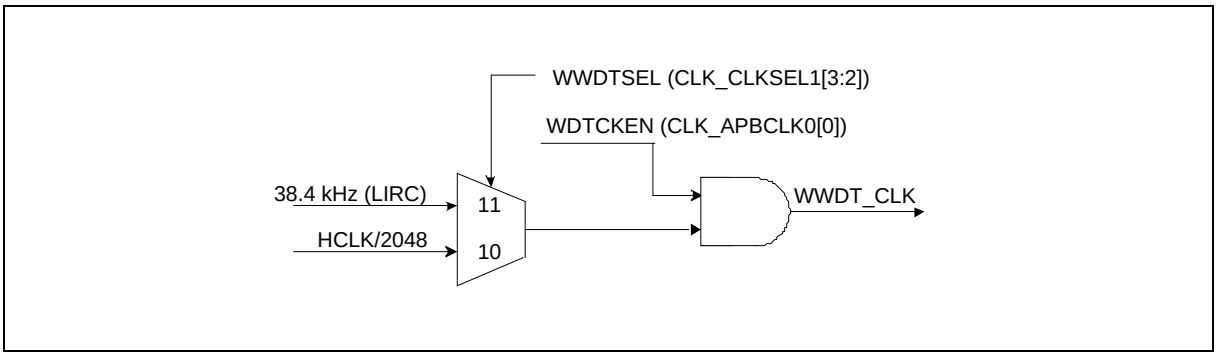


Figure 6.9-2 WWDT Clock Control

6.9.5 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 6.9-1.

PSCSEL	Prescaler Value	Max. Time-Out Period	Max. Time-Out Interval (WWDT_CLK=38.4 kHz)
0000	1	1 * 64 * T_{WWDT}	1.666667 ms
0001	2	2 * 64 * T_{WWDT}	3.33333 ms
0010	4	4 * 64 * T_{WWDT}	6.6667 ms
0011	8	8 * 64 * T_{WWDT}	13.333 ms
0100	16	16 * 64 * T_{WWDT}	26.667 ms
0101	32	32 * 64 * T_{WWDT}	53.333 ms
0110	64	64 * 64 * T_{WWDT}	106.66 ms
0111	128	128 * 64 * T_{WWDT}	213.33 ms
1000	192	192 * 64 * T_{WWDT}	320 ms
1001	256	256 * 64 * T_{WWDT}	426.66 ms
1010	384	384 * 64 * T_{WWDT}	640 ms
1011	512	512 * 64 * T_{WWDT}	853.33 ms
1100	768	768 * 64 * T_{WWDT}	1.28 s
1101	1024	1024 * 64 * T_{WWDT}	1.706 s
1110	1536	1536 * 64 * T_{WWDT}	2.56 s
1111	2048	2048 * 64 * T_{WWDT}	3.413 s

Table 6.9-1 WWDT Prescaler Value Selection

6.9.5.1 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.

6.9.5.2 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.

6.9.5.3 WWDT Reset System

Figure 6.9-3 shows three cases of WWDT reset and reload behavior.

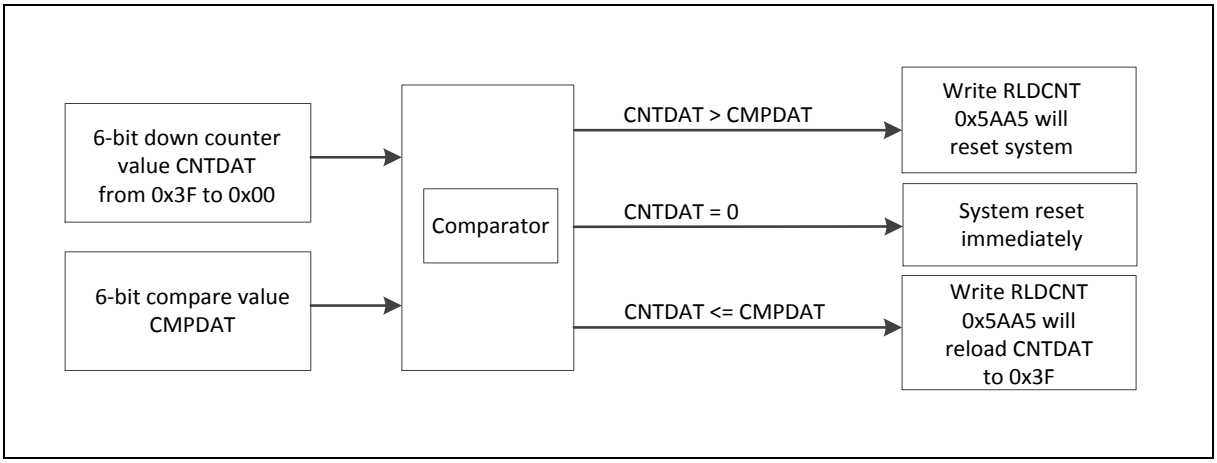


Figure 6.9-3 WWDT Reset and Reload Behavior

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 6.9-4.

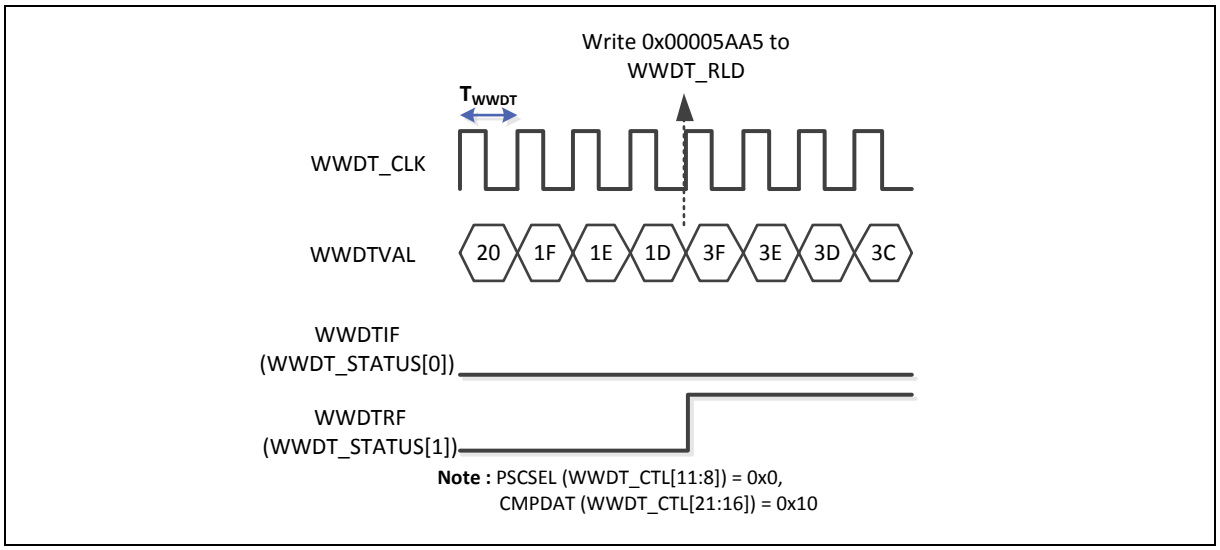


Figure 6.9-4 WWDT Reload Counter When CNTDAT > CMPDAT

When WWDTIF (WWDT_STATUS[0]) is generated, user must reload WWDT counter value to 0x3F by writing 0x00005AA5 to WWDT_RLDCNT register, and also to prevent WWDT counter value reached to 0 and generate WWDT reset system signal to info system reset. Figure 6.9-5 shows the waveform of WWDT reload counter when CNTDAT < CMPDAT and Figure 6.9-6 shows WWDT generate reset system signal (WWDTRF) if user doesn't write 0x00005AA5 to WWDT_RLD before WWDT counter value reach to 0.

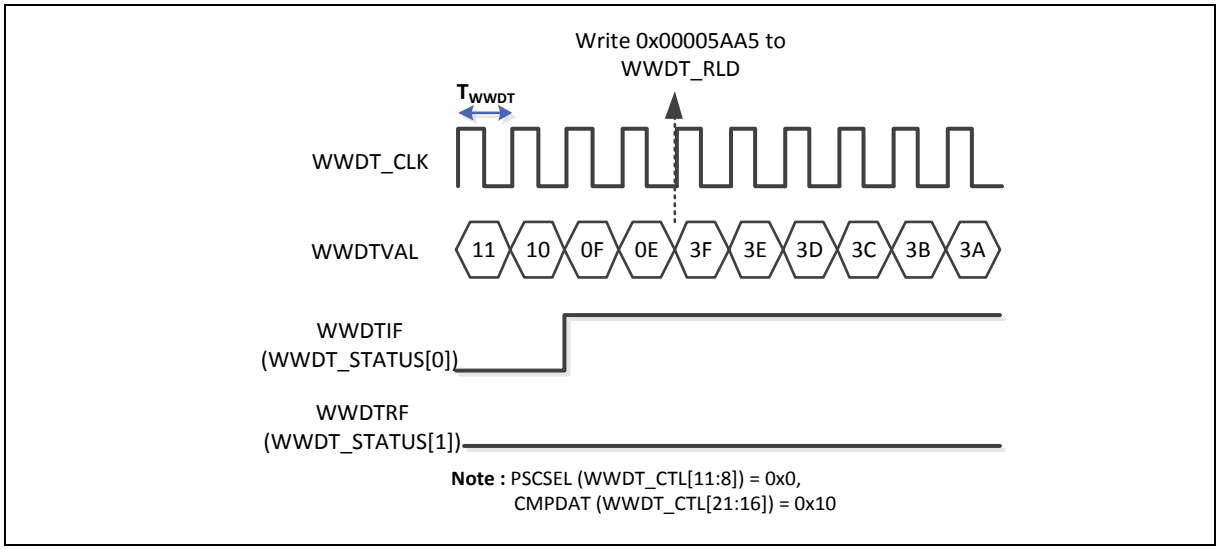


Figure 6.9-5 WWDT Reload Counter When WWDT_CNT < WINCMP

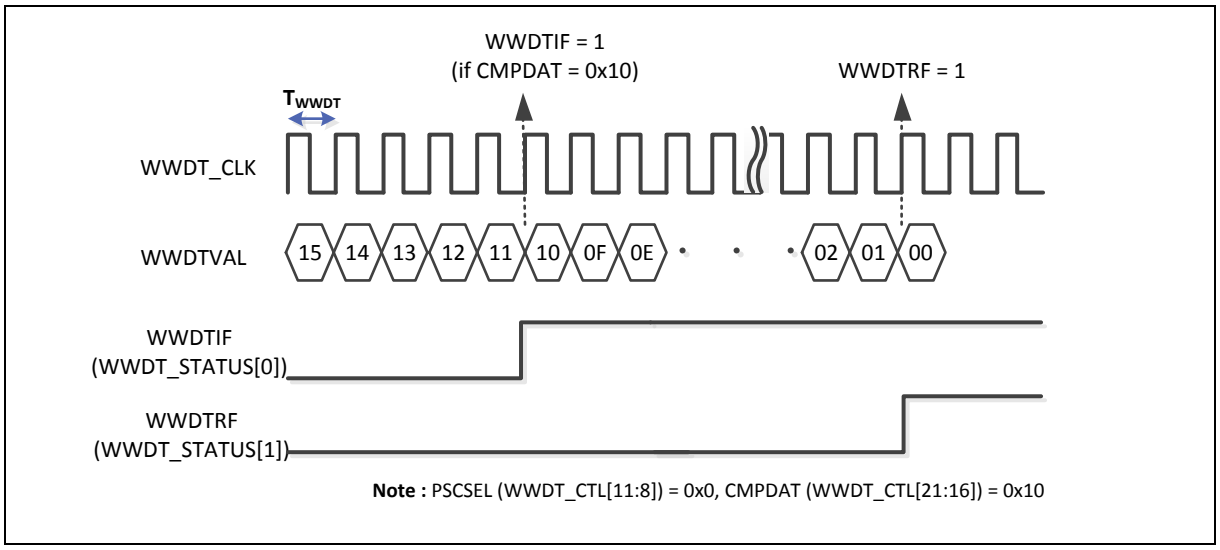


Figure 6.9-6 WWDT Interrupt and Reset Signals

6.9.5.4 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 6.9-2.

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 6.9-2 CMPDAT Setting Limitation

6.9.5.5 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.

6.9.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
WWDT Base Address: **WWDT_BA = 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

6.9.7 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 Register** Set this register to adjust the valid reload window. **Note:** User can only write WWDT_RLDCNT register to reload WWDT counter value when current WWDT counter value between 0 and CMPDAT. If user writes WWDT_RLDCNT register when current WWDT counter value larger than CMPDAT, WWDT reset signal will generate immediately.
[15:12]	**Reserved**	Reserved.
[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.

6.10 Real Time Clock (RTC)

6.10.1 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.

6.10.2 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.

6.10.3 Block Diagram

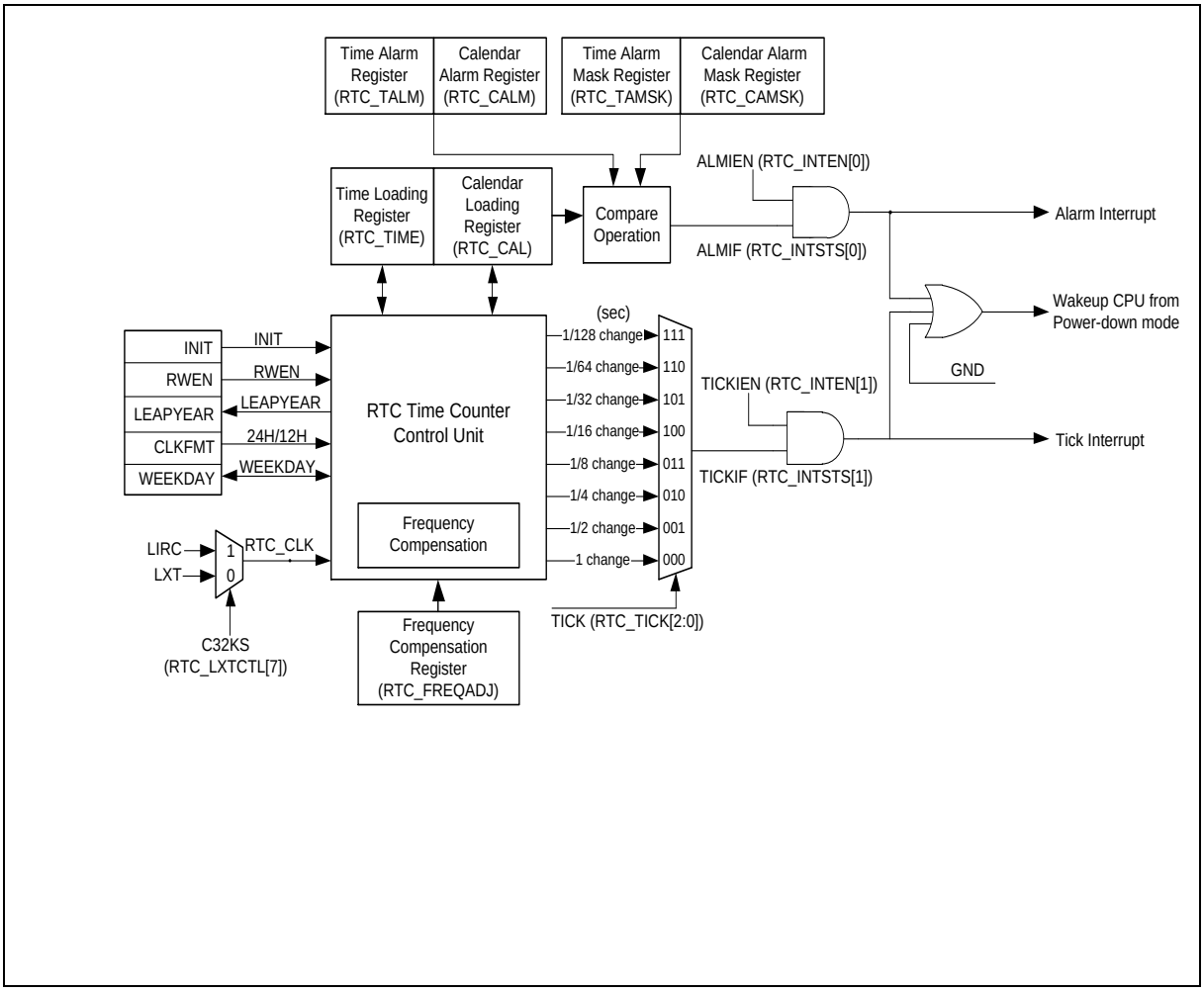


Figure 6.10-1 RTC Block Diagram

6.10.4 Basic Configuration

- Clock Source Configuration
  - The RTC controller clock source is enabled by RTCCKEN (APBCLK0[1]) and RTC Time Counter source is selected by RTC_LXTCTL[7] LXT or LIRC.

6.10.5 Functional Description

6.10.5.1 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 INIT register is written as 0xa5eb1357, the RTC will be in normal active state permanently. User can read ACTIVE(RTC_INIT[0]) to check the RTC is at normal active state or reset state.

The RTC control registers access attributes are shown in Table 6.10-1.

Register	ACTIVE (RTC_INIT[0]) = 0	ACTIVE RTC_INIT[0] = 1
RTC_INIT	R/W (Note 1)	R
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_INTEN	R/W	R/W
RTC_INTSTS	R	R/W
RTC_TICK	Not available	R/W
RTC_TAMSK	Not available	R/W
RTC_CAMSK	Not available	R/W

Table 6.10-1 RTC Read/Write Enable

Note 1: The INIT bits can be wirritten and INIT[0] can be read only.

6.10.5.2 Frequency Compensation

The RTC_FREQADJ register allows user to make digital compensation to a clock input. Please follow the example and formula below to write the actual frequency of 32k crystal to RTC_FREQADJ register. Following are the compensation examples for higher or lower than 32768 Hz.

Example 1:

Frequency counter measurement : 32773.65 Hz

Integer Part: 32773 => RTC_FREQADJ[12:8] = 0x15, Refer the INTEGER(RTC_FREQADJ[13:8]) 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[13:8]) 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. In the meanwhile, user can use clock output function to check the result of RTC frequency compensation.

6.10.5.3 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.

6.10.5.4 12/24 hour Time Scale Selection

The 12/24 hour time scale selection depends on 24HEN (RTC_CLKFMT[0]). When RTC runs as 12-hour time scale mode, RTC_TIME[21] (the high bit of TENHR[1:0]) means AM/PM indication, if RTC_TIME[21] is 1, it indicates PM time message and RTC_TIME[21] is 0 indicates AM time message.) Table 6.10-2 shows RTC_TIME mapping table of 12/24 hour time scale selection.

Note: The Hour Value Write Into RTC_TIME[21:16], Messages Are Expressed In BCD Format.			
24-Hour Time Scale (24HEN = 1)		12-Hour Time Scale (PM Time + 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 6.10-212/24 Hour Time Scale Selection

6.10.5.5 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.

6.10.5.6 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.

6.10.5.7 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.

6.10.5.8 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.

6.10.5.9 Application Note

1. All data in RTC_TALM, RTC_CALM, RTC_TIME and RTC_CAL registers are all expressed in BCD format.
2. User has to make sure that the loaded values are reasonable. For example, Load RTC_CAL

as 201a (year), 13 (month), 00 (day), or RTC_CAL does not match with RTC_WEEKDAY, etc.

3. In RTC_CAL and RTC_CALM, only 2 BCD digits are used to express “year”. The 2 BCD digits of xy means 20xy, rather than 19xy or 21xy.
4. Example of 12-Hour Time Setting
   If current RTC time is PM12:59:30 in 12-Hour Time Scale mode, the RTC_TIME setting as:

   HOUR:
   RTC_TIME[21:16]: 0x32 (0x12+0x20) combined by TENHR (RTC_TIME[21:20]) is 0x3, HR (RTC_TIME[19:16]) is 0x2.

   MIN:
   RTC_TIME[14:8]: 0x59 combined by TENMIN (RTC_TIME[14:12]) is 0x5, MIN (RTC_TIME[11:8]) is 0x9.

   SEC:
   RTC_TIME[6:0]: 0x30 combined by TENSEC (RTC_TIME[6:4]) is 0x3, SEC (RTC_TIME[3:0]) is 0x0.
5. Table 6.10-3 shows registers value after both core power and battery power are 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

Table 6.10-3 Registers Value after Powered On

6.10.6 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 32K.768 kHz LXT Control Register	0x0000_0000

6.10.7 Register Description

RTC Initiation Register (RTC_INIT)

Register	Offset	R/W	Description	Reset Value
RTC_INIT	RTC_BA+0x00	R/W	RTC Initiation Register	0x0000_0000

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

Bits	Description	
[31:1]	**INIT[31:1]**	**RTC Initiation (Write Only)** When RTC block is powered on, RTC is at reset state. User has to write a number (0x a5eb1357) to INIT to make RTC 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]	**INIT[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
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	
[31:13]	**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: FREQADJ's counter will be reset for start to Compensatie when writing RTC_FREQADJ, RTC_TIME, RTC_CAL, and RTC_WEEKDAY. 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_FREQADJ , 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_FREQADJ , 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							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**24HEN**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[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_FREQADJ, 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	
[31:22]	**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 LXT Control Register (RTC_LXTCTLK)

Register	Offset	R/W	Description	Reset Value
RTC_LXTCTL	RTC_BA+0x100	R/W	RTC 32K.768 kHz LXT 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**
C32KS	**Reserved**						

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**OSC32_S**	**Clock 32K Source Selection** 0 = Clock source from LXT32K. 1 = Clock source from LIRC38K.
[6:0]	**Reserved**	Reserved.

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.

6.11 Basic PWM Generator and Capture Timer (BPWM)

6.11.1 Overview

The chip provides two BPWM generators — BPWM0 and BPWM1 as shown in Figure 6.11-1. Each BPWM supports 6 channels of BPWM output or input capture. There is a 12-bit prescaler to support flexible clock to the 16-bit BPWM counter with 16-bit comparator. The BPWM counter supports up, down and up-down counter types, all 6 channels share one counter. BPWM uses the comparator compared with counter to generate events. These events are used to generate BPWM pulse, interrupt and trigger signal for ADC to start conversion. For BPWM output control unit, it supports polarity output, independent pin mask and tri-state output enable.

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

6.11.2 Features

6.11.2.1 BPWM Function Features

- Supports maximum clock frequency up to 144 MHz.
- Supports up to two BPWM modules; each module provides 6 output channels
- Supports independent mode for BPWM output/Capture input channel
- Supports 12-bit 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 ADC in the following events:
  - BPWM counter matches 0, period value or compared value

6.11.2.2 Capture Function Features

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

6.11.3 Block Diagram

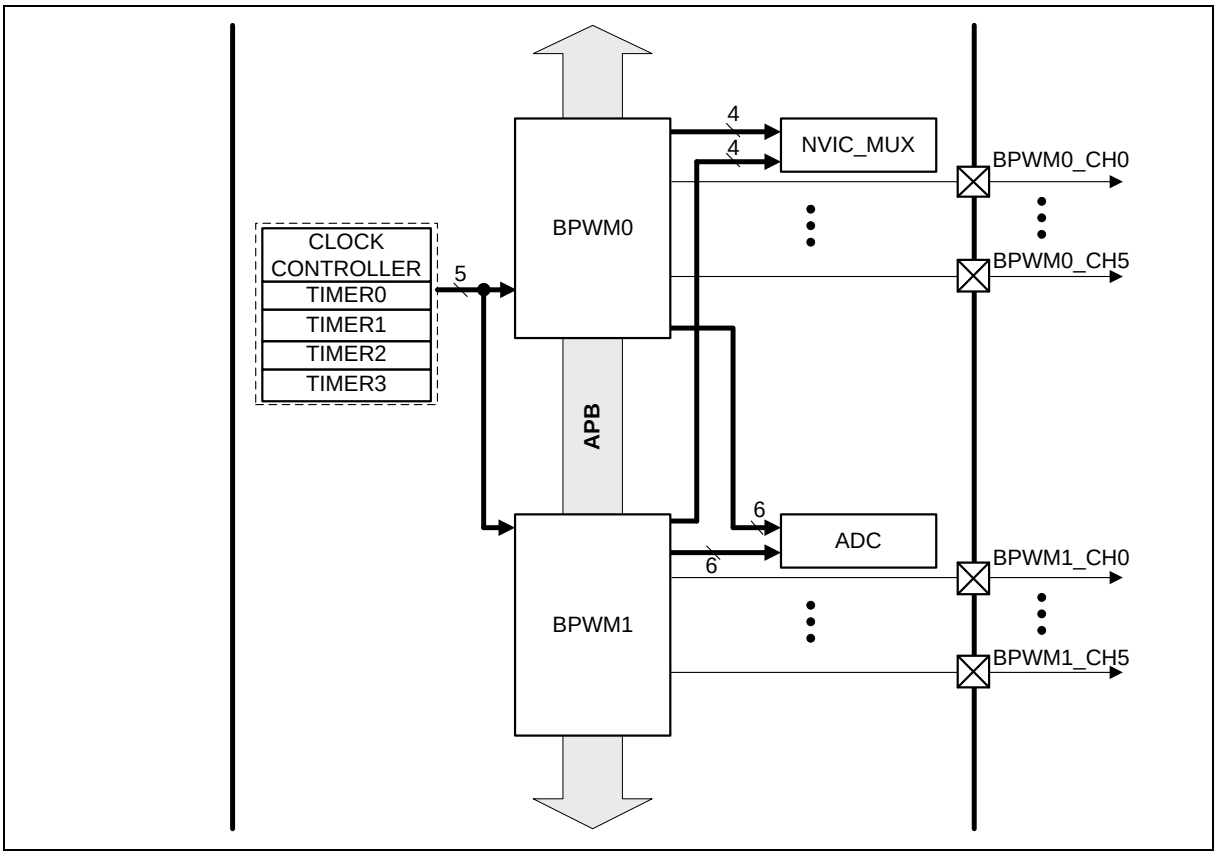


Figure 6.11-1 BPWM Generator Overview Block Diagram

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 6.11-3 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 PLL clock frequency as shown in Figure 6.11-2, and Table 6.11-1 BPWM Clock Source Control Registers Setting Table. BPWM0 and BPWM1 Clock must be selected to PLL clock by setting BPWM0SEL (CLK_CLKSEL2[8]) and BPWM1SEL (CLK_CLKSEL2[9]) to 0.

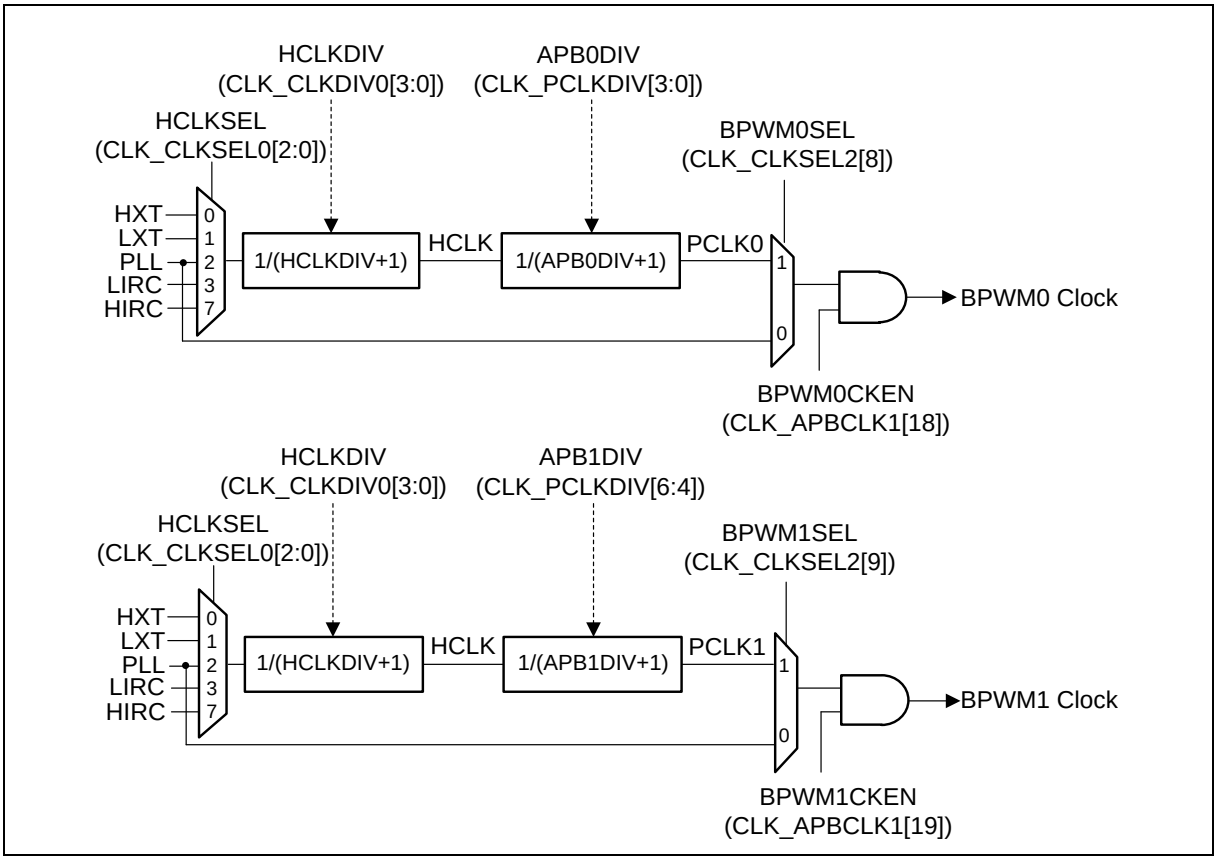


Figure 6.11-2 BPWM System Clock Source Control

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	PLL/ 2	PLL/ 2	PLL	2	1	0	0

Table 6.11-1 BPWM Clock Source Control Registers Setting Table

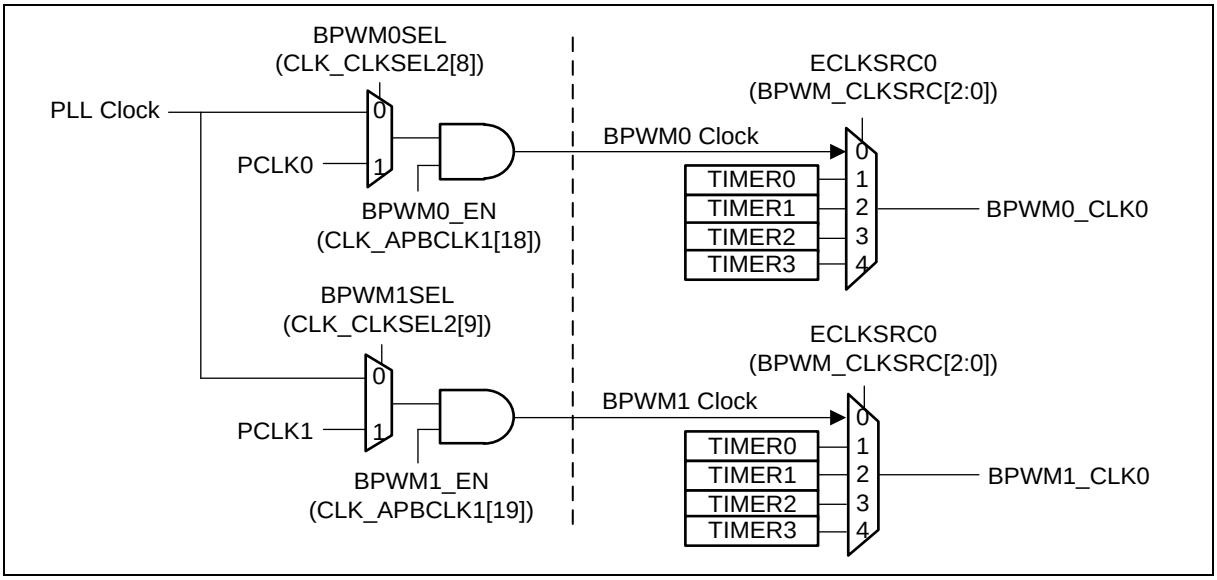


Figure 6.11-3 BPWM Clock Source Control

Figure 6.11-4 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 ADC to start conversion. Output control is used to change the BPWM pulse output state.

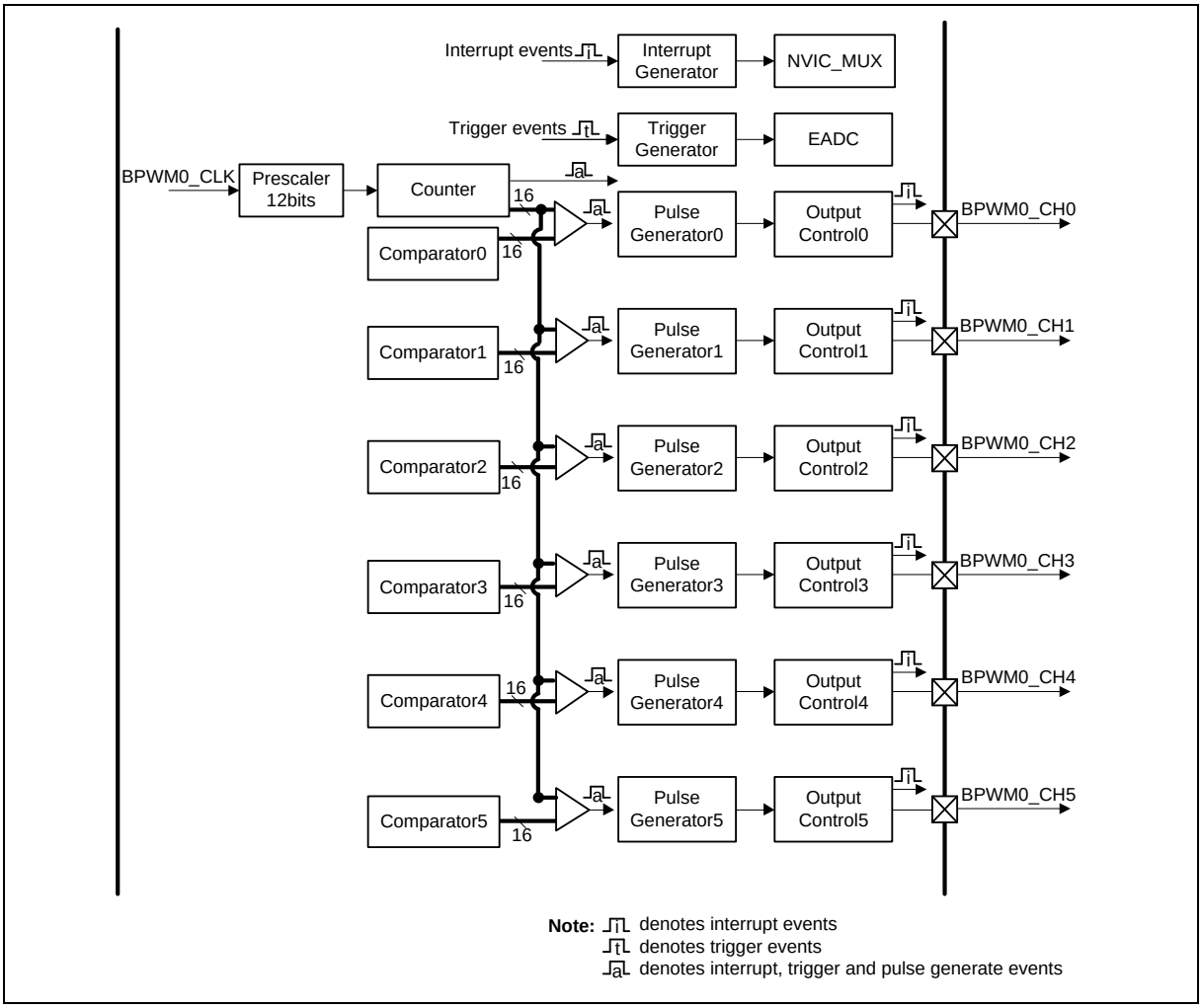


Figure 6.11-4 BPWM Independent Mode Architecture Diagram

6.11.4 Basic Configuration

6.11.4.1 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]).

6.11.4.2 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]).

6.11.5 Functional Description

6.11.5.1 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 6.11-5 shows an example of BPWM channel 0 CLKPSC waveform. The prescale counter will reload CLKPSC at the begin of the next prescale counter down-count.

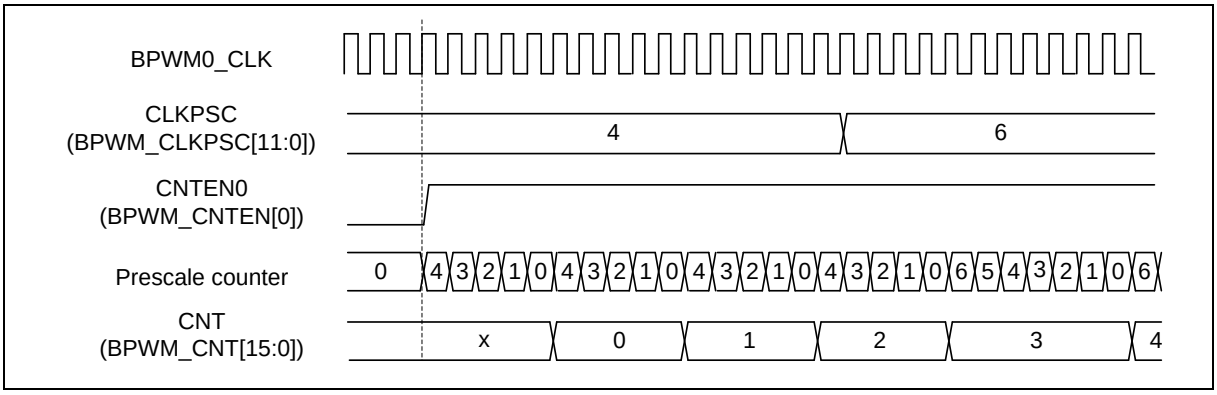


Figure 6.11-5 BPWM_CH0 CLKPSC waveform

6.11.5.2 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.

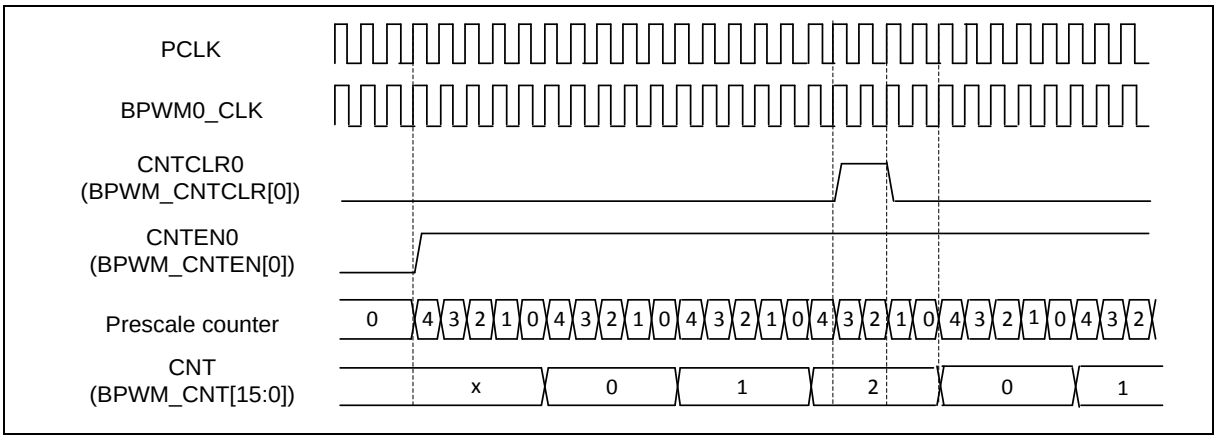


Figure 6.11-6 BPWM Counter Clear waveform

6.11.5.3 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 6.11-7.

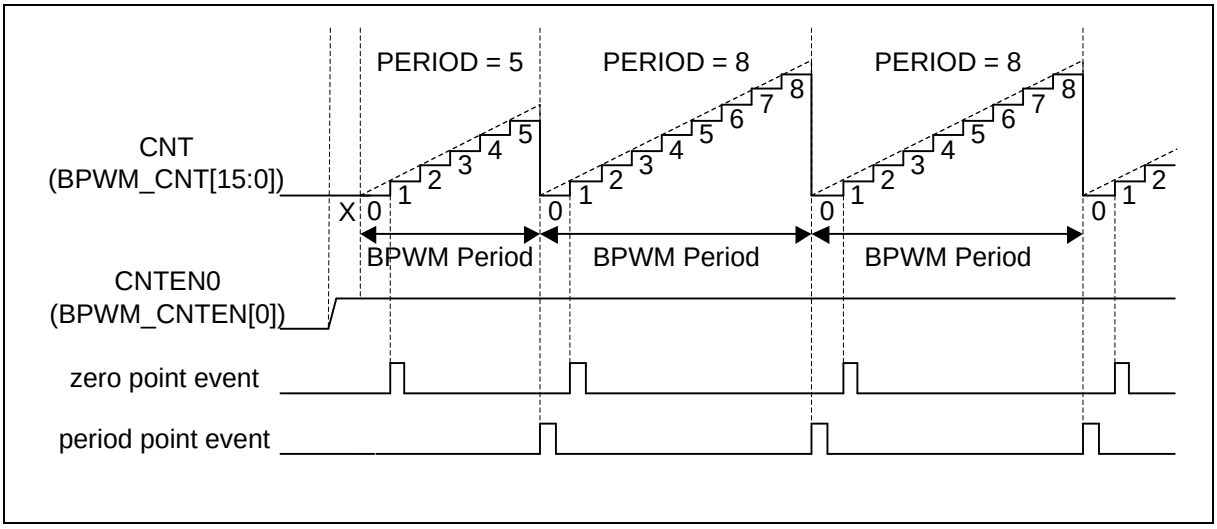


Figure 6.11-7 BPWM Up Counter Type

6.11.5.4 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 6.11-8.

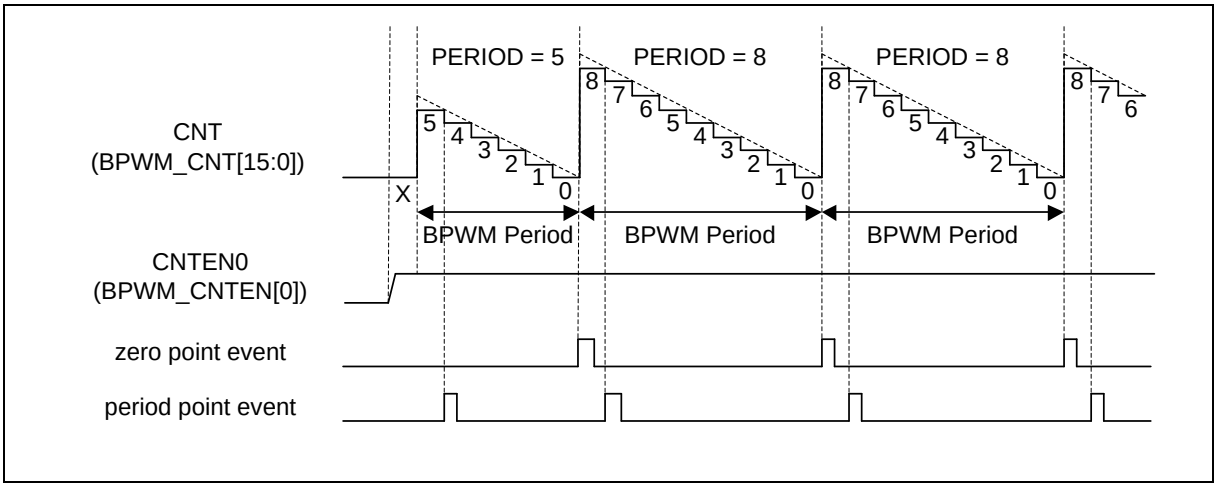
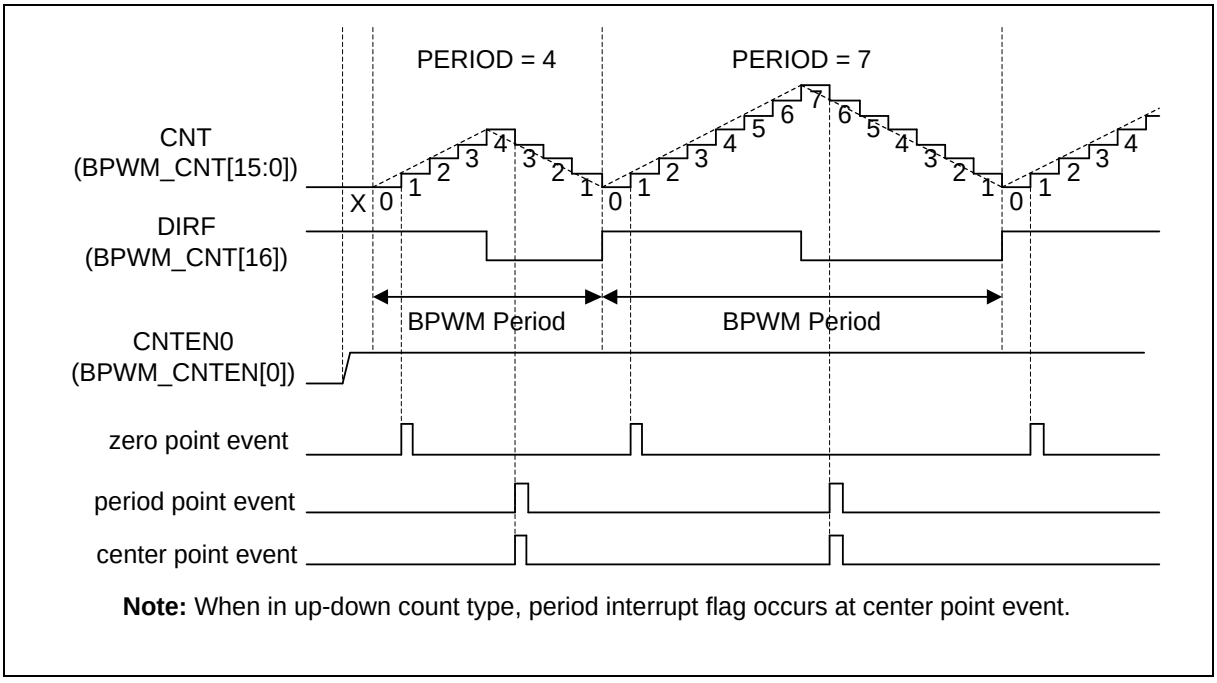


Figure 6.11-8 BPWM Down Counter Type

6.11.5.5 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 6.11-9. 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 6.11-9 BPWM Up-Down Counter Type

6.11.5.6 BPWM Comparator

The CMPDAT (BPWM_CMPDATn[15:0]) is a basic comparator register of BPWM channel n; each channel only has one CMPDAT. The CMPDAT's value is continuously compared to the counter value. When the counter is equal to compared register, BPWM generates an event and uses the event to generate BPWM pulse, interrupt or use to trigger ADC. In up-down counter type, two events will be generated in a BPWM period as shown in Figure 6.11-10.

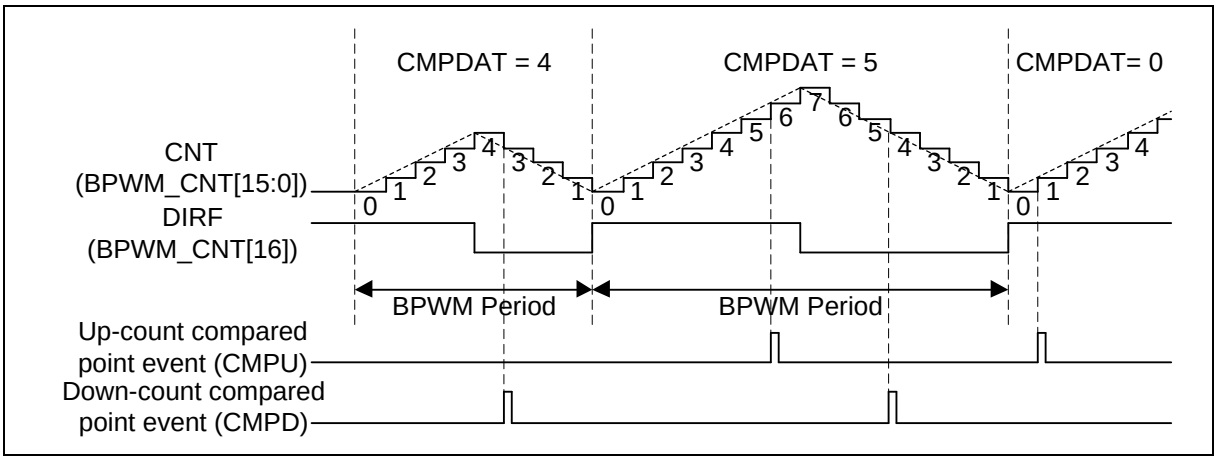


Figure 6.11-10 BPWM CMPDAT Events in Up-Down Counter Type

6.11.5.7 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 6.11-11 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 6.11-11. 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.

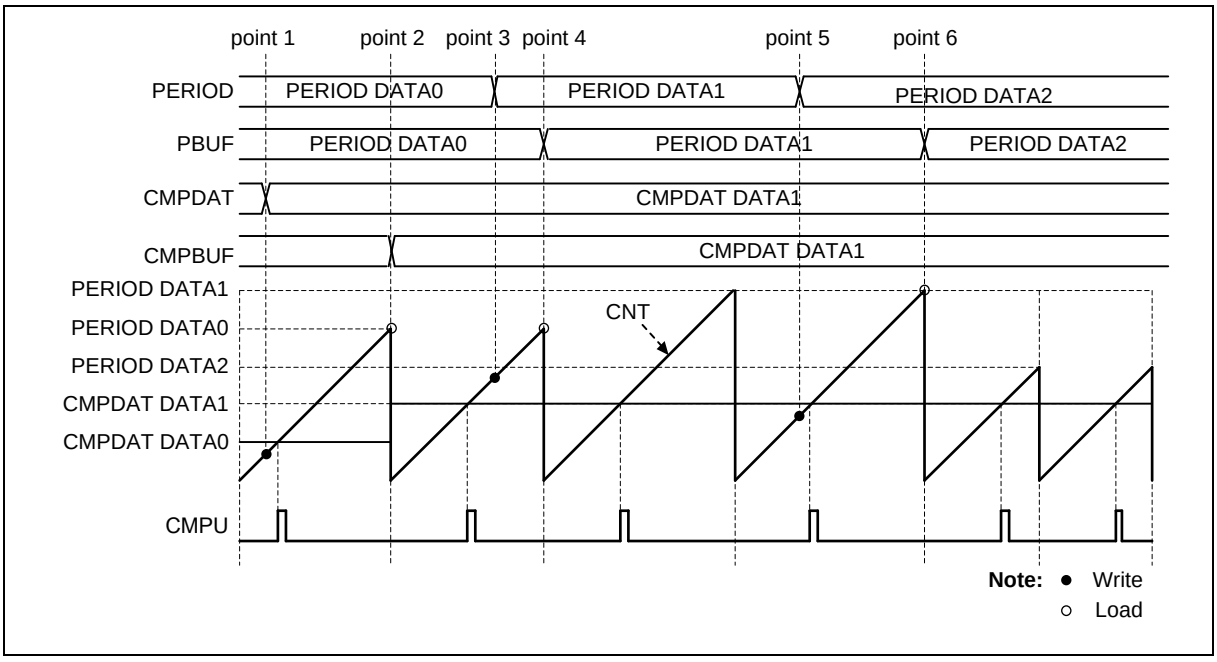


Figure 6.11-11 Period Loading Mode with Up-Counter Type

6.11.5.8 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 CNTMAX0(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 6.11-12 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.

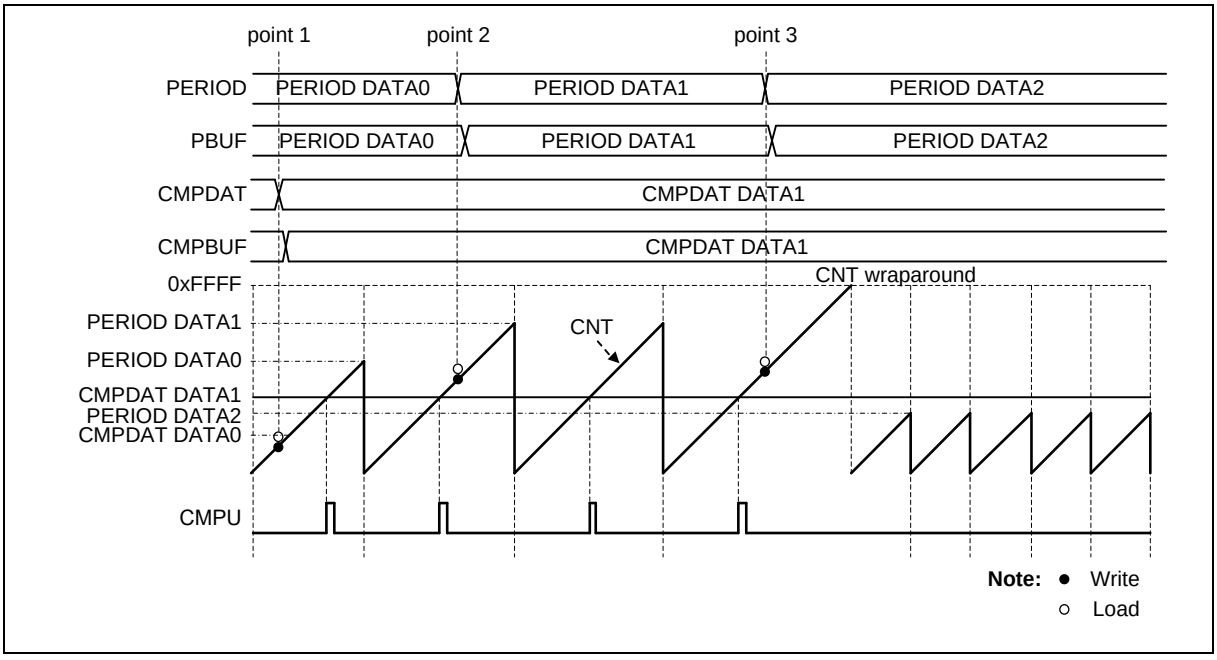


Figure 6.11-12 Immediately Loading Mode with Up-Counter Type

6.11.5.9 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 6.11-13 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.

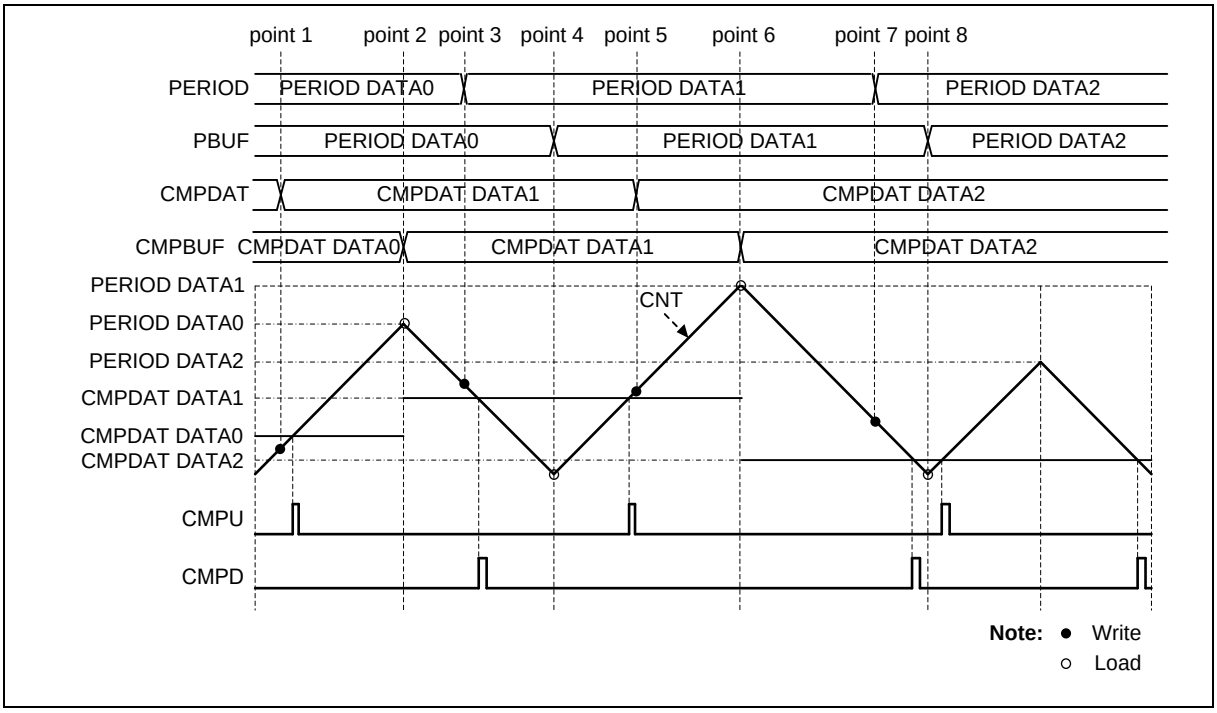


Figure 6.11-13 Center Loading Mode with Up-Down-Counter Type

6.11.5.10 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 6.11-14. In the figure, there is a comparator n to generate BPWM pulse, whrer 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.

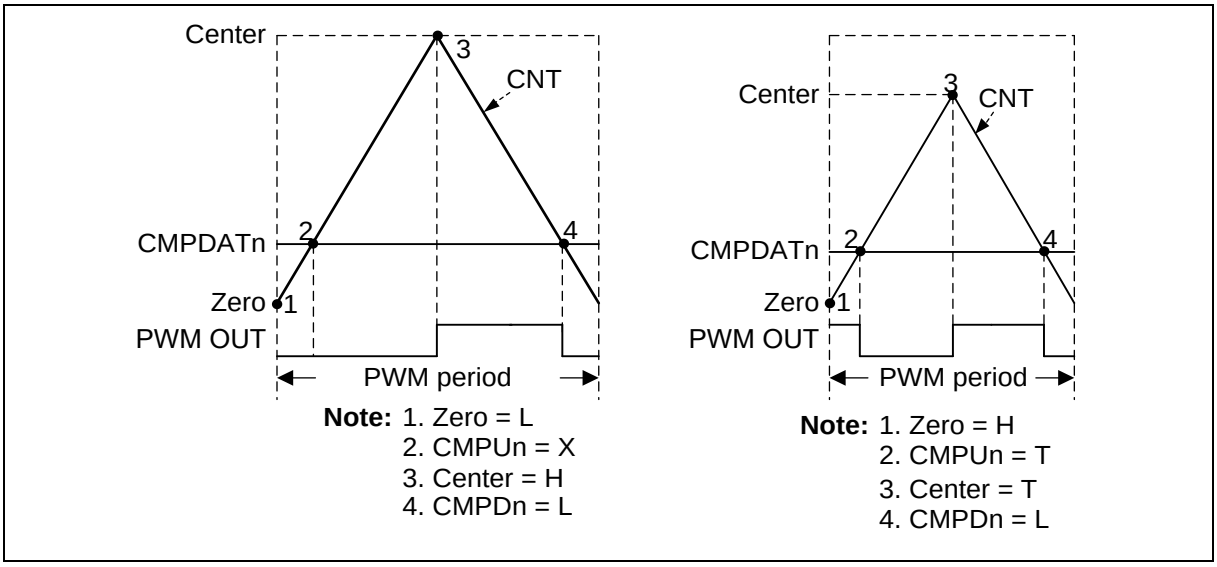


Figure 6.11-14 BPWM Pulse Generation (Left: Asymmetric Pulse, Right: Variety Pulse)

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 6.11-2), down counter type (Table 6.11-3) and up-down counter type (Table 6.11-4). By using event priority, user can easily generate 0% to 100% duty pulse as shown in Figure 6.11-15.

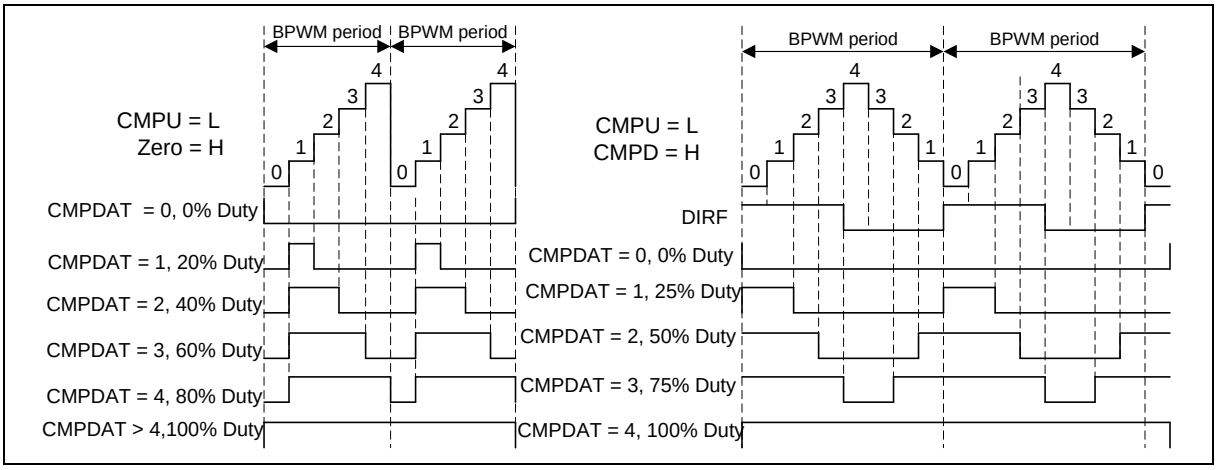


Figure 6.11-15 BPWM 0% to 100% Pulse Generation (Left: Up Counter Type, Right: Up-down Counter Type)

Priority	Up Event
1 (Highest)	Period event (CNT = PERIOD)
2	Compare up event(CNT = CMPUn)
3 (Lowest)	Zero event (CNT = 0)

Table 6.11-2 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 6.11-3 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)

Table 6.11-4 BPWM Pulse Generation Event Priority for Up-Down-Counter

6.11.5.11 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]).

6.11.5.12 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 6.11-16.

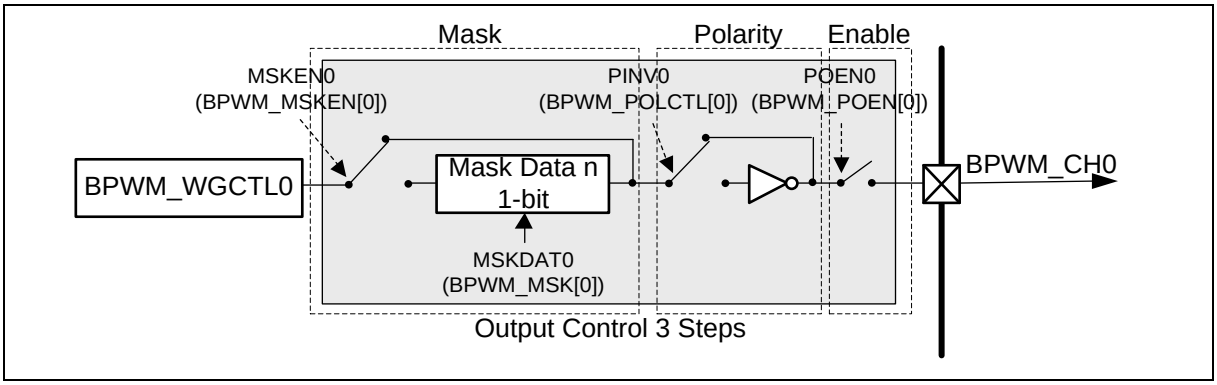


Figure 6.11-16 BPWM_CH0 Output Control 3 Steps

6.11.5.13 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 6.11-17 shows an example of how BPWM mask control can be used for the override feature.

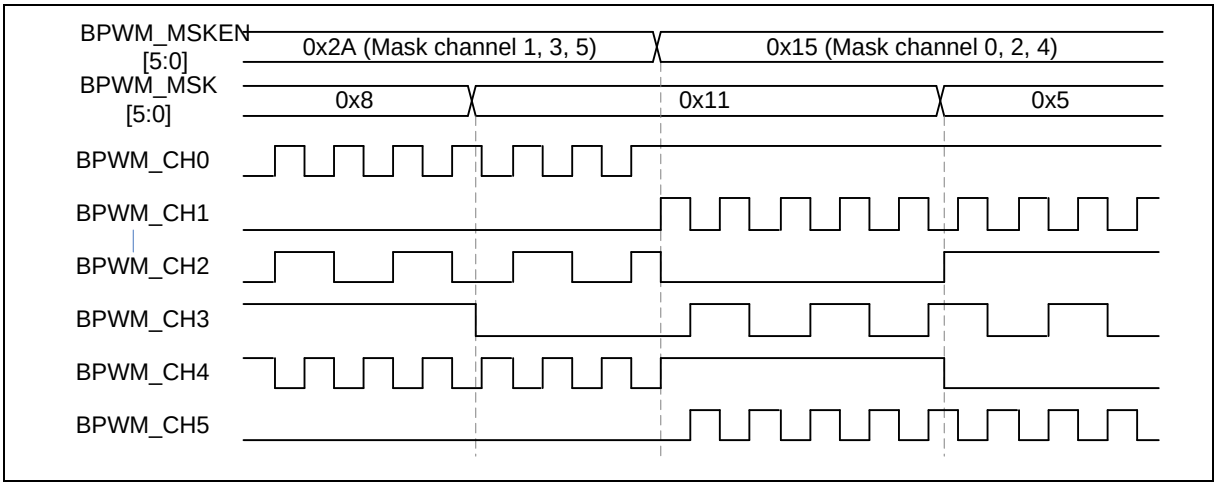
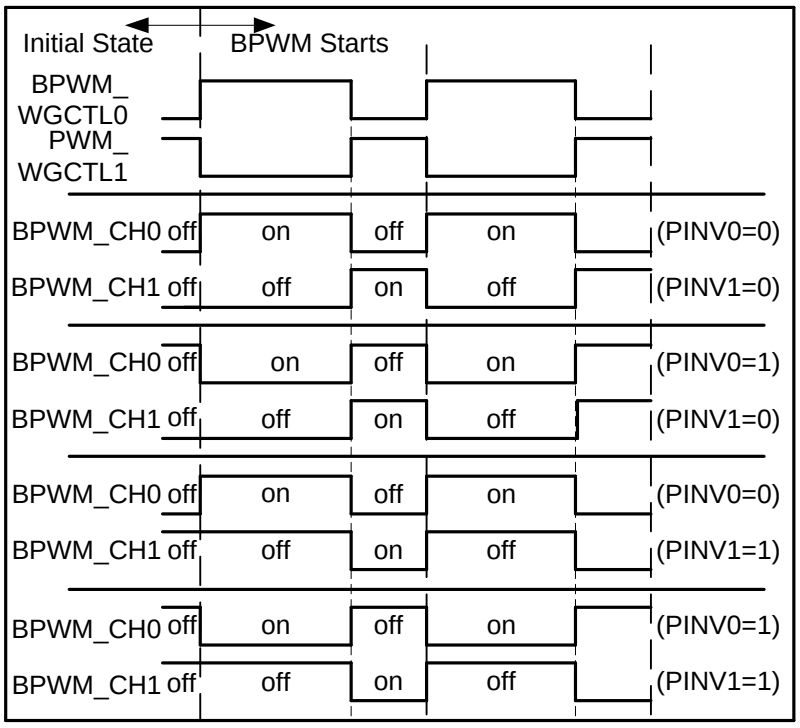


Figure 6.11-17 Mask Control Waveform Illustration

6.11.5.14 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 6.11-18 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 6.11-18 Initial State and Polarity Control

6.11.5.15 BPWM Interrupt Generator

There are two independent interrupts for each BPWM as shown in Figure 6.11-19.

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 6.11-19 demonstrates the architecture of the BPWM interrupts.

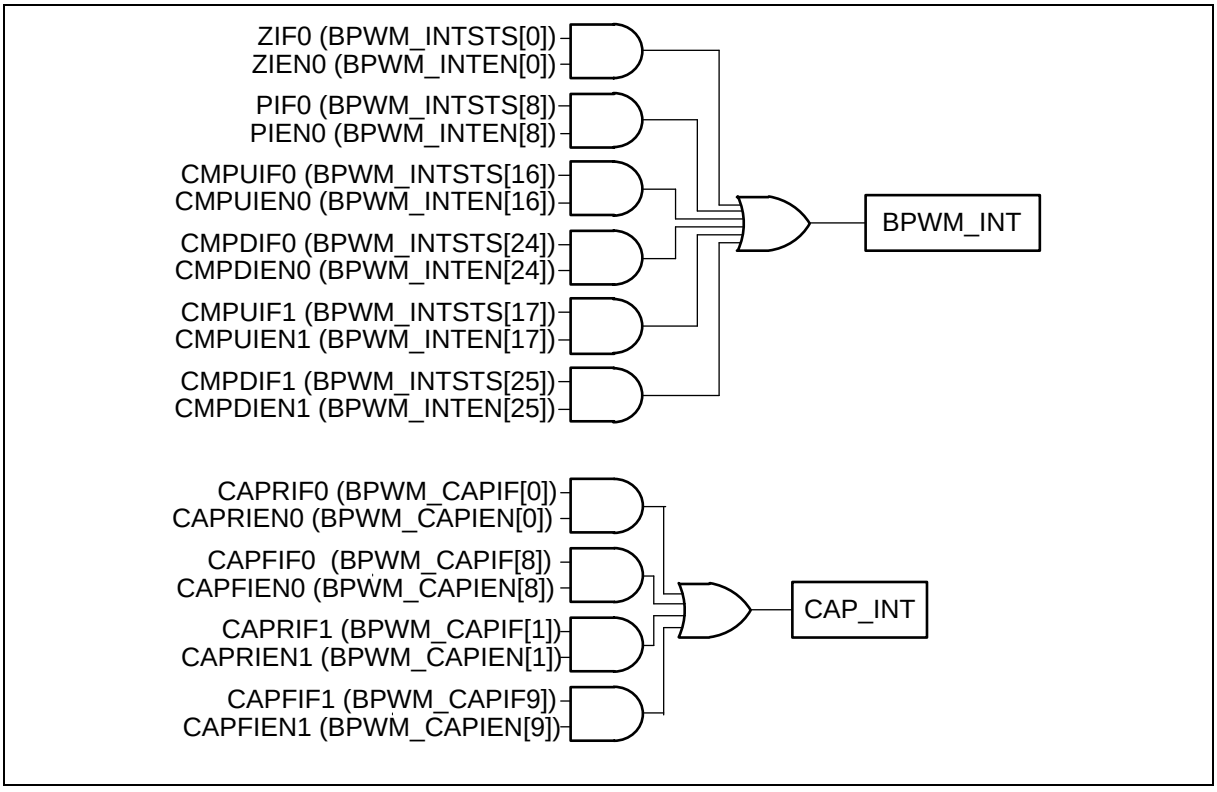


Figure 6.11-19 BPWM_CH0 and BPWM_CH1 Pair Interrupt Architecture Diagram

6.11.5.16 BPWM Trigger ADC Generator

BPWM can be one of the ADC conversion trigger source. Each BPWM pair channels share the same trigger source. Setting TRGSELn is to select the trigger sources, where TRGSELn is TRGSEL0, TRGSEL1, …, and TRGSEL5, which are located in BPWM_ADCTS0[3:0], BPWM_ADCTS0[11:8], BPWM_ADCTS0[19:16], BPWM_ADCTS0[27:24], BPWM_ADCTS1[3:0] and BPWM_ADTS1[11:8], respectively. Setting TRGENn is to enable the trigger output to ADC, where TRGENn is TRGEN0, TRGEN1, …, TRGEN5, which are located in BPWM_ADCTS0[7], BPWM_ADCTS0[15], BPWM_ADCTS0[23], BPWM_ADCTS0[31], BPWM_ADCTS1[7] and BPWM_ADCTS1[15], respectively. The number n (n = 0,1, ..,5) denotes BPWM channel number.

There are 7 BPWM events can be selected as the trigger source for one pair of channels. Figure 6.11-20 is an example of BPWM_CH0 and BPWM_CH1. BPWM can trigger ADC to start conversion in different timings by setting PERIOD and CMPDAT. Figure 6.11-22 is the trigger ADC timing waveform in the up-down counter type.

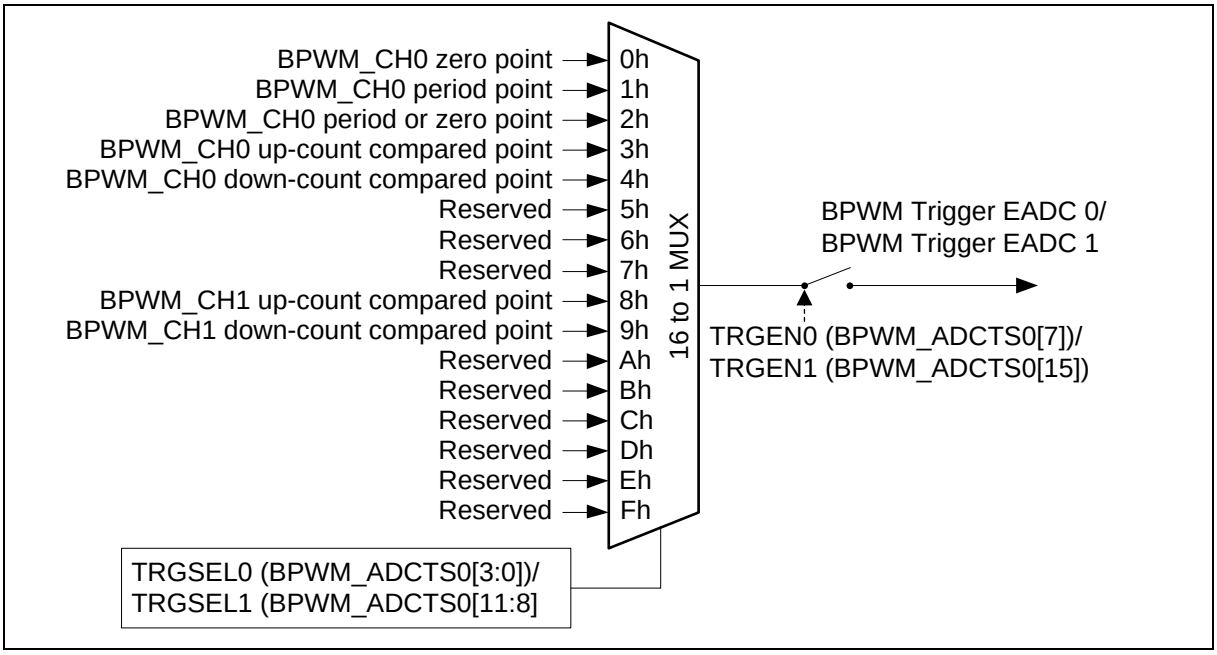


Figure 6.11-20 BPWM_CH0 and BPWM_CH1 Pair Trigger ADC Source Block Diagram

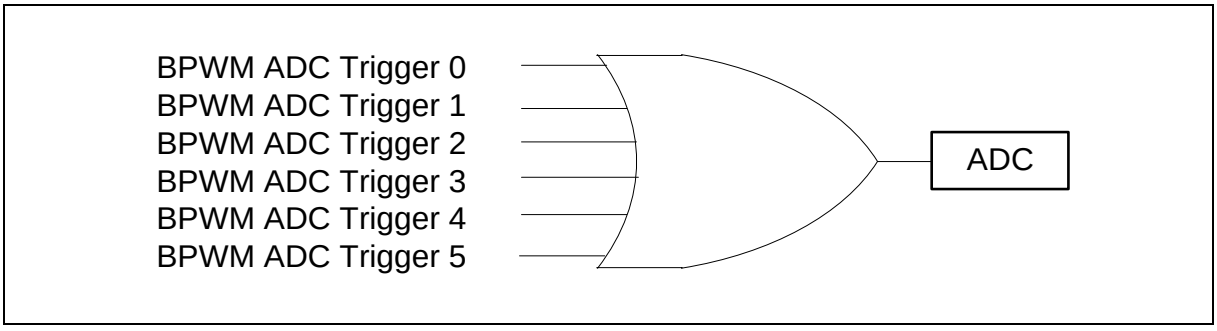


Figure 6.11-21 BPWM CH0~ CH5 Trigger ADC Block Diagram

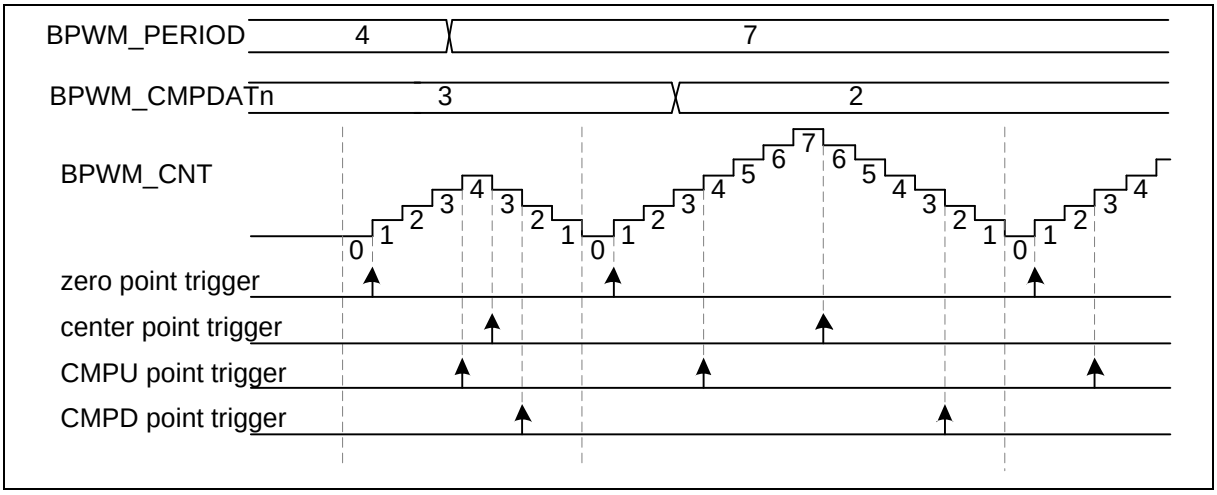


Figure 6.11-22 BPWM Trigger ADC in Up-Down Counter Type Timing Waveform

6.11.5.17 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 6.11-23 is the capture block diagram of channel 0.

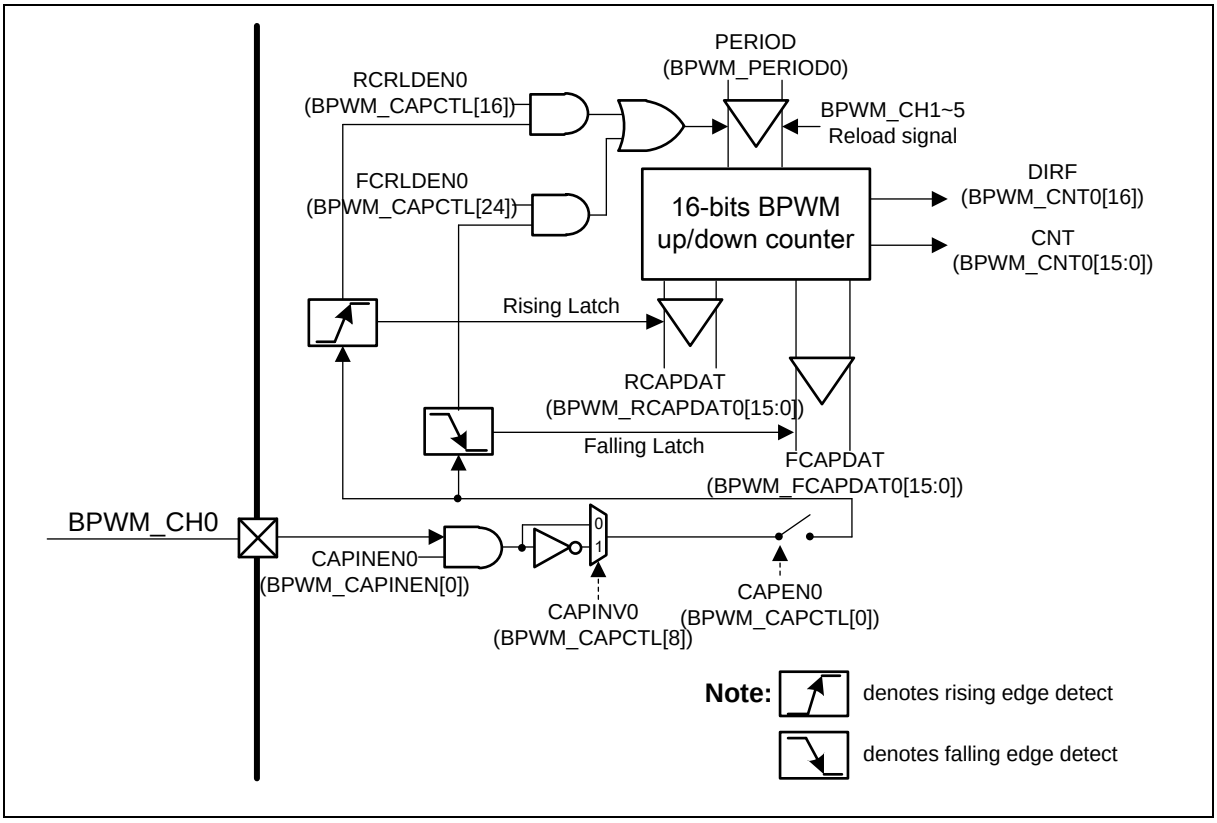


Figure 6.11-23 BPWM_CH0 Capture Block Diagram

Figure 6.11-24 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 6.11-24 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 Over run 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 Over run status CFIFOVn (BPWM_CAPSTS[13:8]).

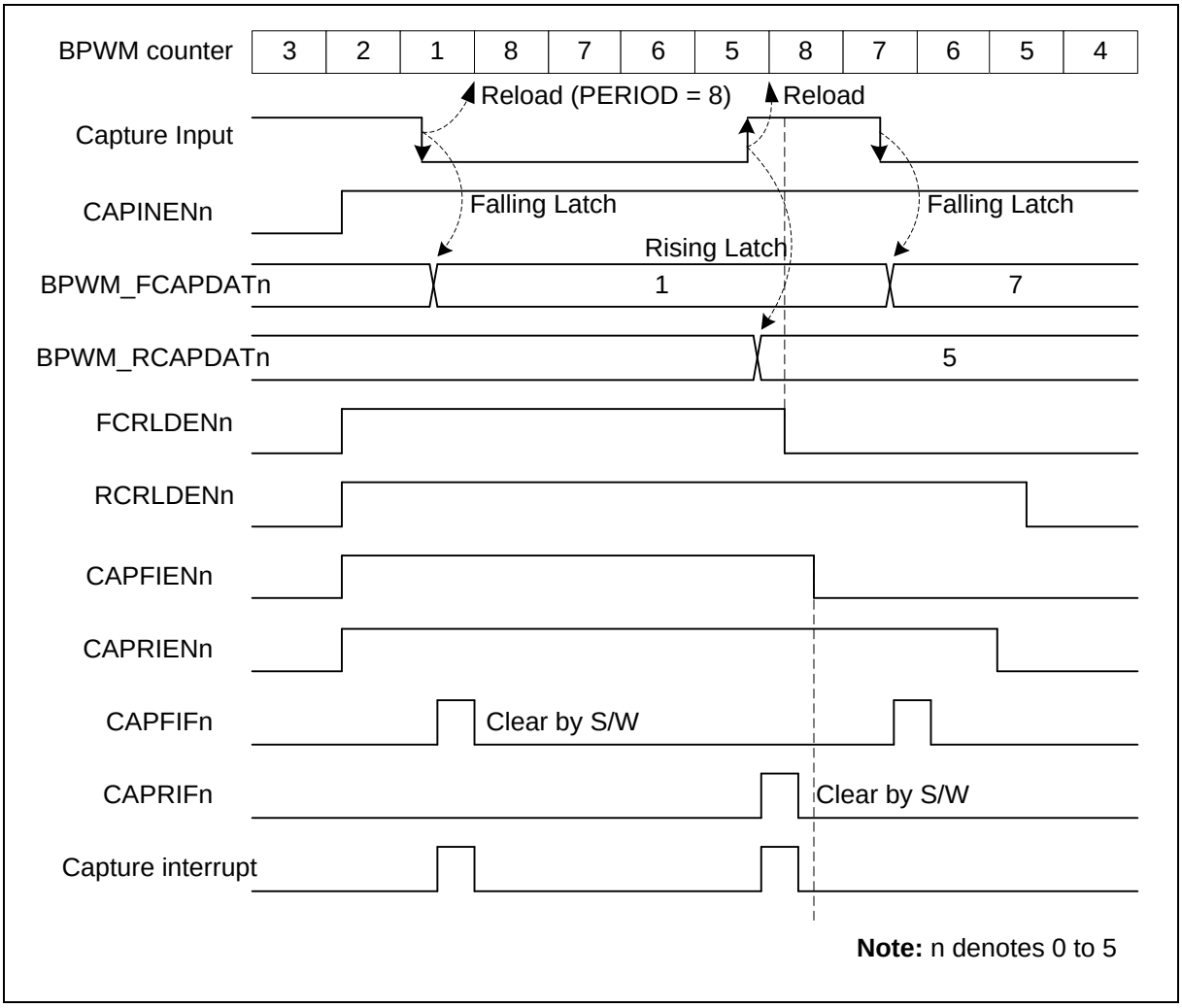


Figure 6.11-24 Capture Operation Waveform

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 (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 6.11-24, 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 6.11-24, high pulse width is 8+1-7 = 2 BPWM counter time.

6.11.6 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_ADCTS0 x=0, 1	BPWMx_BA+0xF8	R/W	BPWM Trigger ADC Source Select Register 0	0x0000_0000
BPWM_ADCTS1 x=0, 1	BPWMx_BA+0xFC	R/W	BPWM Trigger ADC Source Select Register 1	0x0000_0000
BPWM_SSCTL x=0, 1	BPWMx_BA+0x110	R/W	BPWM Synchronous Start Control Register	0x0000_0000
BPWM_SSTRG x=0, 1	BPWMx_BA+0x114	W	BPWM Synchronous Start Trigger Register	0x0000_0000
BPWM_STATUS x=0, 1	BPWMx_BA+0x120	R/W	BPWM Status Register	0x0000_0000
BPWM_CAPINEN x=0, 1	BPWMx_BA+0x200	R/W	BPWM Capture Input Enable Register	0x0000_0000
BPWM_CAPCTL x=0, 1	BPWMx_BA+0x204	R/W	BPWM Capture Control Register	0x0000_0000
BPWM_CAPSTS x=0, 1	BPWMx_BA+0x208	R	BPWM Capture Status Register	0x0000_0000
BPWM_RCAPDAT0 x=0, 1	BPWMx_BA+0x20C	R	BPWM Rising Capture Data Register 0	0x0000_0000
BPWM_FCAPDAT0 x=0, 1	BPWMx_BA+0x210	R	BPWM Falling Capture Data Register 0	0x0000_0000
BPWM_RCAPDAT1 x=0, 1	BPWMx_BA+0x214	R	BPWM Rising Capture Data Register 1	0x0000_0000
BPWM_FCAPDAT1 x=0, 1	BPWMx_BA+0x218	R	BPWM Falling Capture Data Register 1	0x0000_0000
BPWM_RCAPDAT2 x=0, 1	BPWMx_BA+0x21C	R	BPWM Rising Capture Data Register 2	0x0000_0000
BPWM_FCAPDAT2 x=0, 1	BPWMx_BA+0x220	R	BPWM Falling Capture Data Register 2	0x0000_0000
BPWM_RCAPDAT3 x=0, 1	BPWMx_BA+0x224	R	BPWM Rising Capture Data Register 3	0x0000_0000
BPWM_FCAPDAT3 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

6.11.7 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, WINLDENn and 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 CNTR to generate BPWM waveform, interrupt and trigger ADC. 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 CNTR 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. BPWM can control output level when BPWM counter count to zero.

BPWM Generation Register 1 (BPWM_WGCTL1)

Register	Offset	R/W	Description	Reset Value
BPWM_WGCTL1	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. BPWM can control output level when BPWM counter down count 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. BPWM can control output level when BPWM counter up count to CMPDAT.

BPWM Mask Enable Register (BPWM_MSKEN)

Register	Offset	R/W	Description	Reset Value
BPWM_MSKEN	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 control the state of BPWMn output pin, if corresponding mask function is enabled. Each bit n controls the corresponding BPWM channel n. 0 = Output logic low to BPWMn. 1 = Output logic high to BPWMn.

BPWM Pin Polar Inverse Control (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 ADC Source Select Register 0 (BPWM_ADCTS0)

Register	Offset	R/W	Description	Reset Value
BPWM_ADCTS0	BPWMx_BA+0xF8	R/W	BPWM Trigger ADC Source Select Register 0	0x0000_0000

31	30	29	28	27	26	25	24
TRGEN3	Reserved			TRGSEL3			
23	22	21	20	19	18	17	16
TRGEN2	Reserved			TRGSEL2			
15	14	13	12	11	10	9	8
TRGEN1	Reserved			TRGSEL1			
7	6	5	4	3	2	1	0
TRGEN0	Reserved			TRGSEL0			

Bits	Description	
[31]	**TRGEN3**	BPWM_CH3 Trigger ADC Enable Bit
[30:28]	**Reserved**	Reserved.
[27:24]	**TRGSEL3**	**BPWM_CH3 Trigger ADC Source Select** 0000 = BPWM_CH2 zero point. 0001 = BPWM_CH2 period point. 0010 = BPWM_CH2 zero or period point. 0011 = BPWM_CH2 up-count CMPDAT point. 0100 = BPWM_CH2 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = BPWM_CH3 up-count CMPDAT point. 1001 = BPWM_CH3 down-count CMPDAT point. Others reserved.
[23]	**TRGEN2**	BPWM_CH2 Trigger ADC Enable Bit
[22:20]	**Reserved**	Reserved.
[19:16]	**TRGSEL2**	**BPWM_CH2 Trigger ADC Source Select** 0000 = BPWM_CH2 zero point. 0001 = BPWM_CH2 period point. 0010 = BPWM_CH2 zero or period point. 0011 = BPWM_CH2 up-count CMPDAT point. 0100 = BPWM_CH2 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = BPWM_CH3 up-count CMPDAT point.

		1001 = BPWM_CH3 down-count CMPDAT point. Others reserved
[15]	**TRGEN1**	BPWM_CH1 Trigger ADC Enable Bit
[14:12]	**Reserved**	Reserved.
[11:8]	**TRGSEL1**	**BPWM_CH1 Trigger ADC Source Select** 0000 = BPWM_CH0 zero point. 0001 = BPWM_CH0 period point. 0010 = BPWM_CH0 zero or period point. 0011 = BPWM_CH0 up-count CMPDAT point. 0100 = BPWM_CH0 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = BPWM_CH1 up-count CMPDAT point. 1001 = BPWM_CH1 down-count CMPDAT point. Others reserved
[7]	**TRGEN0**	BPWM_CH0 Trigger ADC Enable Bit
[6:4]	**Reserved**	Reserved.
[3:0]	**TRGSEL0**	**BPWM_CH0 Trigger ADC Source Select** 0000 = BPWM_CH0 zero point. 0001 = BPWM_CH0 period point. 0010 = BPWM_CH0 zero or period point. 0011 = BPWM_CH0 up-count CMPDAT point. 0100 = BPWM_CH0 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = BPWM_CH1 up-count CMPDAT point. 1001 = BPWM_CH1 down-count CMPDAT point. Others reserved

BPWM Trigger ADC Source Select Register 1 (BPWM_ADCTS1)

Register	Offset	R/W	Description	Reset Value
BPWM_ADCTS1	BPWMx_BA+0xFC	R/W	BPWM Trigger ADC Source Select Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
TRGEN5	Reserved			TRGSEL5			
7	6	5	4	3	2	1	0
TRGEN4	Reserved			TRGSEL4			

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**TRGEN5**	BPWM_CH5 Trigger ADC Enable Bit
[14:12]	**Reserved**	Reserved.
[11:8]	**TRGSEL5**	**BPWM_CH5 Trigger ADC Source Select** 0000 = BPWM_CH4 zero point. 0001 = BPWM_CH4 period point. 0010 = BPWM_CH4 zero or period point. 0011 = BPWM_CH4 up-count CMPDAT point. 0100 = BPWM_CH4 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = BPWM_CH5 up-count CMPDAT point. 1001 = BPWM_CH5 down-count CMPDAT point. Others reserved
[7]	**TRGEN4**	BPWM_CH4 Trigger ADC Enable Bit
[6:4]	**Reserved**	Reserved.
[3:0]	**TRGSEL4**	**BPWM_CH4 Trigger ADC Source Select** 0000 = BPWM_CH4 zero point. 0001 = BPWM_CH4 period point. 0010 = BPWM_CH4 zero or period point. 0011 = BPWM_CH4 up-count CMPDAT point. 0100 = BPWM_CH4 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved.

		1000 = BPWM_CH5 up-count CMPDAT point. 1001 = BPWM_CH5 down-count CMPDAT point. Others reserved

BPWM Synchronous Start Control Register (BPWM_SSCTL)

Register	Offset	R/W	Description	Reset Value
BPWM_SSCTL	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		ADCTRG5	ADCTRG4	ADCTRG3	ADCTRG2	ADCTRG1	ADCTRG0
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							CNTMAX0

Bits	Description	
[31:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**ADCTRGn**	**ADC Start of Conversion Status** Each bit n controls the corresponding BPWM channel n. 0 = No ADC start of conversion trigger event has occurred. 1 = An ADC start of conversion trigger event has occurred. Software can write 1 to clear this bit.
[15:1]	**Reserved**	Reserved.
[0]	**CNTMAX0**	**Time-base Counter 0 Equal to 0xFFFF Latched Status** 0 = 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		**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 clear 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 clear 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.

6.12 PWM Generator and Capture Timer (PWM)

6.12.1 Overview

The chip provides two PWM generators — PWM0 and PWM1. Each PWM supports 6 channels of PWM output or input capture. There is a 12-bit prescaler to support flexible clock to the 16-bit PWM counter with 16-bit comparator. The PWM counter supports up, down and up-down counter types. PWM uses comparator compared with counter to generate events. These events use to generate PWM pulse, interrupt and trigger signal for ADC to start conversion.

The PWM generator supports two standard PWM output modes: Independent mode and Complementary mode, they have difference architecture. In Complementary mode, there are two comparators to generate various PWM pulse with 12-bit dead-time generator. For PWM output control unit, it supports polarity output, independent pin mask and brake functions.

The PWM generator also supports input capture function to latch PWM counter value to the corresponding register when input channel has a rising transition, falling transition or both transition is happened. Capture function also support PDMA to transfer captured data to memory.

6.12.2 Features

6.12.2.1 PWM Function Features

- Supports maximum clock frequency up to 144 MHz
- Supports up to two PWM modules, each module provides 6 output channels
- Supports independent mode for PWM output/Capture input channel
- Supports complementary mode for 3 complementary paired PWM output channel
  - Dead-time insertion with 12-bit resolution
  - Two compared values during one period
- Supports 12-bit prescaler from 1 to 4096
- Supports 16-bit resolution PWM counter
  - Up, down and up-down counter operation type
- Supports mask function and tri-state enable for each PWM pin
- Supports brake function
  - Brake source from pin and system safety events (clock failed, Brown-out detection, SRAM parity error and CPU lockup)
  - Noise filter for brake source from pin
  - Edge detect brake source to control brake state until brake interrupt cleared
  - Level detect brake source to auto recover function after brake condition removed
- Supports interrupt on the following events:
  - PWM counter matches 0, period value or compared value
  - Brake condition happened
- Supports trigger ADC on the following events:
  - PWM counter matches 0, period value or compared value

6.12.2.2 Capture Function Features

- Supports up to 12 capture input channels with 16-bit resolution

- Supports rising or falling capture condition
- Supports input rising/falling capture interrupt
- Supports rising/falling capture with counter reload option
- Supports PDMA transfer function for PWM all channels

6.12.3 Block Diagram

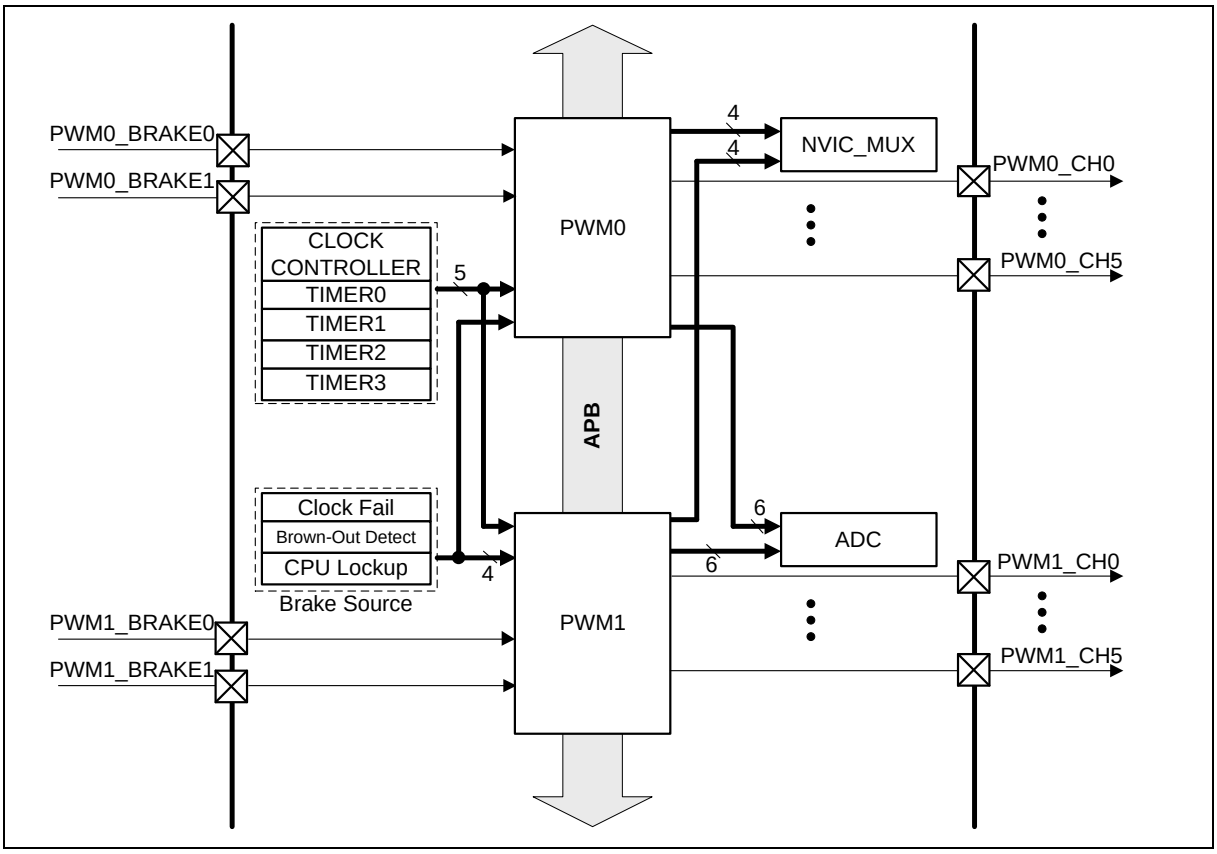


Figure 6.12-1 PWM Generator Overview Block Diagram

PWM 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 PWM generator has three clock source inputs, each clock source can be selected from PWM Clock or four TIMER trigger PWM outputs as Figure 6.12-3 by ECLKSRC0 (PWM_CLKSRC[2:0]) for PWM_CLK0, ECLKSRC2 (PWM_CLKSRC[10:8]) for PWM_CLK2 and ECLKSRC4 (PWM_CLKSRC[18:16]) for PWM_CLK4.

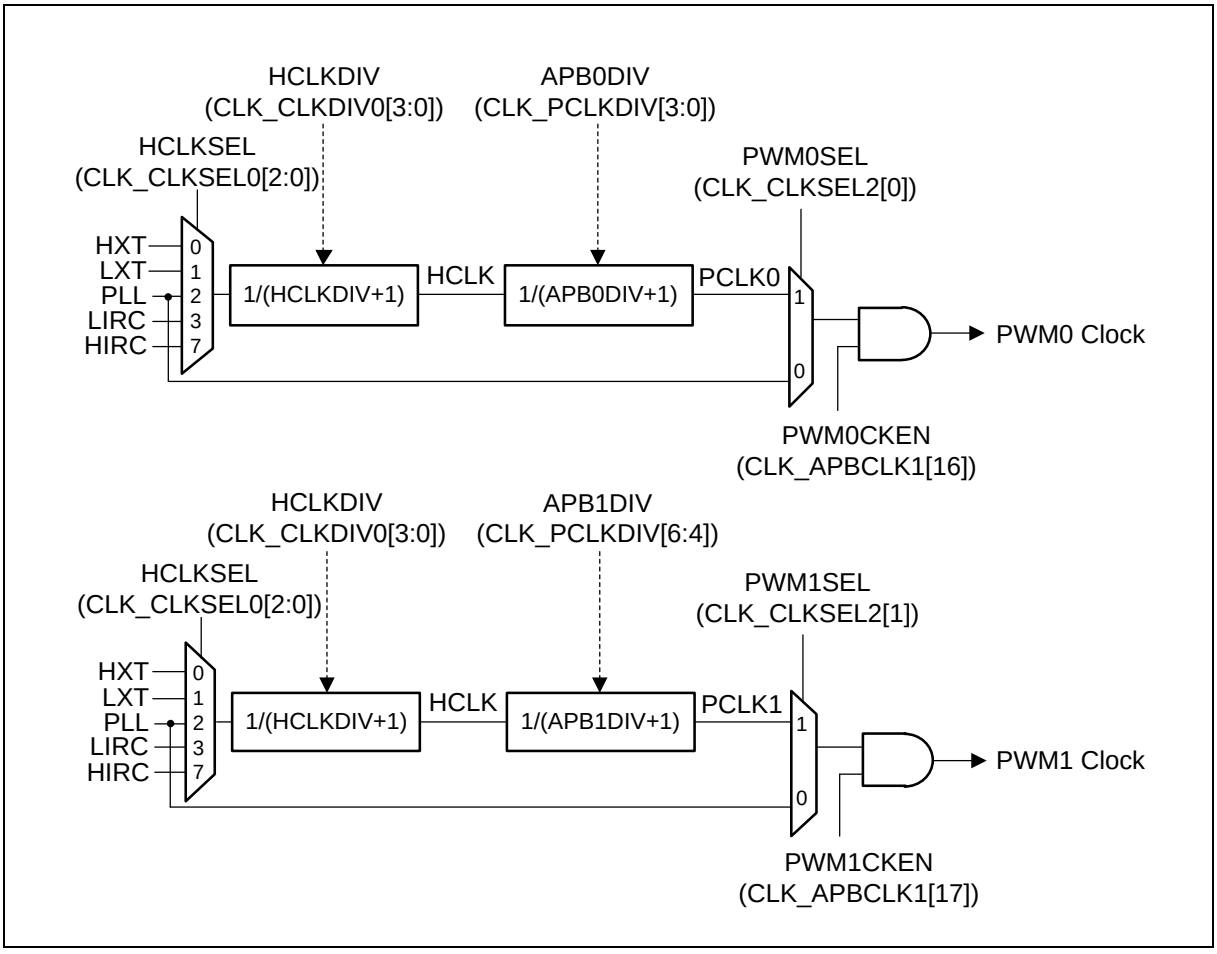


Figure 6.12-2 PWM System Clock Source Control

Frequency Ratio PCLK:PWM Clock	HCLK	PCLK	PWM Clock	HCLKSEL CLK_CLKSEL0[2:0]	HCLKDIV CLK_CLKDIV0[3:0]	APBnDIV (CLK_PCLKDIV [2+4n:4n]), N Denotes 0 Or 1	PWMnSEL (CLK_CLKSEL2[N]), N Denotes 0 Or 1
1:1	HCLK	PCLK	PCLK	Don't care	Don't care	Don't care	1
1:2	PLL	PLL/ 2	PLL	2	0	1	0
1:2	PLL/ 2	PLL/ 2	PLL	2	1	0	0

Table 6.12-1 PWM Clock Source Control Registers Setting Table

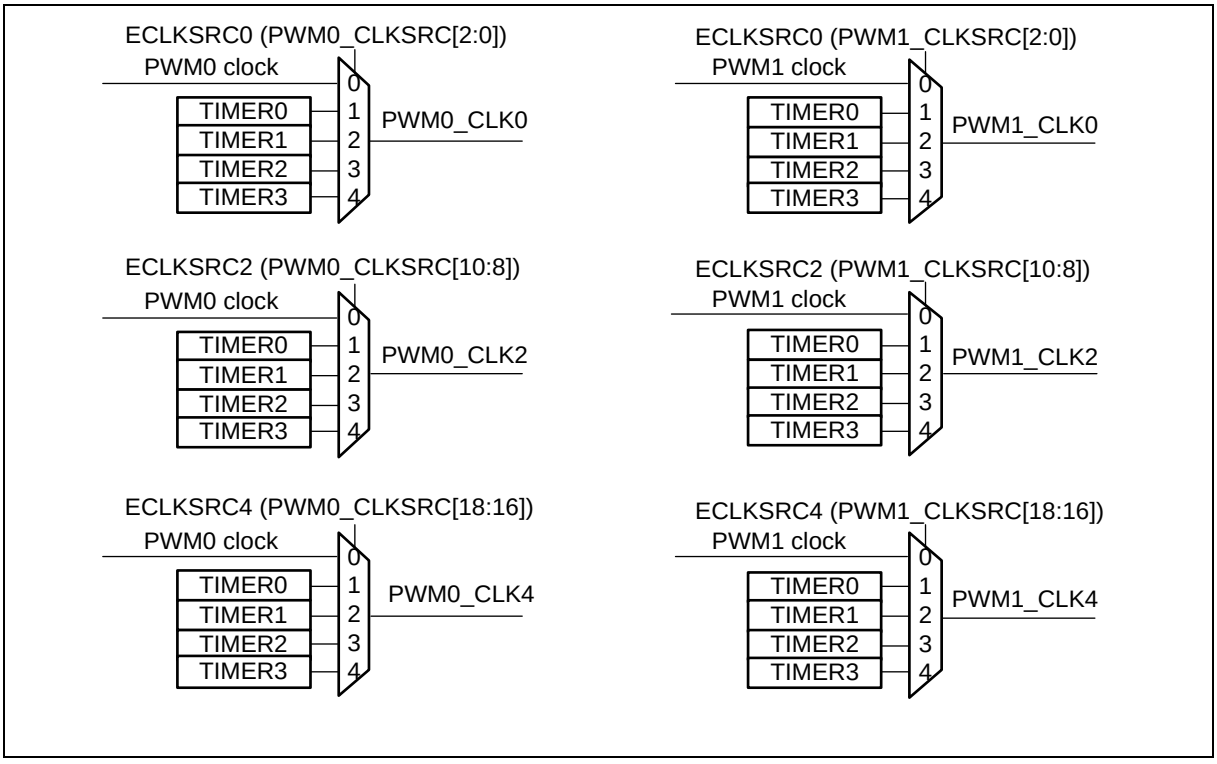


Figure 6.12-3 PWM Clock Source Control

Figure 6.12-4 and Figure 6.12-5 illustrate the architecture of PWM independent mode and complementary mode. No matter independent mode or complementary mode, paired channels' (PWM_CH0 and PWM_CH1, PWM_CH2 and PWM_CH3, PWM_CH4 and PWM_CH5) counters both come from the same clock source and prescaler. When counter count to 0, PERIOD (PWM_PERIODn[15:0]) or equal to comparator, events will be generated. These events are passed to corresponding generators to generate PWM pulse, interrupt signal and trigger signal for ADC to start conversion. Output control is used to changing PWM pulse output state; brake function in output control also generates interrupt events. In complementary mode, even channel use odd channel comparator to generate events.

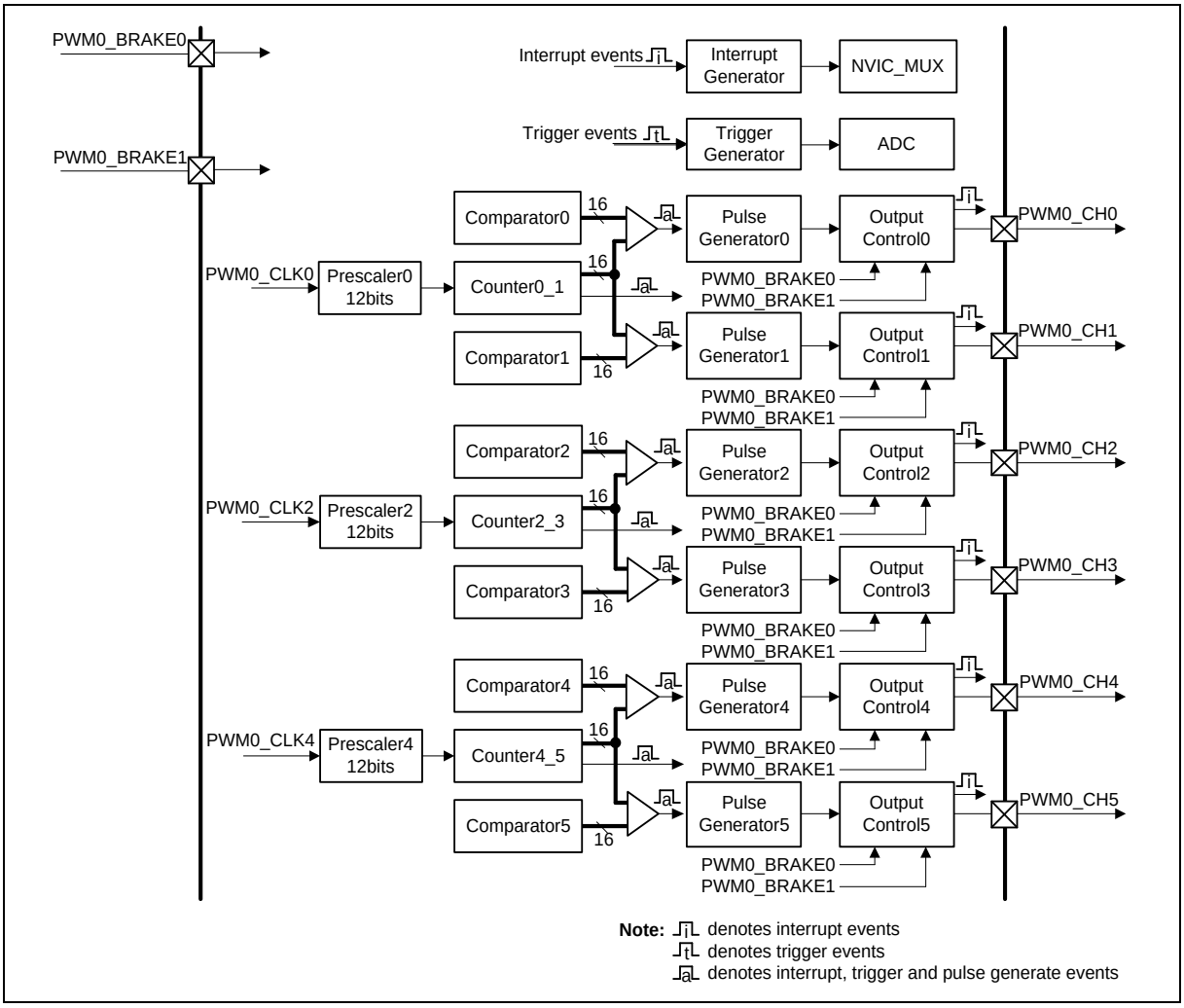


Figure 6.12-4 PWM Independent Mode Architecture Diagram

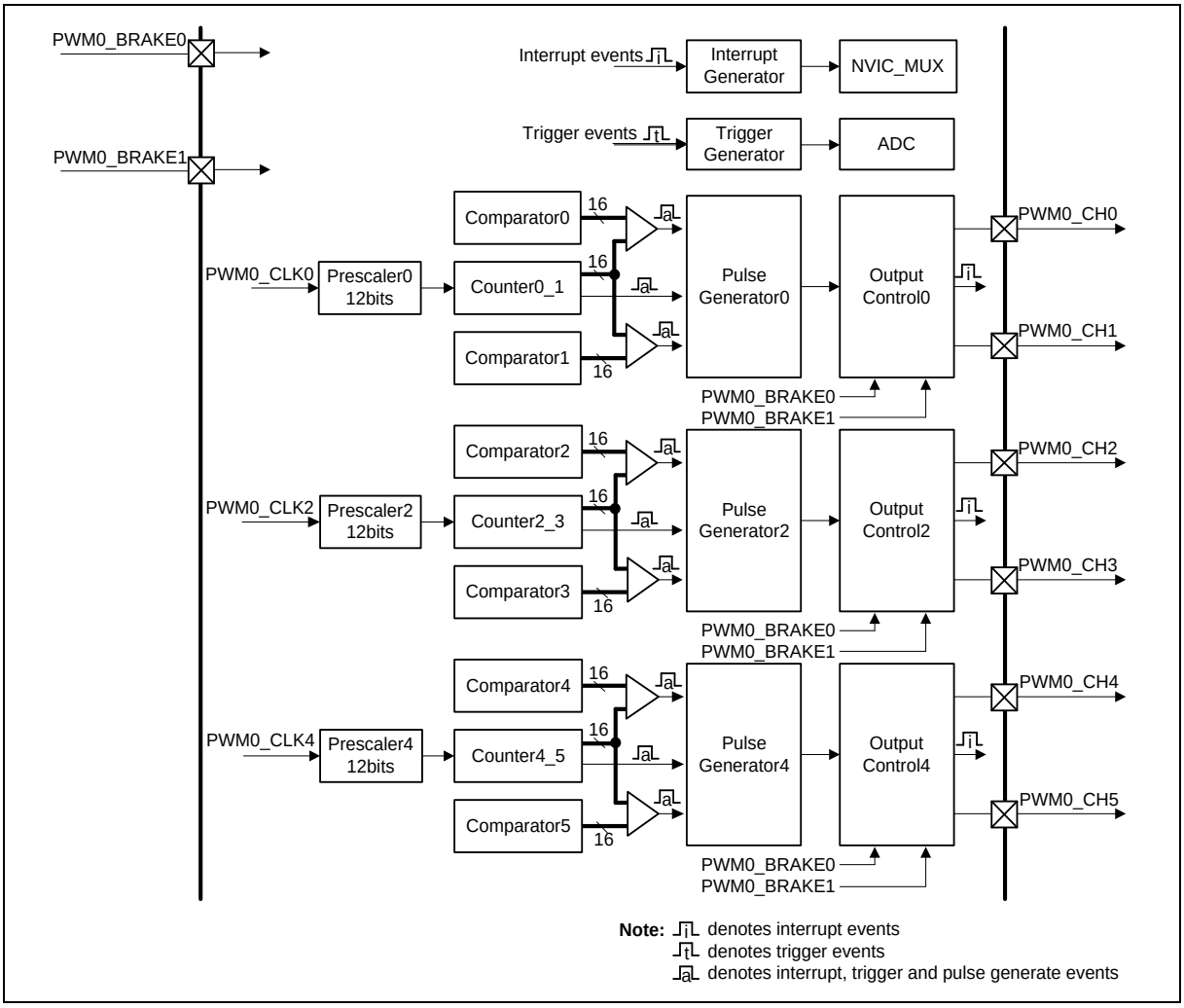


Figure 6.12-5 PWM Complementary Mode Architecture Diagram

6.12.4 Basic Configuration

6.12.4.1 PWM0 Basic Configuration

- Clock Source Configuration
  - Select the source of PWM0 peripheral clock on PWM0SEL (CLK_CLKSEL2[0])
  - Enable PWM0 peripheral clock in PWM0CKEN CLK_APBCLK1[16]).
- Reset Configuration
  - Reset PWM0 in PWM0RST (SYS_IPRST2[16])

6.12.4.2 PWM1 Basic Configuration

- Clock Source Configuration
  - Select the source of PWM1 peripheral clock on PWM1SEL (CLK_CLKSEL2[1])
  - Enable PWM1 peripheral clock in PWM1CKEN (CLK_APBCLK1[17]).
- Reset Configuration
  - Reset PWM1 in PWM1RST (SYS_IPRST2[17])

6.12.5 Functional Description

6.12.5.1 PWM Prescaler

The PWM prescaler is used to divide clock source, prescaler counting CLKPSC +1 times, PWM counter only count once. The prescale double buffer is setting by CLKPSC (PWM_CLKPSCn[11:0], n = 0, 2, 4) bits. Figure 6.12-6 is an example of PWM channel 0 prescale waveform. The prescale counter will reload CLKPSC at the begin of the next prescale counter down-count.

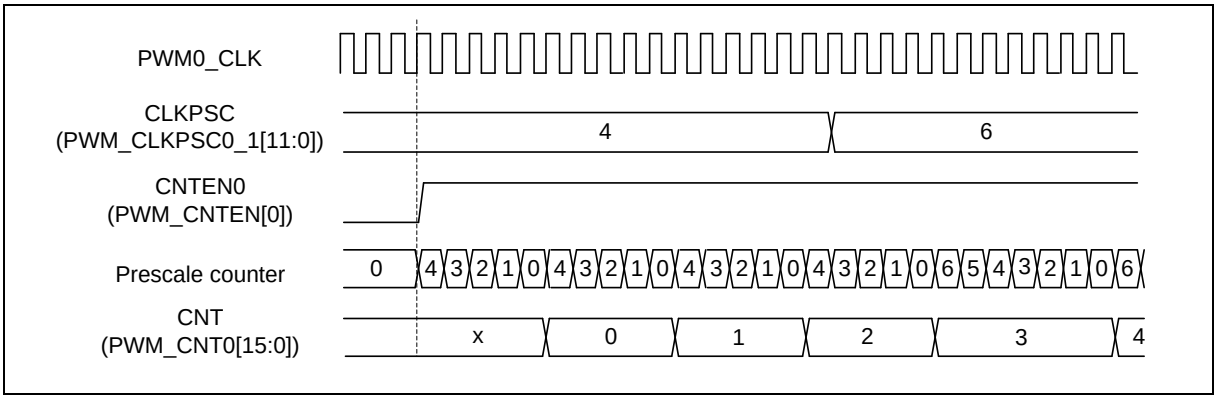


Figure 6.12-6 PWM0_CH0 Prescaler Waveform in Up Counter Type

6.12.5.2 PWM Counter

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

To clear PWM counter, take PWM channel0 for an example, CNT(PWM_CNT0[15:0]) can clear to 0x00 by CNTCLR0 (PWM_CNTCLR[0]). CNT will be cleared when prescale counter count to 0, and CNTCLR0 will be set 0 by hardware automatically.

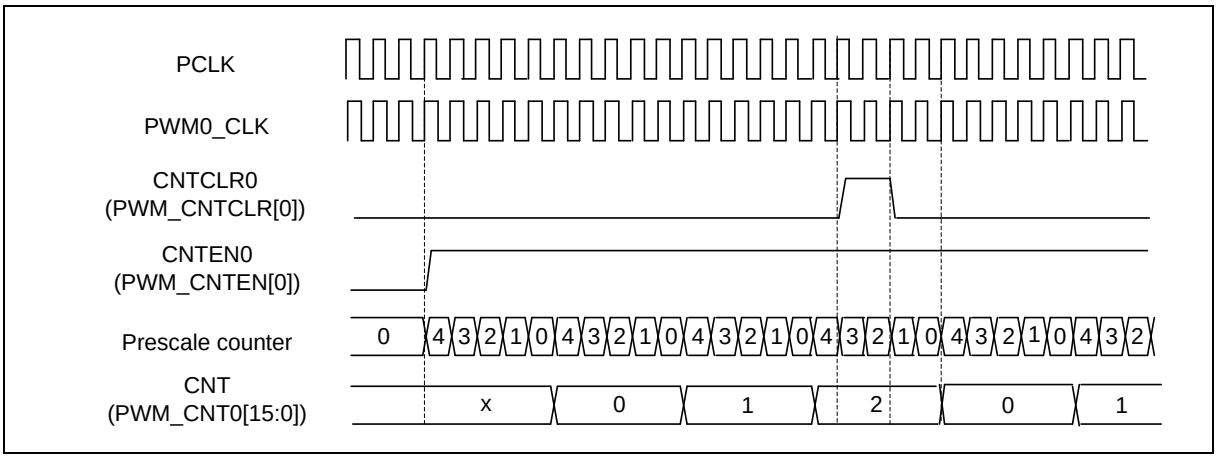


Figure 6.12-7 PWMx Counter Waveform when Setting clear counter

6.12.5.3 Up Counter Type

When PWM counter is set to up counter type, CNTTYPEn (PWM_CTL1[2n+1:2n], n = 0,1..5) is 0x0, it starts up-counting from 0 to PERIOD (PWM_PERIODn[15:0], where n denotes channel number) to complete a PWM period. The current counter value can be read from CNT (PWM_CNTn[15:0]) bits. PWM generates zero point event when the counter counts to 0 and prescale counts to 0. PWM generates period point event when the counter counts to PERIOD and prescale counts to 0. The Figure 6.12-8 shows an example of up counter, wherein

PWM period time = (PERIOD+1) *(CLKPSC+1)* PWMx_CLK.

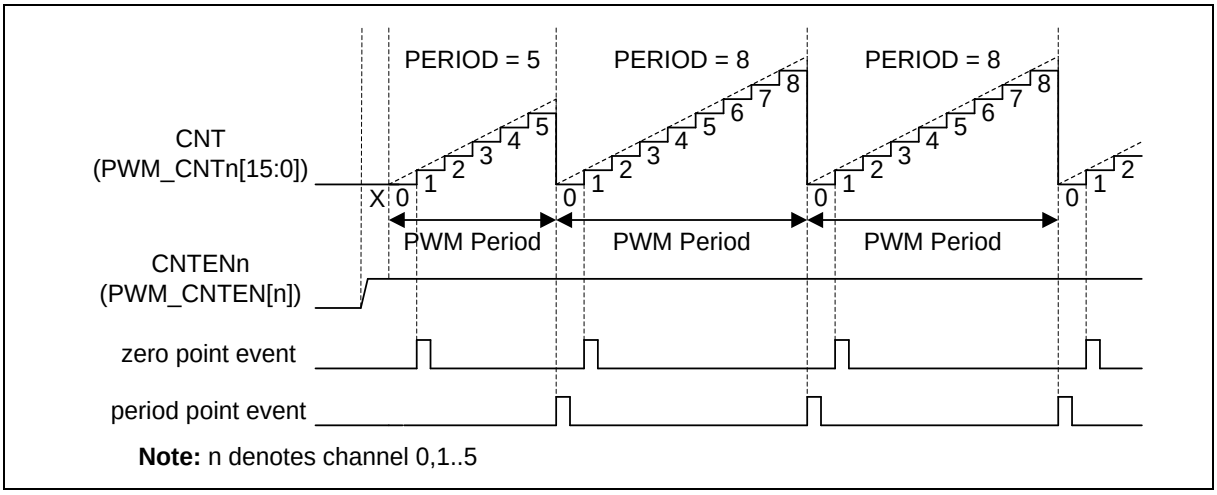


Figure 6.12-8 PWM Up Counter Type

6.12.5.4 Down Counter Type

When PWM counter is set to down counter type, CNTTYPEn (PWM_CTL1[2n+1:2n], n = 0,1..5) is 0x1, it starts down-counting from PERIOD to 0 to complete a PWM period. The current counter value can be read from CNT (PWM_CNTn[15:0]) bits. PWM generates zero point event when the counter counts to 0 and prescale counts to 0. PWM generates period point event when the counter counts to PERIOD and prescale counts to 0. The Figure 6.12-9 shows an example of down counter, wherein

PWM period time = (PERIOD+1) * (CLKPSC+1) * PWMx_CLK.

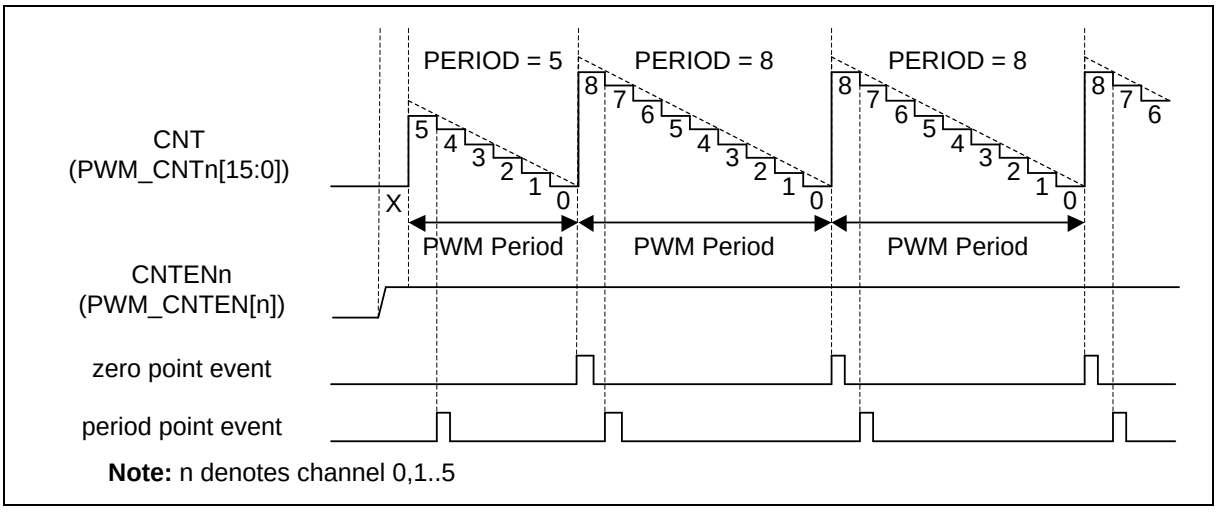


Figure 6.12-9 PWM Down Counter Type

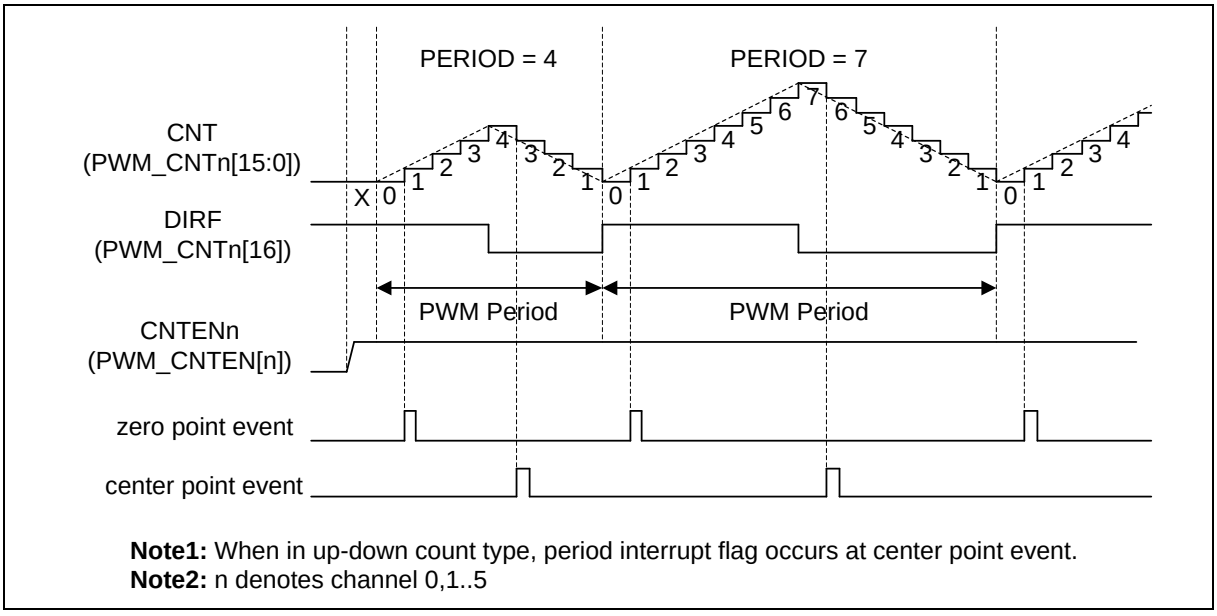
6.12.5.5 Up-Down Counter Type

When PWM counter is set to up-down count type, CNTTYPEn (PWM_CTL1[2n+1:2n], n = 0,1..5) is 0x2, it starts counting-up from 0 to PERIOD and then starts counting down to 0 to complete a PWM period. The current counter value can be read from CNT (PWM_CNTn[15:0]) bits. PWM generates zero point event when the counter counts to 0 and prescale counts to 0. PWM generates center point event 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

PWM period time =(2*PERIOD) * (CLKPSC+1) * PWMx_CLK.

The DIRF (PWM_CNTn[16]) bit is counter direction indicator flag, where high is up counting, and low is down counting.

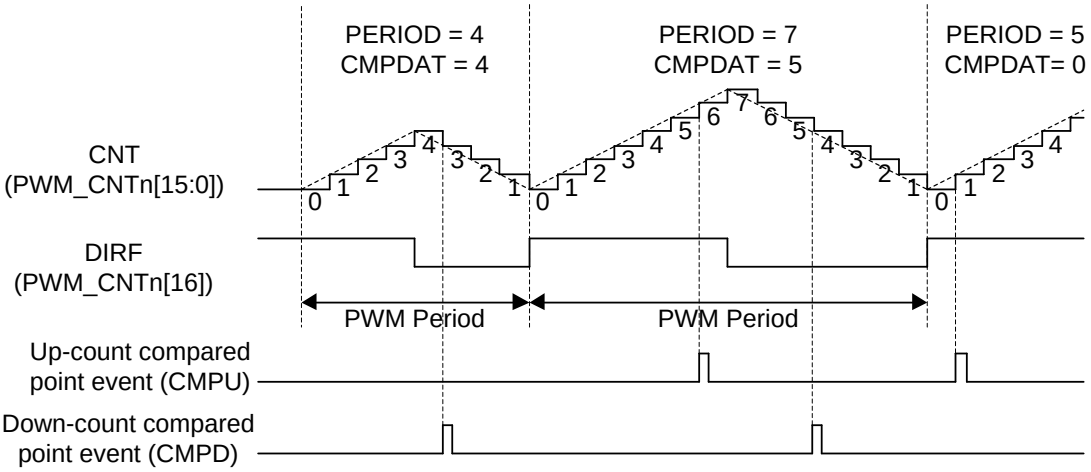


Note1: When in up-down count type, period interrupt flag occurs at center point event.
Note2: n denotes channel 0,1..5

Figure 6.12-10 PWM Up-Down Counter Type

6.12.5.6 PWM Comparator

CMPDATn is a basic comparator register of PWM channel n; In Independent mode each channel only has one comparator, the value of CMPDATn register is continuously compared to the corresponding channel's counter value. In Complementary mode each paired channels has two comparators, and the value of CMPDATn and CMPDATm (n = 0,2,4, m = 1,3,5) registers are continuously compared to the complementary even channel's counter value, because of odd channel's counter is useless. For example, channel 0 and channel 1 are complementary channels, in Complementary mode, channel 1's comparator is continuously compared to channel 0's counter, but not channel 1's. When the counter is equal to value of CMPDAT0 register, PWM generates a compared point event and uses the event to generate PWM pulse, interrupt or use to trigger ADC. In up-down counter type, two events will be generated in a PWM 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 PWM Compared point Events in Up-Down Counter Type

6.12.5.7 PWM Double Buffering

The double buffering uses double buffers to separate software writing and hardware action operation timing. There are three loading modes for loading values to buffer: period loading mode, immediately 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 PWM provides PBUF (PWM_PBUFn[15:0]) as the active PERIOD buffer register, CMPBUF (PWM_CMPBUFn[15:0]) as the active CMPDAT 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 and CMPDAT through software, PWM will load new values to their buffer PBUF (PWM_PBUFn[15:0]) and CMPBUF (PWM_CMPBUFn[15:0]) at start of the next period without affecting the current period counter operation.

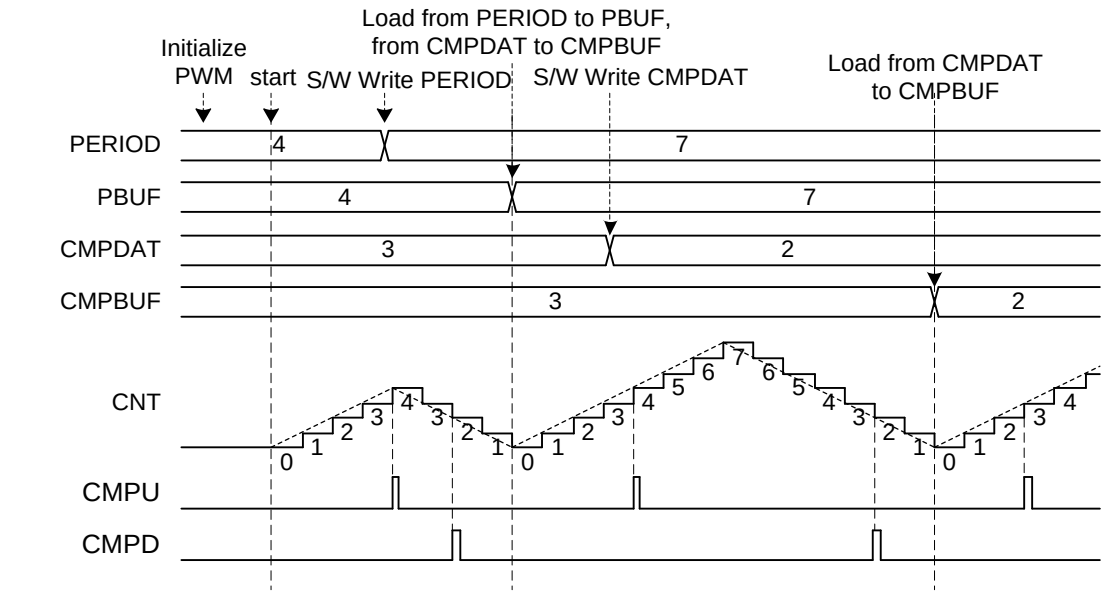


Figure 6.12-12 PWM Double Buffering Illustration

6.12.5.8 Period Loading Mode

When immediately loading mode and center loading mode are disabled that IMMLDENn (PWM_CTL0[21:16]) and CTRLDn (PWM_CTL0[5:0]) are set to 0, PWM operates at period Loading mode. In period Loading mode, PERIOD(PWM_PERIODn[15:0]) and CMP(PWM_CMPDATn[15:0]) will all load to their active PBUF and CMPBUF registers while each period is completed. For example, after PWM counter up counts from zero to PERIOD in the up-counter operation or down counts from PERIOD to zero in the down-counter operation or 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. CMPDAT also follows this rule. The following describes steps sequence of Figure 6.12-13. User can know the PERIOD and CMPDAT update condition, by watching PWM period and CMPU event.

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

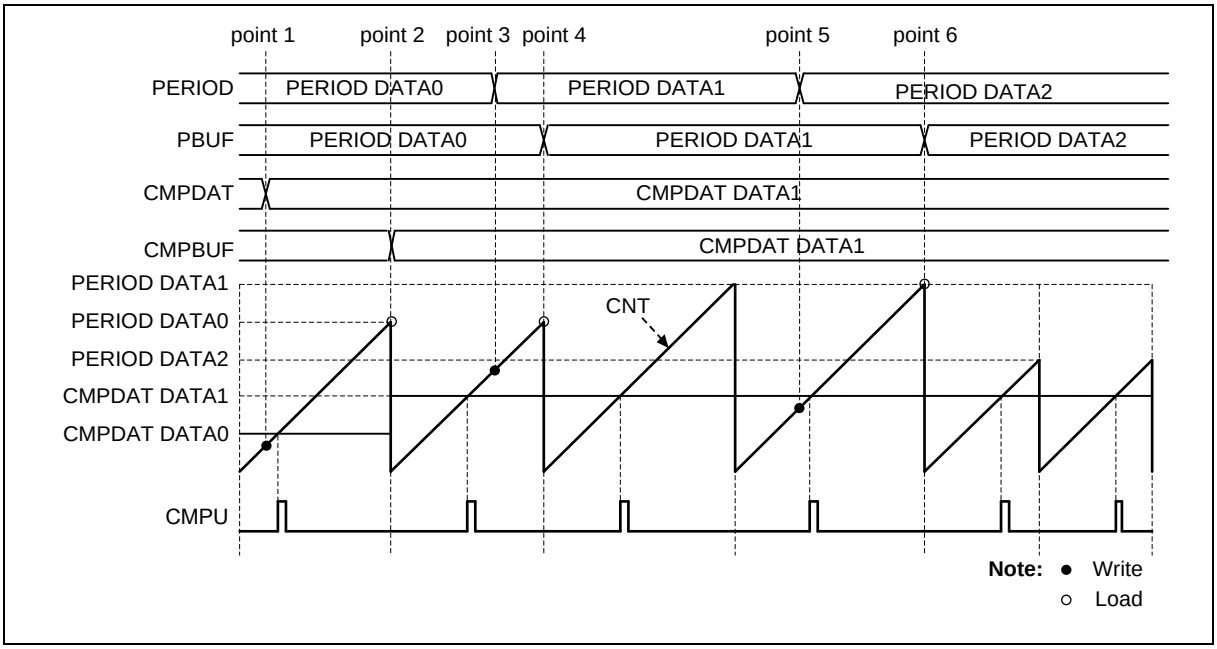


Figure 6.12-13 Period Loading in Up-Count Mode

6.12.5.9 Immediately Loading Mode

If the IMMLDENn (PWM_CTL0[21:16]) bit is set to 1, PWM operates at immediately loading mode. In immediately loading mode, when user update PERIOD (PWM_PERIODn[15:0]) or CMP (PWM_CMPDATn[15:0]), PERIOD or CMPDAT will be load to active PBUF (PWM_PBUFn[15:0]) or CMPBUF (PWM_CMPBUFn[15:0]) after current counter count is completed. If the updated PERIOD value is less than current counter value, counter will count to 0xFFFF. When counter counts to 0xFFFF and prescale also counts to 0, the flag CNTMAXF(PWMx_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 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.

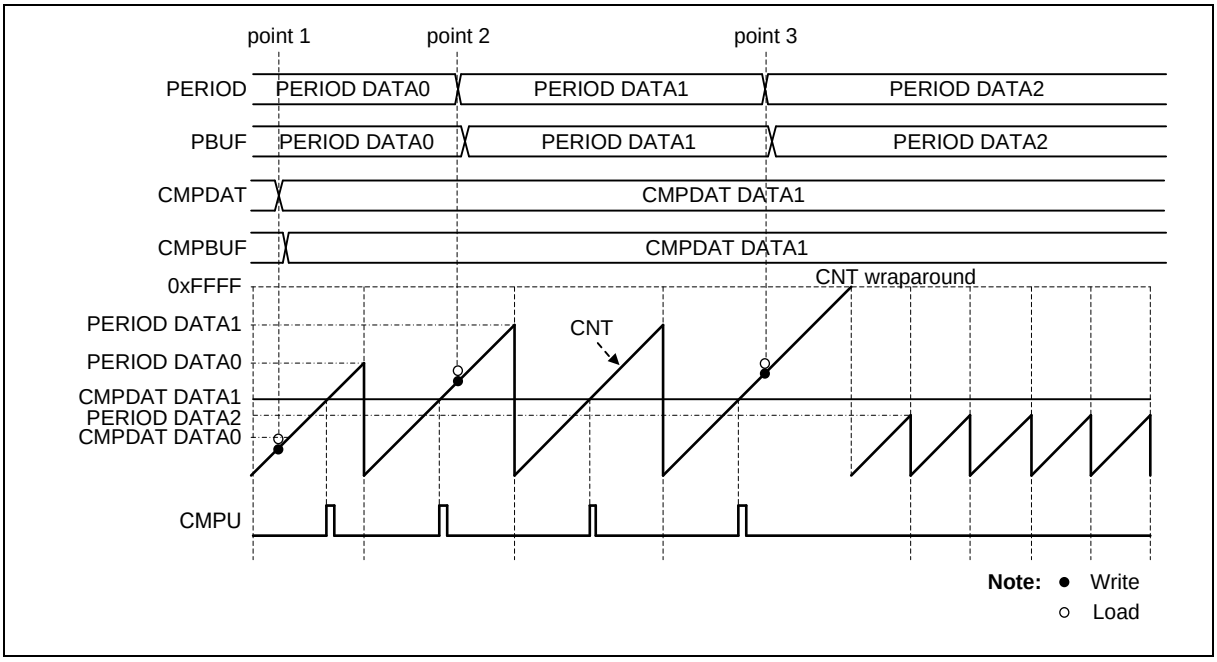


Figure 6.12-14 Immediately Loading in Up-Count Mode

6.12.5.10 Center Loading Mode

When the CTRLDn (PWM_CTL0[5:0]) bit is set to 1 and PWM counter is set to up-down count type, CNTTYPEn (PWM_CTL1[2n+1:2n], n = 0,1..5) is 0x2, PWM operates at center loading mode. In center loading mode, CMP(PWM_CMPDATn[15:0]) will load to active CMPBUF register in center of each period, that is, counter counts to PERIOD. PERIOD(PWM_PERIODn[15:0]) will all load to their active PBUF registers while each period is completed. Figure 6.12-15 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.

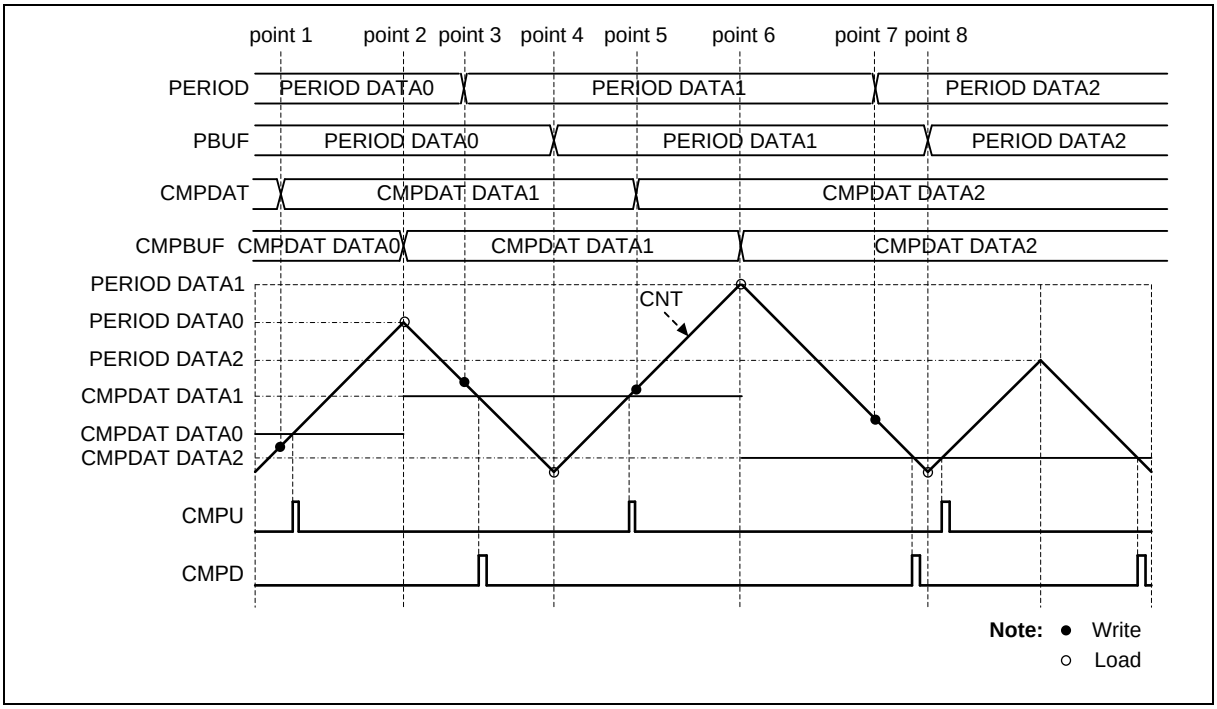


Figure 6.12-15 Center Loading in Up-Down-Count Mode

6.12.5.11 PWM Counter Operation Mode

The PWM counter supports Auto-reload mode.

In Auto-reload mode, CMPDAT and PERIOD registers should be written first and then the CNTENn(PWM_CNTEN[n]) bit is set to 1 to enable PWM prescaler and start to run counter. The value of CLKPSC(PWM_CLKPSCn_m[11:0]), PERIOD(PWM_PERIODn[15:0]) and CMP(PWM_CMPDATn[15:0]) will auto reload to their active buffer according different loading mode. If PERIOD(PWM_PERIODn[15:0]) is set to 0, PWM counter will be set to 0.

6.12.5.12 PWM Pulse Generator

The PWM pulse generator uses counter and comparator events to generate PWM pulse. The events are: zero point, period point in up counter type and down counter type, center point in up-down counter type and counter equal to comparator point in three types. As to up-down counter type, there are two counter equal comparator points, one at up count and the other at down count. Besides, Complementary mode has two comparators compared with counter, and thus comparing equal points will become four in up-down counter type and two for up or down counter type.

Each event point can decide PWM waveform to do nothing (X), set Low (L), set High (H) or toggle (T) by setting the PWM_WGCTL0 and PWM_WGCTL1 registers. Using these points can easily generate asymmetric PWM pulse or variant waveform as shown in Figure 6.12-16. In the figure, PWM is in complementary mode, there are two comparators n and m to generate PWM pulse. n denotes even channel number 0, 2, or 4, and m denotes odd channel number 1, 3, or 5. n channel and m channel are complementary paired. Complementary mode uses two channels (CH0 and CH1, CH2 and CH3, or CH4 and CH5) as a pair of PWM outputs to generate complement paired waveforms. CMPU denotes CNT(PWM_CNTn[15:0]) is equal to CMP(PWM_CMPDATn[15:0]) when counting up. CMPD denotes CNT bits is equal to CMP bits when counting down.

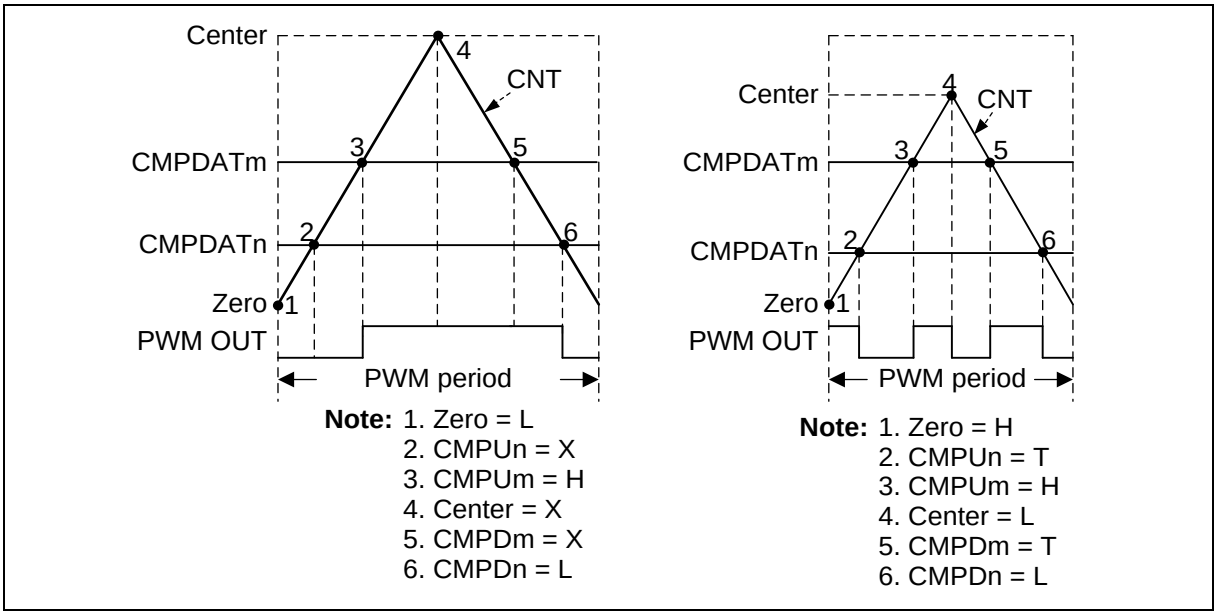


Figure 6.12-16 PWM Pulse Generation

The generation events may sometimes set to the same value, as the reason, events priority between different counter types are list below, up counter type (Table 6.12-2), 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 6.12-17.

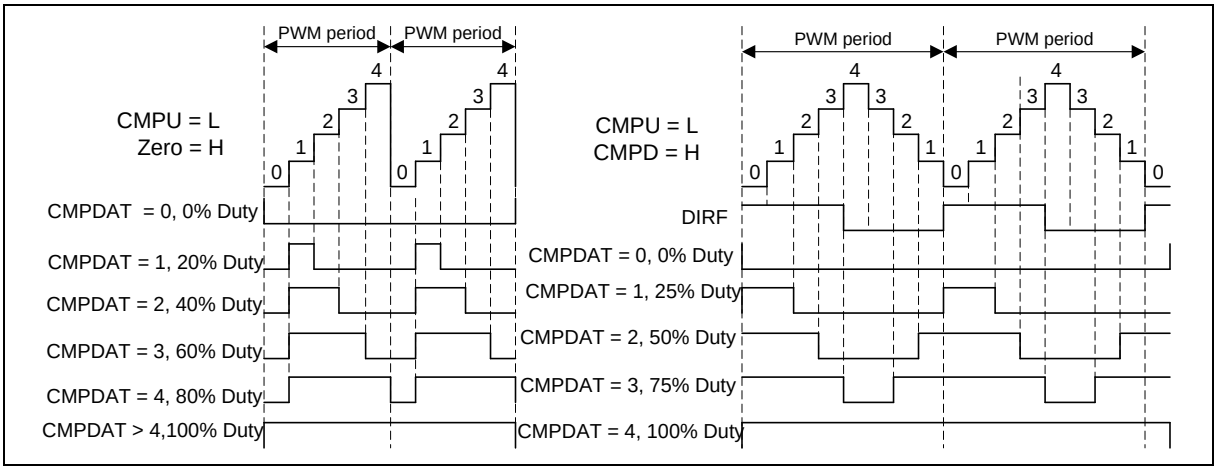


Figure 6.12-17 PWM 0% to 100% Pulse Generation

Priority	Up Event
1 (Highest)	Period event (CNT = PERIOD)
2	Compare up event of odd channel (CNT = CMPUm)
3	Compare up event of even channel (CNT = CMPUn)
4 (Lowest)	Zero event (CNT = 0)

Table 6.12-2 PWM 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 PWM Pulse Generation Event Priority for Down-Counter

Priority	Up Event	Down Event
1 (Highest)	Compare up event of odd channel (CNT = CMPUm	Compare down event of odd channel (CNT = CMPDm)
2	Compare up event of even channel (CNT = CMPUn)	Compare down event of even channel (CNT = CMPDn)
3 (Lowest)	Zero event (CNT = 0)	Period (center) event (CNT = PERIOD)

Table 6.12-4 PWM Pulse Generation Event Priority for Up-Down-Counter

6.12.5.13 PWM Output Mode

The PWM supports two output modes: Independent mode which may be applied to DC motor system, Complementary mode with dead-time insertion which may be used in the application of AC induction motor and permanent magnet synchronous motor.

6.12.5.14 Independent mode

By default, the PWM is operating in independent mode, independent mode is enabled when channel n corresponding PWMMODEn (PWM_CTL1[26:24]) bit is set to 0. In this mode six PWM channels: PWM_CH0, PWM_CH1, PWM_CH2, PWM_CH3, PWM_CH4 and PWM_CH5 are running off its own period and duty as shown in Figure 6.12-18.

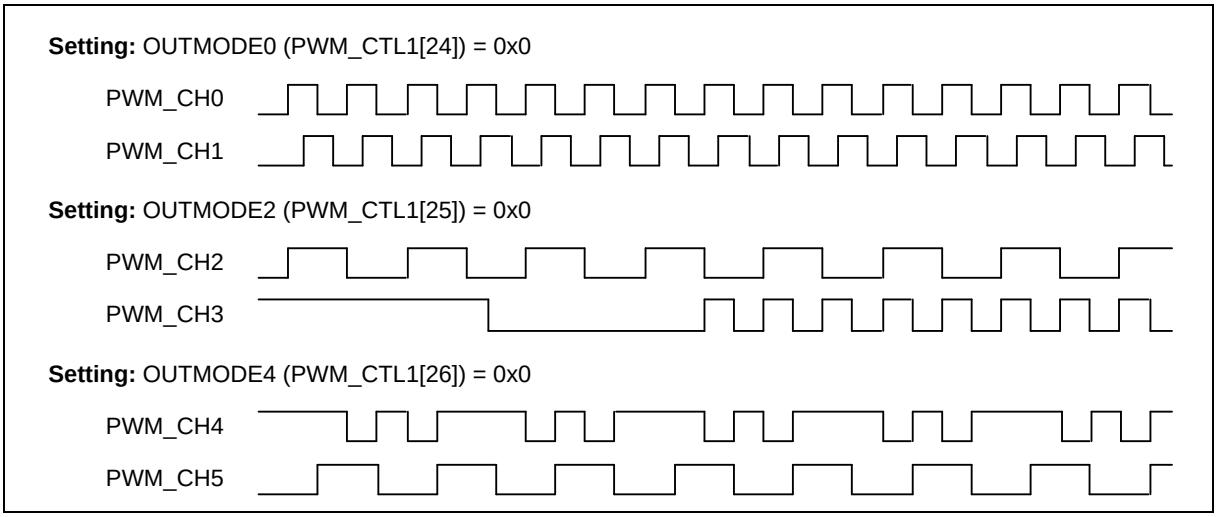


Figure 6.12-18 PWM Independent Mode Waveform

6.12.5.15 Complementary mode

Complementary mode is enabled when the paired channel corresponding PWMMODEn (PWM_CTL1[26:24]) bit set to 1. In this mode there are 3 PWM generators utilized for complementary

mode, with total of 3 PWM output paired pins in this module. In Complimentary modes, the internal odd PWM signal must always be the complement of the corresponding even PWM signal. PWM_CH1 will be the complement of PWM_CH0. PWM_CH3 will be the complement of PWM_CH2 and PWM_CH5 will be the complement of PWM_CH4 as shown in Figure 6.12-19.

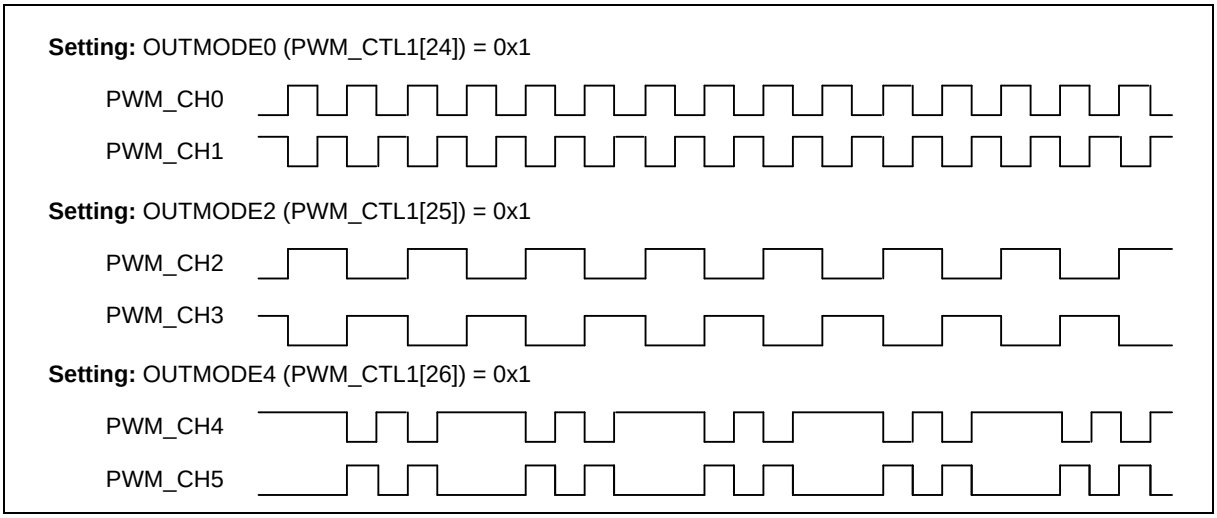


Figure 6.12-19 PWM Complementary Mode Waveform

6.12.5.16 PWM Output Control

After PWM pulse generation, there are four to six steps to control the output of PWM channels. In independent mode, there are Mask, Brake, Pin Polarity and Output Enable four steps as shown in Figure 6.12-20. In complementary mode, it needs two more steps to precede these four steps, Complementary channels and Dead-Time Insertion as shown in Figure 6.12-21.

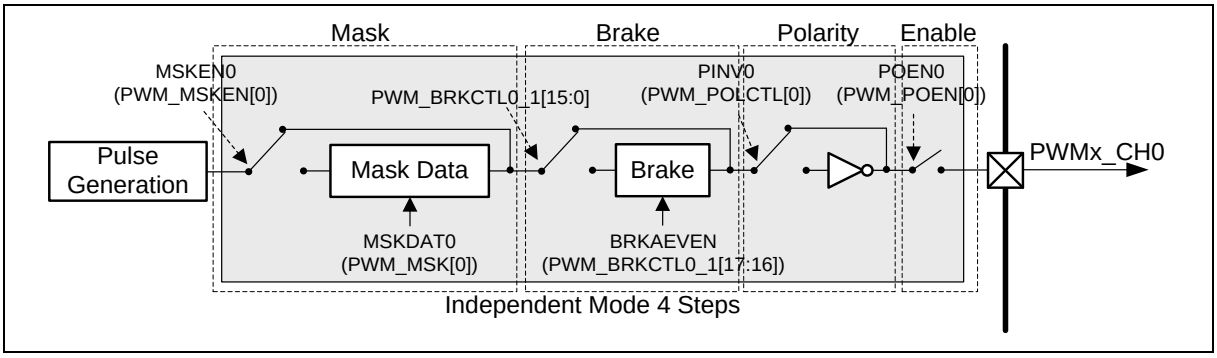


Figure 6.12-20 PWMx_CH0 Output Control in Independent Mode

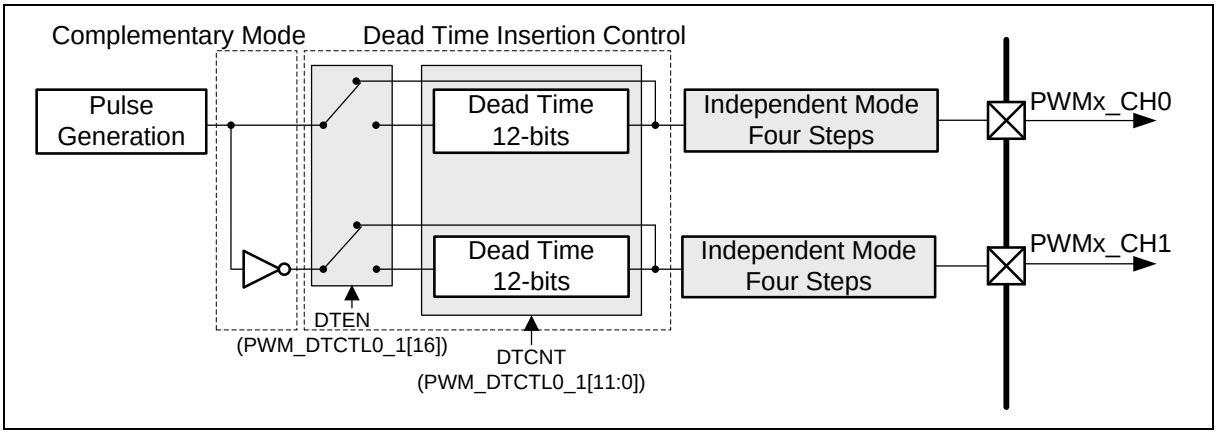


Figure 6.12-21 PWMx_CH0 and PWMx_CH1 Output Control in Complementary Mode

6.12.5.17 Dead-Time Insertion

In the complementary application, the complement channels may drive the external devices like power switches. The dead-time generator inserts a low level period called “dead-time” between complementary outputs to drive these devices safely and to prevent system or devices from the burn-out damage. Hence the dead-time control is a crucial mechanism to the proper operation of the complementary system. By setting corresponding channel n DTEN (PWM_DTCTLn_m[16]) bit to enable dead-time function and DTCNT (PWM_DTCTLn_m[11:0]) to control dead-time period, the dead-time can be calculated from the following formula:

Dead-time = (DTCNT (PWM_DTCTLn[11:0])+1) * PWMx_CLK period

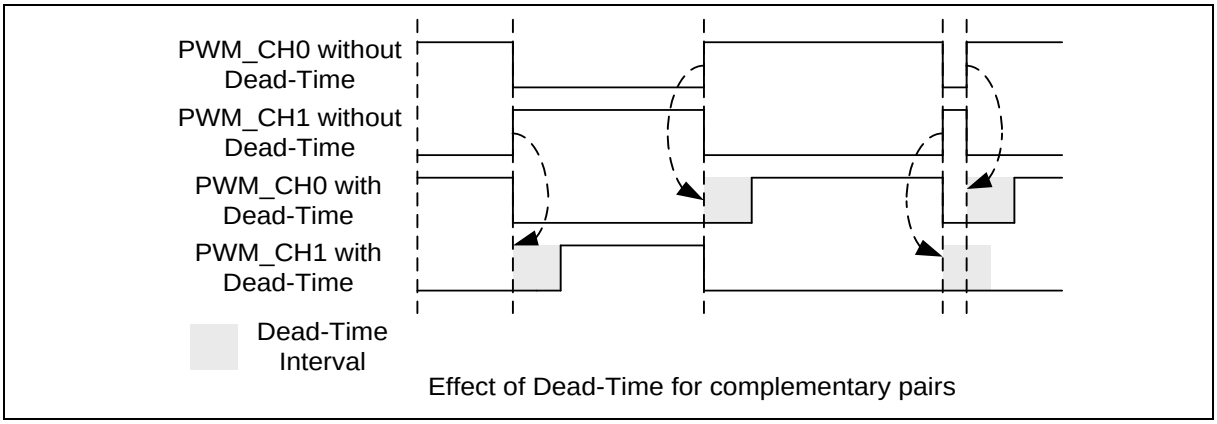
Dead-time insertion clock source can be selected from prescaler output by setting DTCKSEL (PWM_DTCTLn_m[24]) to 1. By default, clock source comes from PWM_CLK, which is prescaler input. Then the dead-time can be calculated from the following formula:

Dead-time = (DTCNT (PWM_DTCTLn[11:0])+1) *

(CLKPSC (PWM_CLKPSCn [11:0])+1)*PWMx_CLK period

Please note that the PWM_DTCTLn_m are write-protected registers.

Figure 6.12-22 indicates the dead-time insertion for one pair of PWM signals.



Effect of Dead-Time for complementary pairs

Figure 6.12-22 Dead-Time Insertion

6.12.5.18 PWM Mask Output Function

Each of the PWM channel output value can be manually overridden with the settings in the PWM Mask Enable Control Register (PWM_MSKEN) and the PWM Masked Data Register (PWM_MSK). With these settings, the PWM channel outputs can be assigned to specified logic states independent of the duty cycle comparison units. The PWM mask bits are useful when controlling various types of Electrically Commutated Motor (ECM) like a BLDC motor. The PWM_MSKEN register contains six bits, MSKENn(PWM_MSKEN[5:0]). If the MASKENn is set to active-high, the PWM channel n output will be overridden. The PWM_MSK register contains six bits, MSKDATn(PWM_MSK[5:0]). The bit value of the MSKDATn determines the state value of the PWM channel n output when the channel is overridden. Figure 6.12-23 shows an example of how PWM mask control can be used for the override feature.

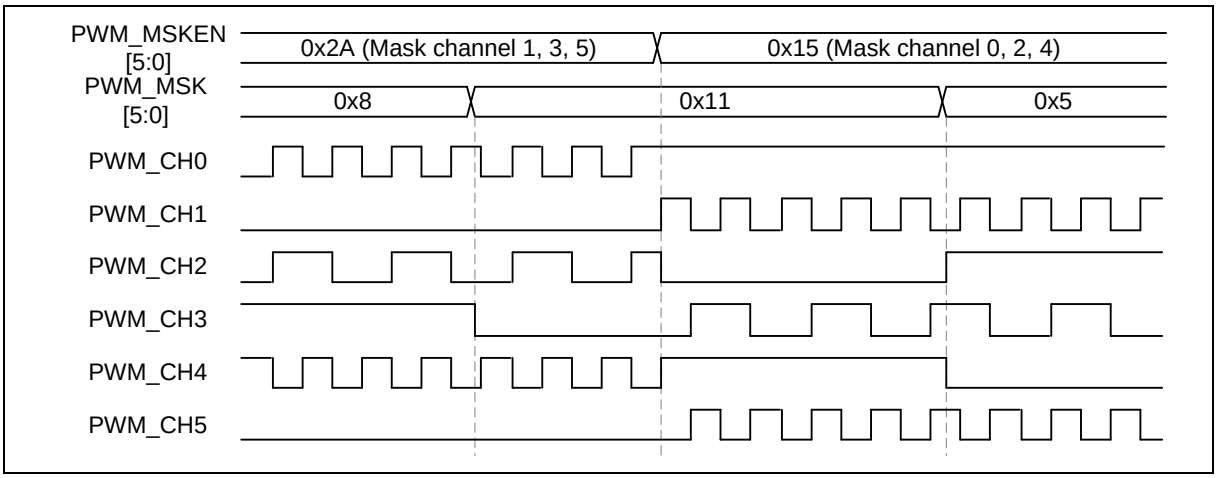


Figure 6.12-23 Illustration of Mask Control Waveform

6.12.5.19 PWM Brake

Each PWM module has two external input brake control signals. User can select active brake pin source is from PWMx_BRAKEy pin by BKxSRC bits of 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 BNF reigster, and noise filter sampling clock can be selected by setting BRKxNFSEL bits of 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 BNF register to realize the polarity setup for the brake control signals. Set BRKxPINV bit to 0, brake event will occurred when PWMx_BRAKEy(x=0,1, y=0,1) pin status is from low to high; set BRKxPINV to 1, brake event will occurred when PWMx_BRAKEy pin status is from high to low.

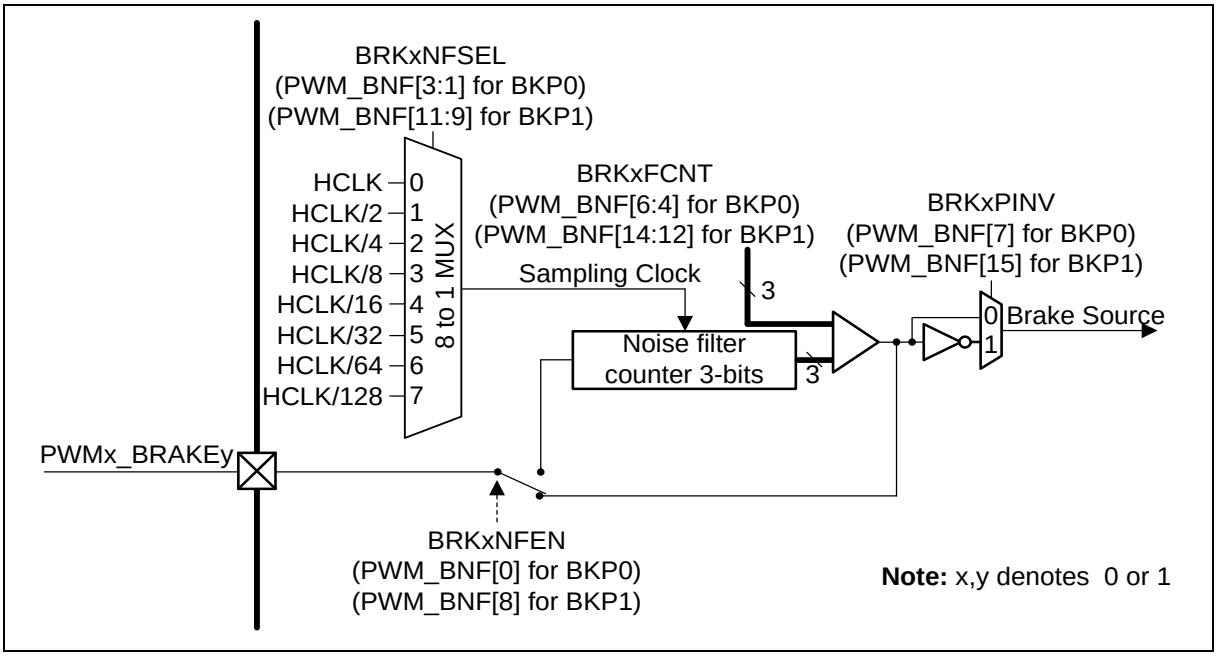


Figure 6.12-24 Brake Noise Filter Block Diagram

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 PWM brake function, as shown Figure 6.12-25. To control paired channels to output safety state, user can setup BRKAEVEN (PWM_BRKCTL0_1[17:16]) for even channels and BRKAODD (PWM_BRKCTL0_1[19:18]) for odd channels when the fault brake event happens. There are two brake detectors: Edge detector and Level detector. When the edge detector detects the brake signal and BRKEIENn_m (PWM_INTEN1[2:0]) is enabled, the brake function generates BRK_INT. This interrupt needs software to clear, and the BRKESTSn (PWM_INTSTS1[21:16]) brake state will keep until the next PWM period starts after the interrupt cleared. The brake function can also operate in another way through the level detector. Once the level detector detects the brake signal and the BRKLIENn_m (PWM_INTEN1[10:8]) is also enabled, the brake function will generate BRK_INT, but BRKLSTSn (PWM_INTSTS1[29:24]) brake state will auto recovery to normal output while level brake source recovery to high level and pass through “Low Level Detection” at the PWM waveform period when brake condition removed without clear interrupt.

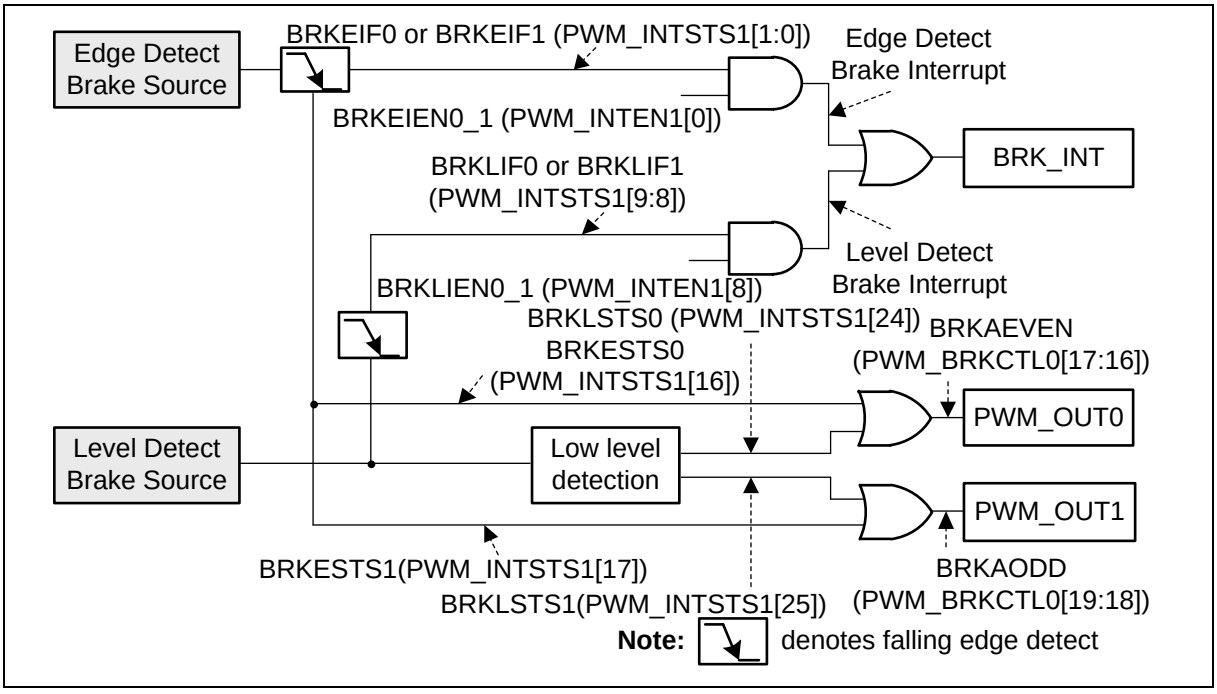


Figure 6.12-25 Brake Block Diagram for PWMx_CH0 and PWMx_CH1 Pair

Figure 6.12-26 illustrates the edge detector waveform for PWMx_CH0 and PWMx_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 PWMx_CH0 and PWMx_CH1. For the first occurring event, software writes 1 to clear the BRKEIF0 flag. After that, the BRKESTS0 bit is cleared by hardware at the next start of the PWM period. At the same moment, the PWMx_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, PWMx_CH1 outputs normally at the next start of the PWM period.

As a contrast to the edge detector example, Figure 6.12-27 illustrates the level detector waveform for PWMx_CH0 and PWMx_CH1 pair. In this case, the BRKLIF0 and BRKLIF1 flags can only indicate the brake event having occurred. The BRKLSTS0 and BRKLSTS1 brake states will automatically recover at the start of the next PWM period no matter at what states the BRKLIF0 and BRKLIF1 flags are at that moment.

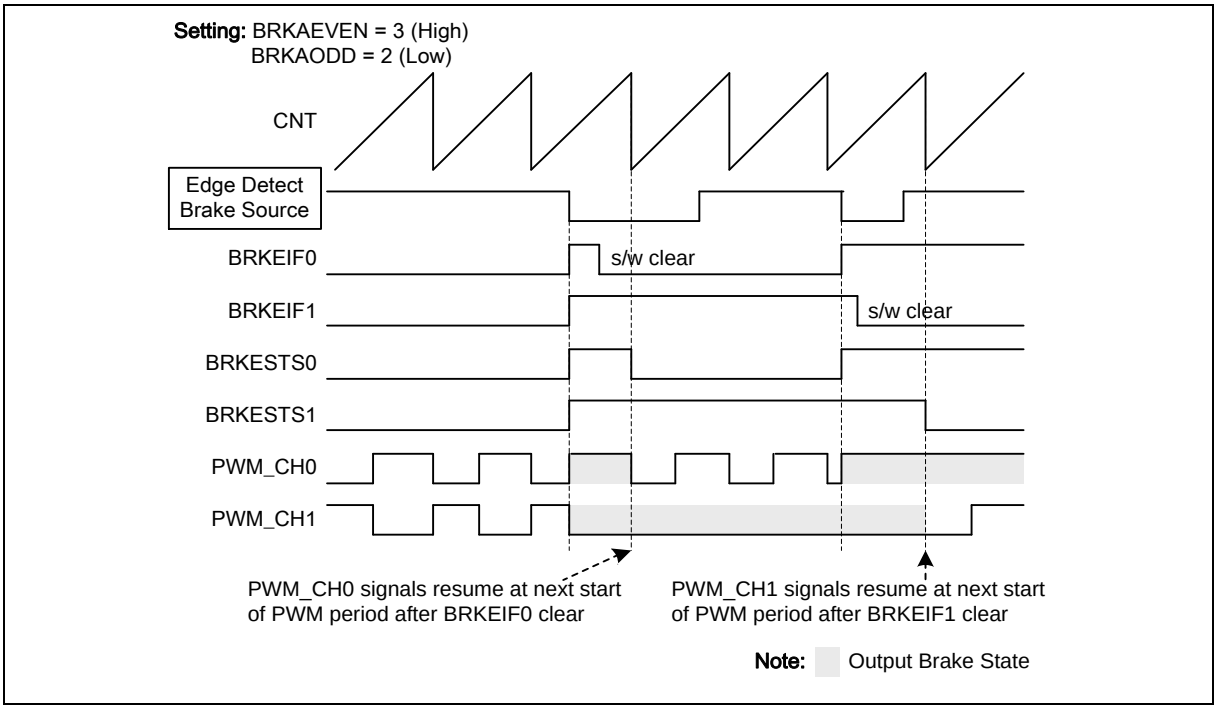


Figure 6.12-26 Edge Detector Waveform for PWMx_CH0 and PWMx_CH1 Pair

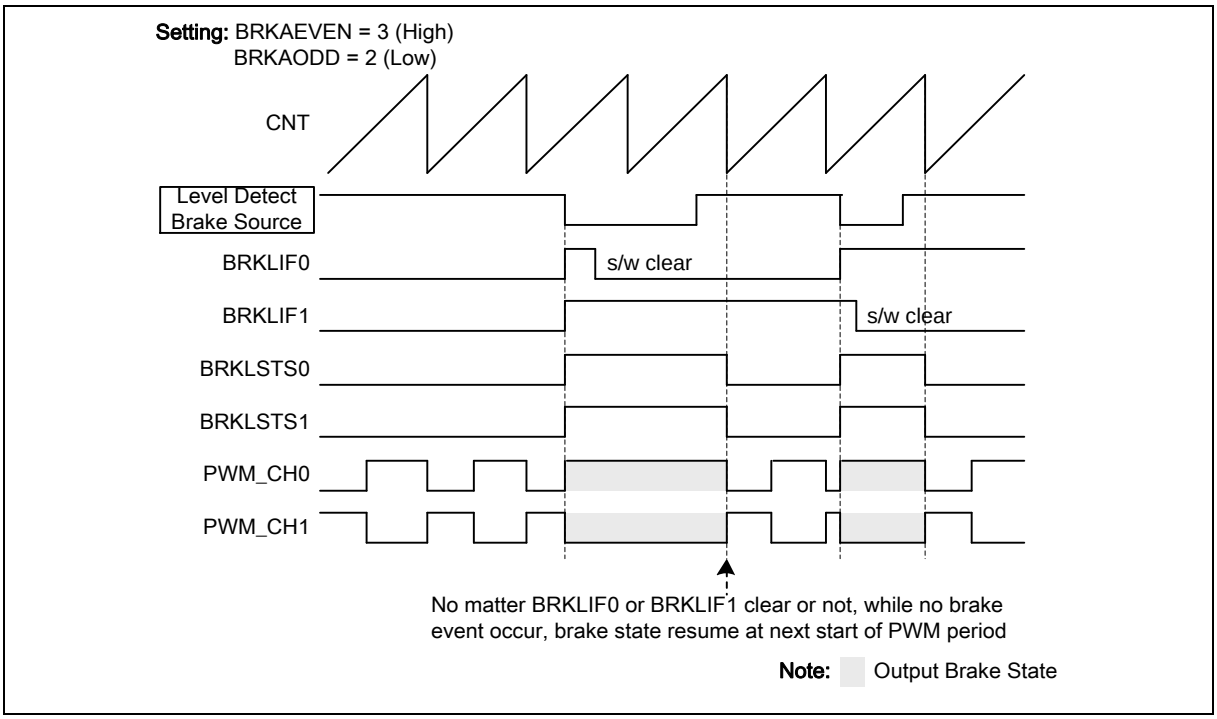


Figure 6.12-27 Level Detector Waveform for PWMx_CH0 and PWMx_CH1 Pair

The two kinds of detectors detect the same six brake sources: two from external input signals, two from analog comparators(ACMP), and one from system fail, and one from software triggered, that are shown in Figure 6.12-28. 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 6.12-29 shows that by setting corresponding enable bits, the enabled system fail condition can be one of the sources to issue the Brake system fail to the PWM brake.

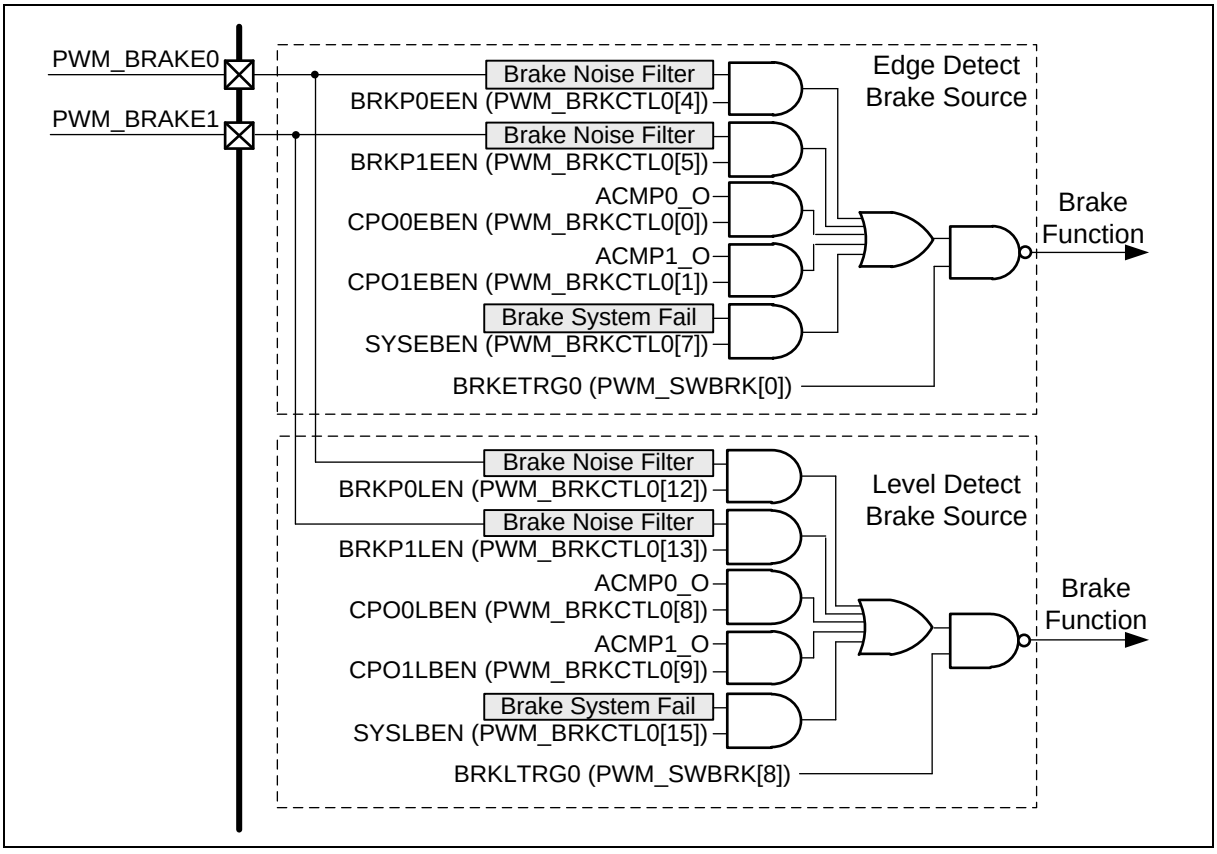


Figure 6.12-28 Brake Source Block Diagram

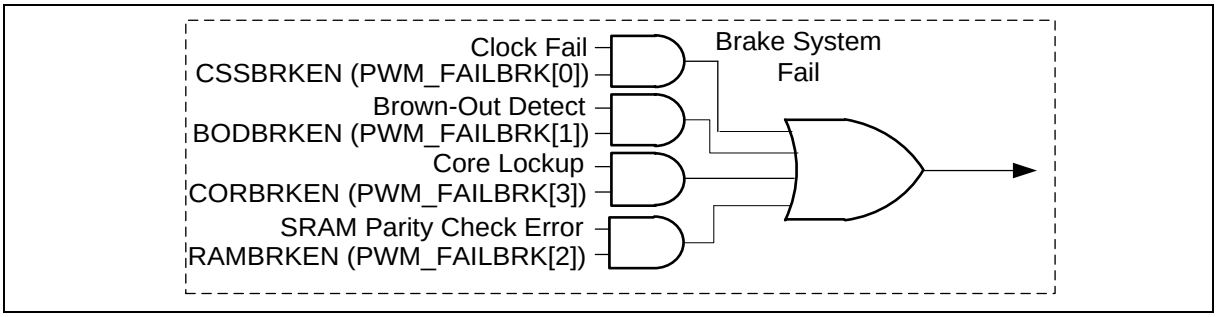


Figure 6.12-29 Brake System Fail Block Diagram

6.12.5.20 Polarity Control

Each PWM port, from PWM_CH0 to PWM_CH5, has an independent polarity control module to configure the polarity of the active state of the PWM output. By default, the PWM output is active high. This implies the PWM OFF state is low and ON state is high. This definition is variable through setting the PWM Negative Polarity Control Register (PWM_POLCTL), for each individual PWM channel. Figure 6.12-30 shows the initial state before PWM starting with different polarity settings.

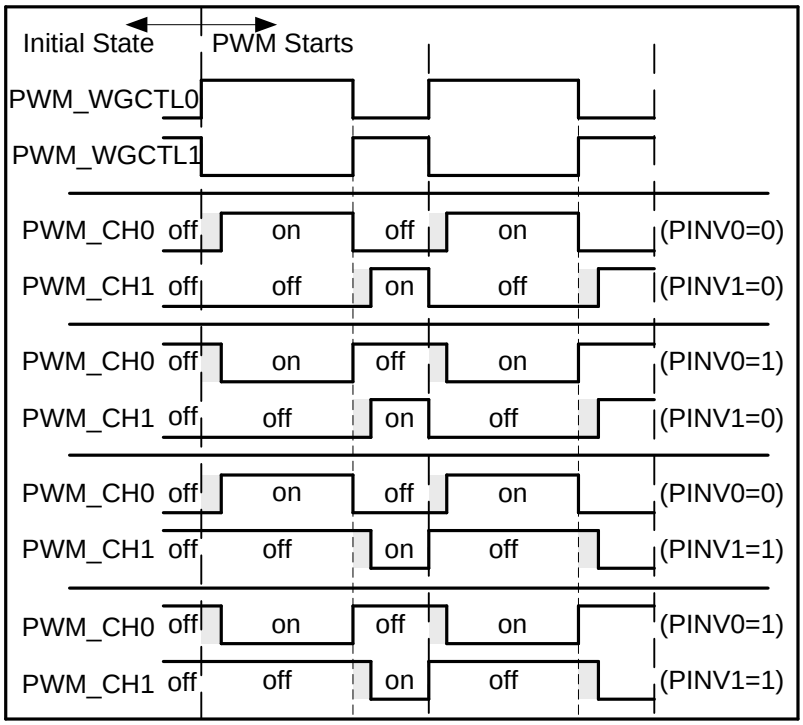


Figure 6.12-30 Initial State and Polarity Control with Rising Edge Dead-Time Insertion

6.12.5.21 Synchronous start function

The synchronous start function can be enabled when SSEN0 (PWM_SSCTL[0]) is set. User can select synchronous source which is from PWM0 or PWM1 by SSRC (PWM_SSCTL[9:8]). The selected PWM channels (include channel0 to channel5 of each PWM) will start counting at the same time once the synchronous start function is enabled and set CNTSEN (PWM_SSTRG). It is noted that set CNTSEN (PWM_SSTRG) will also set the counter enable bit (CNTENn, n denotes channel 0 to 5) to start counting.

6.12.5.22 PWM Interrupt Generator

There are three independent interrupts for each PWM as shown in Figure 6.12-31.

The 1st PWM interrupt (PWM_INT) comes from PWM complementary pair events. The counter can generate the Zero point Interrupt Flag ZIFn (PWM_INTSTS0[n], n=0,2,4) and the Period point Interrupt Flag PIFn (PWM_INTSTS0[n+8], n=0,2,4). When PWM channel n's counter equals to the comparator value stored in PWM_CMPDATn, the different interrupt flags will be triggered depending on the counting direction. If the matching occurs at up-count direction, the Up Interrupt Flag CMPUIFn (PWM_INTSTS0[21:16]) is set and if matching at the opposite direction, the Down Interrupt Flag CMPDIFn (PWM_INTSTS0[29:24]) is set. If the corresponding interrupt enable bits are set, the trigger events will generates interrupt signals.

The 2nd interrupt is the capture interrupt (CAP_INT). It shares the PWM_INT vector in NVIC. The CAP_INT can be generated when the CRLIFn (PWM_CAPIF[5:0]) is triggered and the Capture Rising Interrupt Enable bit CAPRIENn (PWM_CAPIEN[5:0]) is set to 1. Or in the falling edge condition, the CFLIFn (PWM_CAPIF[13:8]) can be triggered when the Capture Falling Interrupt Enable bit CAPFIENn (PWM_CAPIEN[13:8]) is set to 1.

The last one is the brake interrupt (BRK_INT). The detail of the BRK_INT is described in the PWM

Brake section.

Figure 6.12-31 demonstrates the architecture of the PWM interrupts.

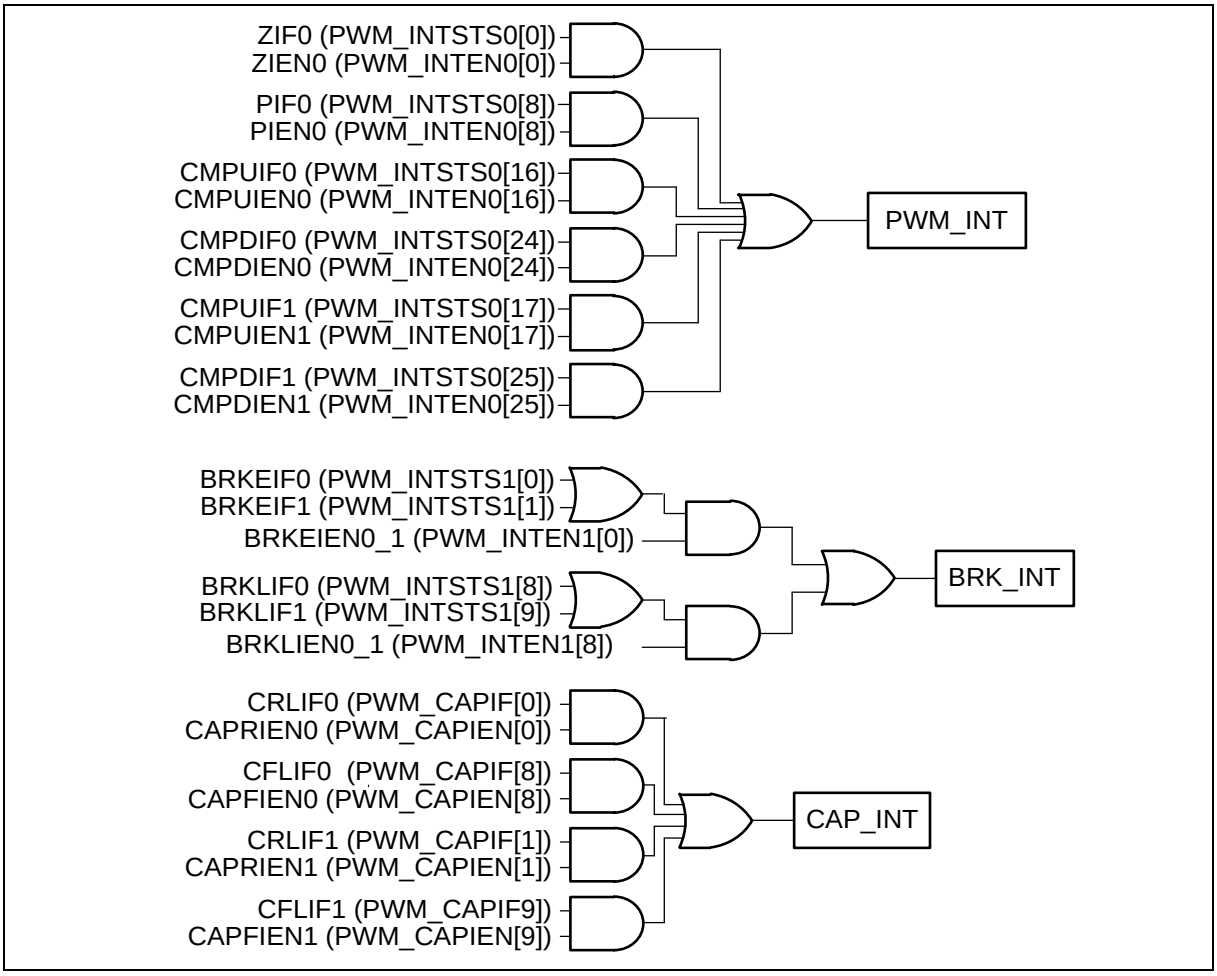


Figure 6.12-31 PWM_CH0 and PWM_CH1 Pair Interrupt Architecture Diagram

6.12.5.23 PWM Trigger ADC Generator

PWM can be one of the ADC conversion trigger source. Each PWM paired channels share the same trigger source. Setting TRGSELn is to select the trigger sources, where TRGSELn is TRGSEL0, TRGSEL1, …, and TRGSEL5, which are located in PWM_ADCTS0[3:0], PWM_ADCTS0[11:8], PWM_ADCTS0[19:16], PWM_ADCTS0[27:24], PWM_ADCTS1[3:0] and PWM_ADCTS1[11:8], respectively. Setting TRGENn is to enable the trigger output to ADC, where TRGENn is TRGEN0, TRGEN1, …, TRGEN5, which are located in PWM_ADCTS0[7], PWM_ADCTS0[15], PWM_ADCTS0[23], PWM_ADCTS0[31], PWM_ADCTS1[7] and PWM_ADCTS1[15], respectively. The number n (n = 0,1, ..,5) denotes PWM channel number.

There are 7 PWM events can be selected as the trigger source for one pair of channels. Figure 6.12-32 is an example of PWM_CH0 and PWM_CH1. PWM can trigger ADC to start conversion in different timings by setting PERIOD and CMPDAT. Figure 6.12-33 is the trigger ADC timing waveform in the up-down counter type.

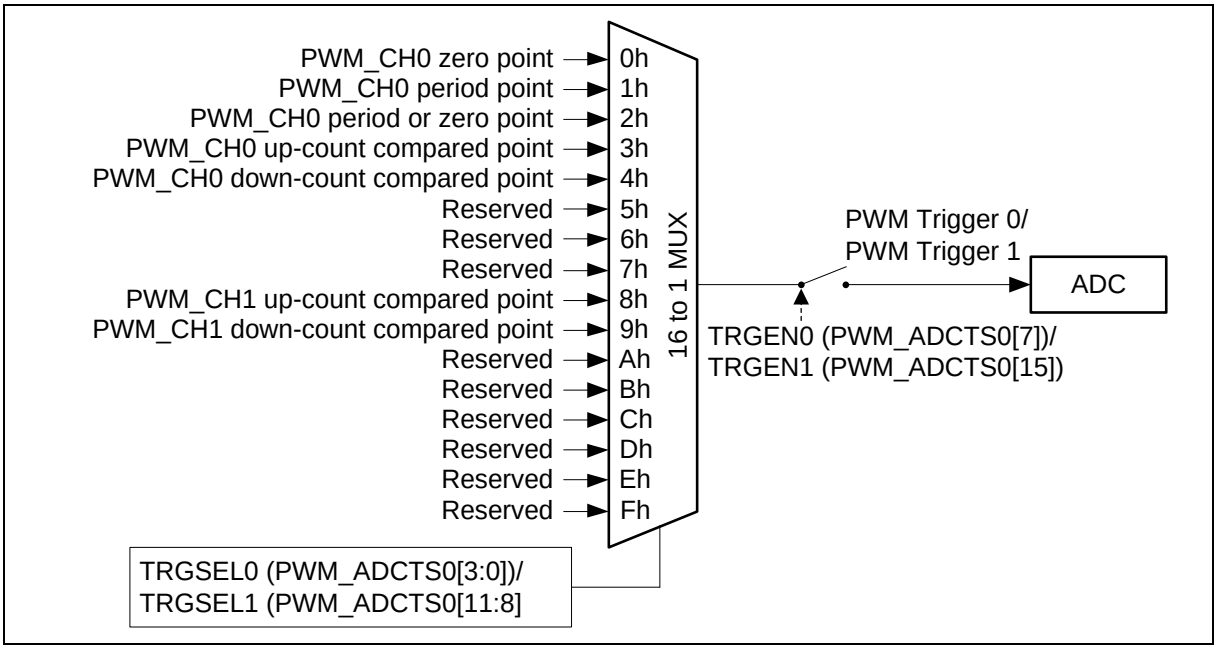


Figure 6.12-32 PWMx_CH0 and PWMx_CH1 Pair Trigger ADC Block Diagram

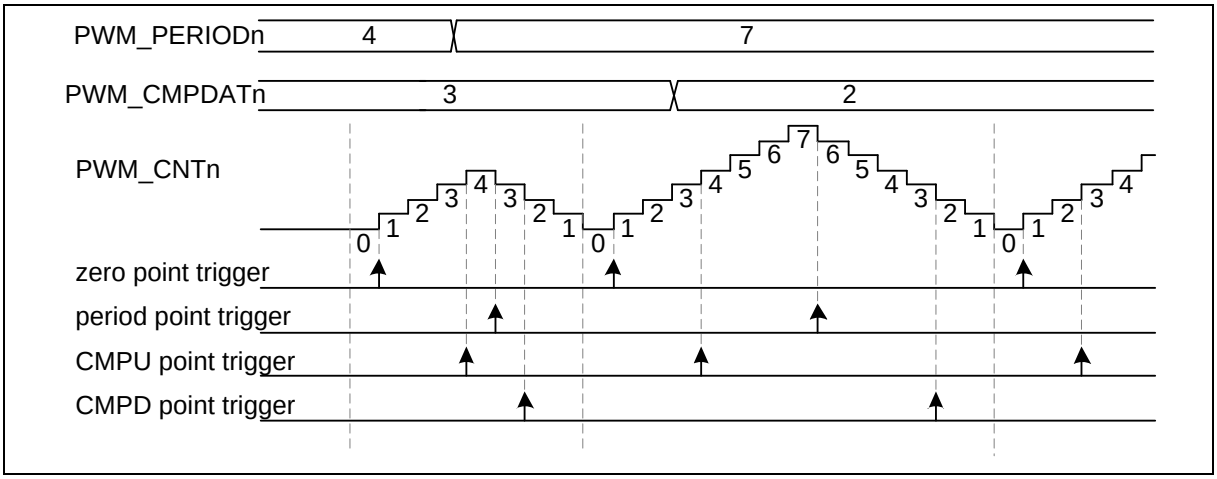


Figure 6.12-33 PWM Trigger ADC in Up-Down Counter Type Timing Waveform

6.12.5.24 Capture Operation

The channels of the capture input and the PWM output share the same pin and counter. The counter can operate in up or down counter type. The capture function will always latch the PWM counter to the RCAPDATn (PWM_RCAPDATn[15:0]) bits or the FCAPDATn (PWM_FCAPDATn[15:0]) bits, if the input channel has a rising transition or a falling transition, respectively. The capture function will also generate an interrupt CAP_INT (using PWM_INT vector) if the rising or falling latch occurs and the corresponding channel n's rising or falling interrupt enable bits are set, where the CAPRIENn (PWM_CAPIEN[5:0]) bit is for the rising edge and the CAPFIENn (PWM_CAPIEN[13:8]) bit is for the falling edge. When rising or falling latch occurs, the corresponding PWM counter may be reloaded with the value of PWM_PERIODn register, depending on the setting of RCRLDENn or FCRLDENn bits (where RCRLDENn and FCRLDENn are located at PWM_CAPCTL[21:16] and PWM_CAPCTL[29:24], respectively). Note that the corresponding GPIO pins must be configured as the capture function by enable the CAPINENn (PWM_CAPINEN[5:0]) bits for the corresponding capture channel n. Figure 6.12-34is the capture block diagram of channel 0.

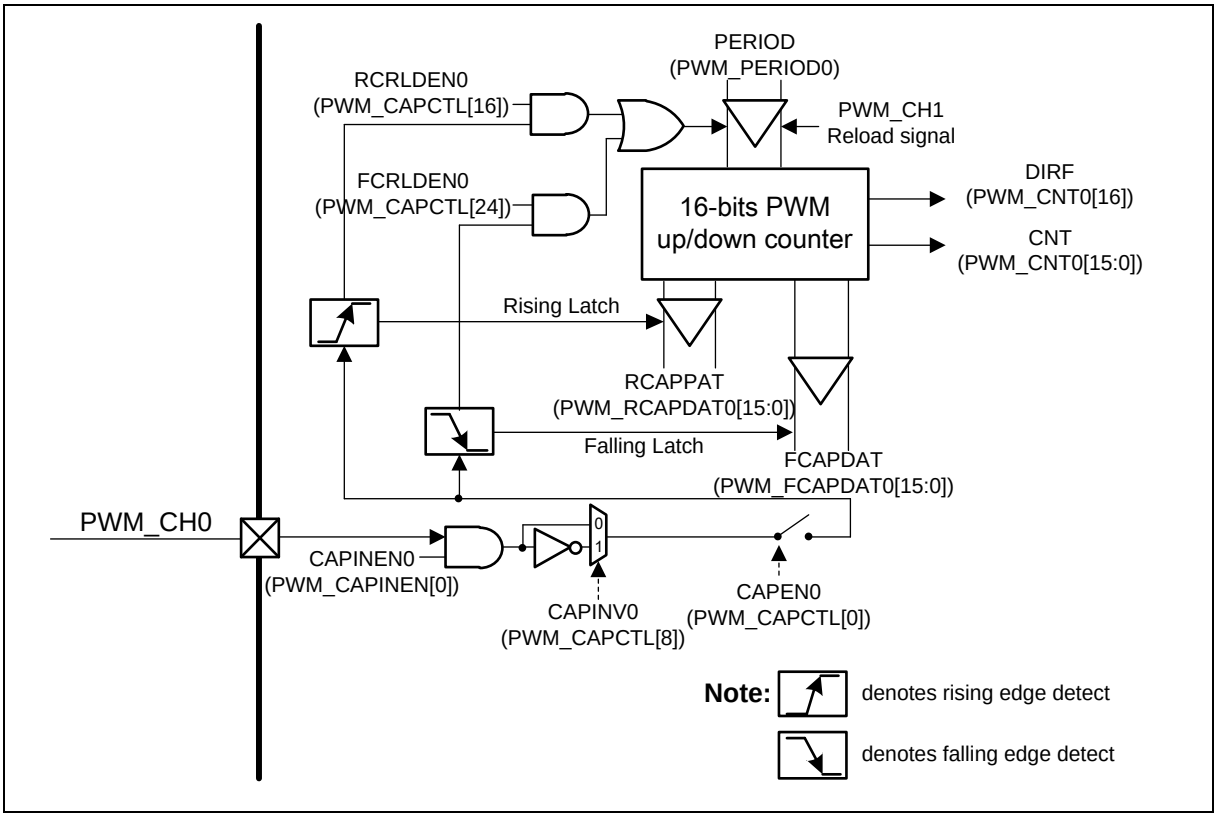


Figure 6.12-34 PWM_CH0 Capture Block Diagram

Figure 6.12-35 illustrates the capture function timing. In this case, the capture counter is set as PWM down counter type and the PERIOD is set to 8 so that the counter counts in the down direction, from 8 to 0. When detecting a falling edge at the capture input pin, the capture function latches the counter value to the PWM_FCAPDATn register. When detecting the rising edge, it latches the counter value to the PWM_RCAPDATn register. In this timing diagram, when the falling edge is detected at the first time, the capture function will reload the counter value from the PERIOD setting because the FCRLDENn bit is enabled. But at the second time, the falling edge does not result in a reload because of the disabled FCRLDENn bit. In this example, the counter also reloads at the rising edge of the capture input because the RCRLDENn bit is enabled, too.

Moreover, if the case is setup as the up counter type, the counter will reload the value zero and count up to the value PERIOD.

Figure 6.12-35 also illustrates the timing example for the interrupt and interrupt flag generation. When the rising edge at channel n is detected, the corresponding CRLIFn (PWM_CAPIF[5:0]) bit is set by hardware. Similarly, a falling edge detection at chnnel n causes the corresponding CFLIFn (PWM_CAPIF[13:8]) bit is set by hardware. CRLIFn and CFLIFn bits can be cleared by software by writing ‘1’. If the CRLIFn bit is set and the CAPRIENn bit is enabled, the capture function generates an interrupt. If the CFLIFn bit is set and the CAPFIENn 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 (PWM_CAPSTS[5:0]) bit will be set to 1 by hardware to indicate the CRLIF flag overrunning. Also, if the falling latch happens again, the same hardware operation occurs for the CFLIF interrupt flag and the Over run status CFLIFOVn (PWM_CAPSTS[13:8]).

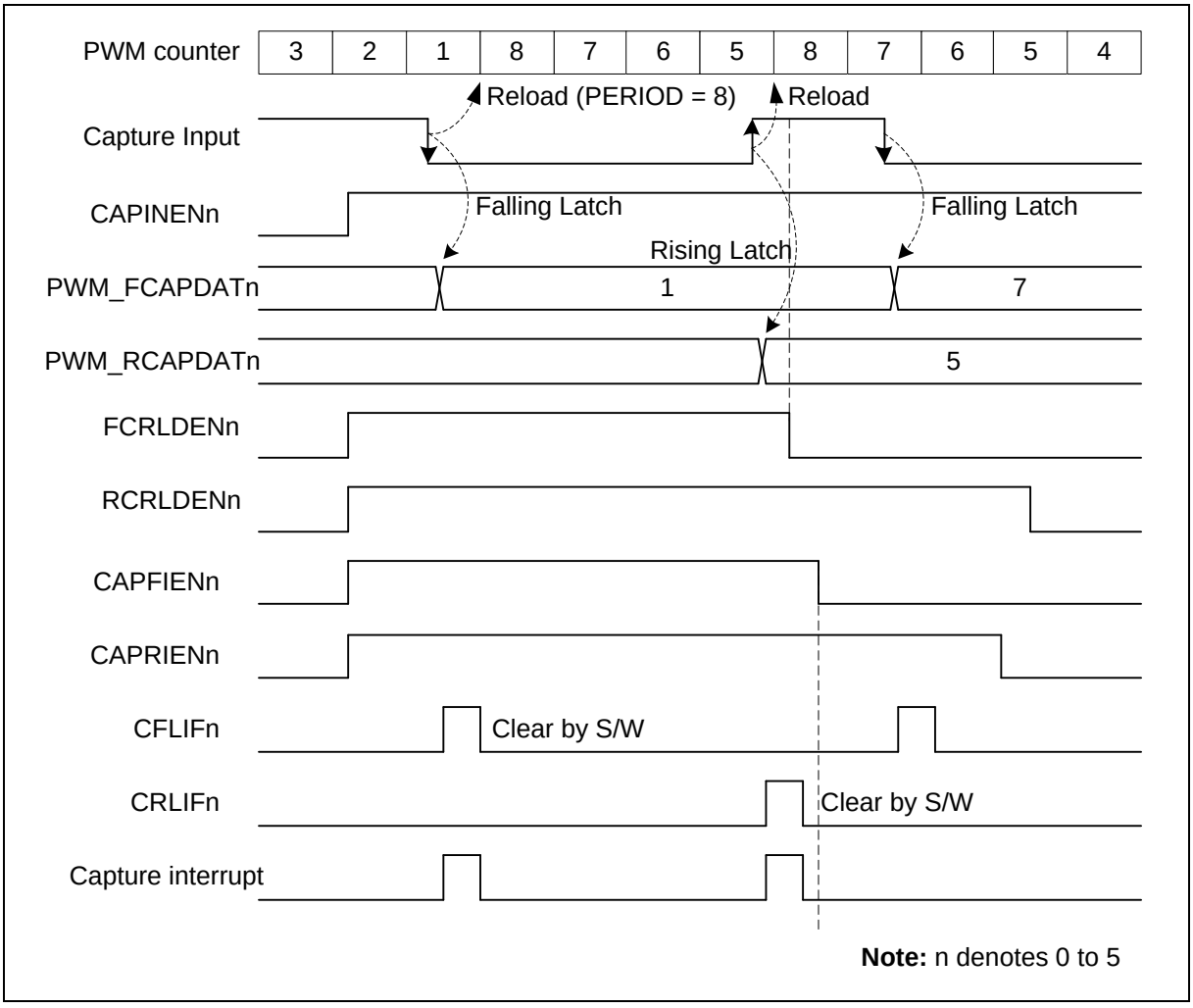


Figure 6.12-35 Capture Operation Waveform

The capture pulse width meeting the following conditions can be calculated according to the formula.

4. The capture positive or negative pulse width is shorter than a counter period.
5. The counter operates in down counter type.
6. 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 (PWM_PERIODn + 1 - PWM_RCAPDATn) PWM counter time, where one PWM counter time is (CLKPSC+1) * PWMx_CLK clock time. In Figure 6.12-35, the low pulse width is 8+1-5 = 4 PWM counter time .

For the positive pulse case, the channel high pulse width is calculated as (PWM_PERIODn + 1 - PWM_FCAPDATn) PWM counter time, where one PWM counter time is (CLKPSC+1) * PWMx_CLK clock time. In Figure 6.12-35, the high pulse width is 8+1-7 = 2 PWM counter time.

6.12.5.25 Capture PDMA Function

The PWM module supports the PDMA transfer function when operating in the capture mode (**Note:** If the PDMA function is not supported, this bit field will become invalid. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.). When the corresponding PDMA enable bit CHENn_m (CHEN0_1 at PWM_PDMACTL[0], CHEN2_3 at PWM_PDMACTL[8] and CHEN4_5 at PWM_PDMACTL[16], where n and m denote complement pair channels) is set, the

capture module will issue a request to PDMA controller when the preceding capture event has happened. The PDMA controller will issue an acknowledgement to the capture module after it has read back the CAPBUF (PWM_PDMACAPn_m[15:0], n, m denotes complement pair channels) register in the capture module and has sent the register value to the memory. By setting CAPMODn_m (CAPMOD0_1 at PWM_PDMACTL[2:1], CAPMOD2_3 at PWM_PDMACTL[10:9] and CAPMOD4_5 at PWM_PDMACTL[18:17]) bits, the PDMA can transfer the rising edge captured data or falling edge captured data or both of them to the memory. When using the PDMA to transfer both of the falling and rising edge data, remember to set CAPORDn_m (CAPORD0_1 at PWM_PDMACTL[3], CAPORD2_3 at PWM_PDMACTL[11] and CAPORD4_5 at PWM_PDMACTL[19]) bit to decide the order of the transferred data (falling edge captured is first or rising edge captured first). The complement pair channels share a PDMA channel. Therefore, a selection bit CHSELn_m (CHSEL0_1 (PWM_PDMACTL[4]), CHSEL2_3 (PWM_PDMACTL[12]) and CHSEL4_5 (PWM_PDMACTL[20])) bit is used to decide either channel n or channel m can be serviced by the PDMA channel.

Figure 6.12-36 is capture PDMA waveform. In this case, the CHSEL0_1 (PWM_PDMACTL[4]) bit is set to 0. Hence the PDMA will service channel 0 for the capture data transfer. CAPMOD0_1 (PWM_PDMACTL[2:1]) bits are set to 3. That means both of the rising and falling edge captured data will be transferred to the memory. The CAPORD0_1 (PWM_PDMACTL[3]) 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 6.12-36, the last assertions of the CAPRIF0 CRLIF0 and CAPFIF0 CFLIF0 signal have some overlap. The PWM_RCAPDAT0 value 11 will be loaded to PWM_PDMACAP0_1 register to wait for transfer but not the PWM_FCAPDAT0 value 6. The PWM_PDMACAP0_1 register saves the data which will be transferred to the memory by PDMA. The HWDATA in this figure denotes the data which are being transferred by PDMA.

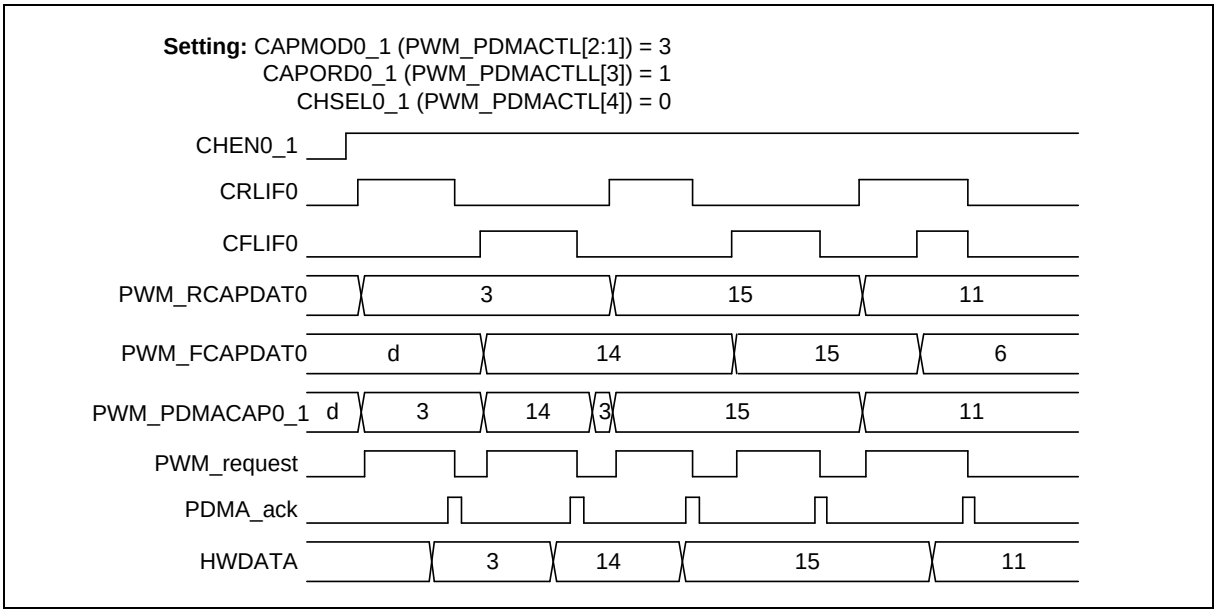


Figure 6.12-36 Capture PDMA Operation Waveform of Channel 0

6.12.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
PWM Base Address: **PWM0_BA = 0x4005_8000** **PWM1_BA = 0x4005_9000**				
PWM_CTL0 **x=0, 1**	PWMx_BA+0x00	R/W	PWM Control Register 0	0x0000_0000
PWM_CTL1 **x=0, 1**	PWMx_BA+0x04	R/W	PWM Control Register 1	0x0000_0000
PWM_CLKSRC **x=0, 1**	PWMx_BA+0x10	R/W	PWM Clock Source Register	0x0000_0000
PWM_CLKPSC0_1 **x=0, 1**	PWMx_BA+0x14	R/W	PWM Clock Prescale Register 0/1	0x0000_0000
PWM_CLKPSC2_3 **x=0, 1**	PWMx_BA+0x18	R/W	PWM Clock Prescale Register 2/3	0x0000_0000
PWM_CLKPSC4_5 **x=0, 1**	PWMx_BA+0x1C	R/W	PWM Clock Prescale Register 4/5	0x0000_0000
PWM_CNTEN **x=0, 1**	PWMx_BA+0x20	R/W	PWM Counter Enable Register	0x0000_0000
PWM_CNTCLR **x=0, 1**	PWMx_BA+0x24	R/W	PWM Clear Counter Register	0x0000_0000
PWM_PERIOD0 **x=0, 1**	PWMx_BA+0x30	R/W	PWM Period Register 0	0x0000_0000
PWM_PERIOD2 **x=0, 1**	PWMx_BA+0x38	R/W	PWM Period Register 2	0x0000_0000
PWM_PERIOD4 **x=0, 1**	PWMx_BA+0x40	R/W	PWM Period Register 4	0x0000_0000
PWM_CMPDAT0 **x=0, 1**	PWMx_BA+0x50	R/W	PWM Comparator Register 0	0x0000_0000
PWM_CMPDAT1 **x=0, 1**	PWMx_BA+0x54	R/W	PWM Comparator Register 1	0x0000_0000
PWM_CMPDAT2 **x=0, 1**	PWMx_BA+0x58	R/W	PWM Comparator Register 2	0x0000_0000
PWM_CMPDAT3 **x=0, 1**	PWMx_BA+0x5C	R/W	PWM Comparator Register 3	0x0000_0000
PWM_CMPDAT4 **x=0, 1**	PWMx_BA+0x60	R/W	PWM Comparator Register 4	0x0000_0000
PWM_CMPDAT5 **x=0, 1**	PWMx_BA+0x64	R/W	PWM Comparator Register 5	0x0000_0000
PWM_DTCTL0_1	PWMx_BA+0x70	R/W	PWM Dead-time Control Register 0/1	0x0000_0000

x=0, 1				
PWM_DTCTL2_3 **x=0, 1**	PWMx_BA+0x74	R/W	PWM Dead-time Control Register 2/3	0x0000_0000
PWM_DTCTL4_5 **x=0, 1**	PWMx_BA+0x78	R/W	PWM Dead-time Control Register 4/5	0x0000_0000
PWM_CNT0 **x=0, 1**	PWMx_BA+0x90	R	PWM Counter Register 0	0x0000_0000
PWM_CNT2 **x=0, 1**	PWMx_BA+0x98	R	PWM Counter Register 2	0x0000_0000
PWM_CNT4 **x=0, 1**	PWMx_BA+0xA0	R	PWM Counter Register 4	0x0000_0000
PWM_WGCTL0 **x=0, 1**	PWMx_BA+0xB0	R/W	PWM Generation Register 0	0x0000_0000
PWM_WGCTL1 **x=0, 1**	PWMx_BA+0xB4	R/W	PWM Generation Register 1	0x0000_0000
PWM_MSKEN **x=0, 1**	PWMx_BA+0xB8	R/W	PWM Mask Enable Register	0x0000_0000
PWM_MSK **x=0, 1**	PWMx_BA+0xBC	R/W	PWM Mask Data Register	0x0000_0000
PWM_BNF **x=0, 1**	PWMx_BA+0xC0	R/W	PWM Brake Noise Filter Register	0x0000_0000
PWM_FAILBRK **x=0, 1**	PWMx_BA+0xC4	R/W	PWM System Fail Brake Control Register	0x0000_0000
PWM_BRKCTL0_1 **x=0, 1**	PWMx_BA+0xC8	R/W	PWM Brake Edge Detect Control Register 0/1	0x0000_0000
PWM_BRKCTL2_3 **x=0, 1**	PWMx_BA+0xCC	R/W	PWM Brake Edge Detect Control Register 2/3	0x0000_0000
PWM_BRKCTL4_5 **x=0, 1**	PWMx_BA+0xD0	R/W	PWM Brake Edge Detect Control Register 4/5	0x0000_0000
PWM_POLCTL **x=0, 1**	PWMx_BA+0xD4	R/W	PWM Pin Polar Inverse Register	0x0000_0000
PWM_POEN **x=0, 1**	PWMx_BA+0xD8	R/W	PWM Output Enable Register	0x0000_0000
PWM_SWBRK **x=0, 1**	PWMx_BA+0xDC	W	PWM Software Brake Control Register	0x0000_0000
PWM_INTEN0 **x=0, 1**	PWMx_BA+0xE0	R/W	PWM Interrupt Enable Register 0	0x0000_0000
PWM_INTEN1 **x=0, 1**	PWMx_BA+0xE4	R/W	PWM Interrupt Enable Register 1	0x0000_0000
PWM_INTSTS0 **x=0, 1**	PWMx_BA+0xE8	R/W	PWM Interrupt Flag Register 0	0x0000_0000

PWM_INTSTS1 **x=0, 1**	PWMx_BA+0xEC	R/W	PWM Interrupt Flag Register 1	0x0000_0000
PWM_ADCTS0 **x=0, 1**	PWMx_BA+0xF8	R/W	PWM Trigger ADC Source Select Register 0	0x0000_0000
PWM_ADCTS1 **x=0, 1**	PWMx_BA+0xFC	R/W	PWM Trigger ADC Source Select Register 1	0x0000_0000
PWM_SSCTL **x=0, 1**	PWMx_BA+0x110	R/W	PWM Synchronous Start Control Register	0x0000_0000
PWM_SSTRG **x=0, 1**	PWMx_BA+0x114	W	PWM Synchronous Start Trigger Register	0x0000_0000
PWM_STATUS **x=0, 1**	PWMx_BA+0x120	R/W	PWM Status Register	0x0000_0000
PWM_CAPINEN **x=0, 1**	PWMx_BA+0x200	R/W	PWM Capture Input Enable Register	0x0000_0000
PWM_CAPCTL **x=0, 1**	PWMx_BA+0x204	R/W	PWM Capture Control Register	0x0000_0000
PWM_CAPSTS **x=0, 1**	PWMx_BA+0x208	R	PWM Capture Status Register	0x0000_0000
PWM_RCAPDAT0 **x=0, 1**	PWMx_BA+0x20C	R	PWM Rising Capture Data Register 0	0x0000_0000
PWM_FCAPDAT0 **x=0, 1**	PWMx_BA+0x210	R	PWM Falling Capture Data Register 0	0x0000_0000
PWM_RCAPDAT1 **x=0, 1**	PWMx_BA+0x214	R	PWM Rising Capture Data Register 1	0x0000_0000
PWM_FCAPDAT1 **x=0, 1**	PWMx_BA+0x218	R	PWM Falling Capture Data Register 1	0x0000_0000
PWM_RCAPDAT2 **x=0, 1**	PWMx_BA+0x21C	R	PWM Rising Capture Data Register 2	0x0000_0000
PWM_FCAPDAT2 **x=0, 1**	PWMx_BA+0x220	R	PWM Falling Capture Data Register 2	0x0000_0000
PWM_RCAPDAT3 **x=0, 1**	PWMx_BA+0x224	R	PWM Rising Capture Data Register 3	0x0000_0000
PWM_FCAPDAT3 **x=0, 1**	PWMx_BA+0x228	R	PWM Falling Capture Data Register 3	0x0000_0000
PWM_RCAPDAT4 **x=0, 1**	PWMx_BA+0x22C	R	PWM Rising Capture Data Register 4	0x0000_0000
PWM_FCAPDAT4 **x=0, 1**	PWMx_BA+0x230	R	PWM Falling Capture Data Register 4	0x0000_0000
PWM_RCAPDAT5 **x=0, 1**	PWMx_BA+0x234	R	PWM Rising Capture Data Register 5	0x0000_0000
PWM_FCAPDAT5	PWMx_BA+0x238	R	PWM Falling Capture Data Register 5	0x0000_0000

x=0, 1				
PWM_PDMACTL **x=0, 1**	PWMx_BA+0x23C	R/W	PWM PDMA Control Register	0x0000_0000
PWM_PDMACAP0_1 **x=0, 1**	PWMx_BA+0x240	R	PWM Capture Channel 01 PDMA Register	0x0000_0000
PWM_PDMACAP2_3 **x=0, 1**	PWMx_BA+0x244	R	PWM Capture Channel 23 PDMA Register	0x0000_0000
PWM_PDMACAP4_5 **x=0, 1**	PWMx_BA+0x248	R	PWM Capture Channel 45 PDMA Register	0x0000_0000
PWM_CAPIEN **x=0, 1**	PWMx_BA+0x250	R/W	PWM Capture Interrupt Enable Register	0x0000_0000
PWM_CAPIF **x=0, 1**	PWMx_BA+0x254	R/W	PWM Capture Interrupt Flag Register	0x0000_0000
PWM_PBUF0 **x=0, 1**	PWMx_BA+0x304	R	PWM PERIOD0 Buffer	0x0000_0000
PWM_PBUF2 **x=0, 1**	PWMx_BA+0x30C	R	PWM PERIOD2 Buffer	0x0000_0000
PWM_PBUF4 **x=0, 1**	PWMx_BA+0x314	R	PWM PERIOD4 Buffer	0x0000_0000
PWM_CMPBUF0 **x=0, 1**	PWMx_BA+0x31C	R	PWM CMPDAT0 Buffer	0x0000_0000
PWM_CMPBUF1 **x=0, 1**	PWMx_BA+0x320	R	PWM CMPDAT1 Buffer	0x0000_0000
PWM_CMPBUF2 **x=0, 1**	PWMx_BA+0x324	R	PWM CMPDAT2 Buffer	0x0000_0000
PWM_CMPBUF3 **x=0, 1**	PWMx_BA+0x328	R	PWM CMPDAT3 Buffer	0x0000_0000
PWM_CMPBUF4 **x=0, 1**	PWMx_BA+0x32C	R	PWM CMPDAT4 Buffer	0x0000_0000
PWM_CMPBUF5 **x=0, 1**	PWMx_BA+0x330	R	PWM CMPDAT5 Buffer	0x0000_0000

6.12.7 Register Description

PWM Control Register 0 (PWM_CTL0)

Register	Offset	R/W	Description	Reset Value
PWM_CTL0	PWMx_BA+0x00	R/W	PWM Control Register 0	0x0000_0000

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

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

PWM Control Register 1 (PWM_CTL1)

Register	Offset	R/W	Description	Reset Value
PWM_CTL1	PWMx_BA+0x04	R/W	PWM Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					OUTMODE4	OUTMODE2	OUTMODE0
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						CNTTYPE4	
7	6	5	4	3	2	1	0
Reserved		CNTTYPE2		Reserved		CNTTYPE0	

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:24]	**OUTMODEn**	**PWM Output Mode** Each bit n controls the output mode of corresponding PWM channel n. 0 = PWM independent mode. 1 = PWM complementary mode. **Note:** When operating in group function, these bits must all set to the same mode.
[23:10]	**Reserved**	Reserved.
[9:8]	**CNTTYPE4**	**PWM Counter Behavior Type 4** The two bits control channel5 and channel4 00 = Up counter type (supported in capture mode). 01 = Down count type (supported in capture mode). 10 = Up-down counter type. 11 = Reserved.
[7:6]	**Reserved**	Reserved.
[5:4]	**CNTTYPE2**	**PWM Counter Behavior Type 2** The two bits control channel3 and channel2 00 = Up counter type (supported in capture mode). 01 = Down count type (supported in capture mode). 10 = Up-down counter type. 11 = Reserved.
[3:2]	**Reserved**	Reserved.
[1:0]	**CNTTYPE0**	**PWM Counter Behavior Type 0** The two bits control channel1 and channel0 00 = Up counter type (supported in capture mode). 01 = Down count type (supported in capture mode).

		10 = Up-down counter type. 11 = Reserved.

PWM Clock Source Register (PWM_CLKSRC)

Register	Offset	R/W	Description	Reset Value
PWM_CLKSRC	PWMx_BA+0x10	R/W	PWM Clock Source Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					ECLKSRC4		
15	14	13	12	11	10	9	8
Reserved					ECLKSRC2		
7	6	5	4	3	2	1	0
Reserved					ECLKSRC0		

Bits	Description	
[31:19]	**Reserved**	Reserved.
[18:16]	**ECLKSRC4**	**PWM_CH45 External Clock Source Select** 000 = PWMx_CLK, x denotes 0 or 1. 001 = TIMER0 overflow. 010 = TIMER1 overflow. 011 = TIMER2 overflow. 100 = TIMER3 overflow. Others = Reserved.
[15:11]	**Reserved**	Reserved.
[10:8]	**ECLKSRC2**	**PWM_CH23 External Clock Source Select** 000 = PWMx_CLK, x denotes 0 or 1. 001 = TIMER0 overflow. 010 = TIMER1 overflow. 011 = TIMER2 overflow. 100 = TIMER3 overflow. Others = Reserved.
[7:3]	**Reserved**	Reserved.
[2:0]	**ECLKSRC0**	**PWM_CH01 External Clock Source Select** 000 = PWMx_CLK, x denotes 0 or 1. 001 = TIMER0 overflow. 010 = TIMER1 overflow. 011 = TIMER2 overflow. 100 = TIMER3 overflow. Others = Reserved.

PWM Clock Pre-scale Register 0_1, 2_3, 4_5 (PWM_CLKPSC0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
PWM_CLKPSC0_1	PWMx_BA+0x14	R/W	PWM Clock Prescale Register 0/1	0x0000_0000
PWM_CLKPSC2_3	PWMx_BA+0x18	R/W	PWM Clock Prescale Register 2/3	0x0000_0000
PWM_CLKPSC4_5	PWMx_BA+0x1C	R/W	PWM 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**	**PWM Counter Clock Prescale** The clock of PWM counter is decided by clock prescaler. Each PWM pair share one PWM counter clock prescaler. The clock of PWM counter is divided by (CLKPSC+ 1).

PWM Counter Enable Register (PWM_CNTEN)

Register	Offset	R/W	Description	Reset Value
PWM_CNTEN	PWMx_BA+0x20	R/W	PWM Counter Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			CNTEN4	Reserved	CNTEN2	Reserved	CNTEN0

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**CNTEN4**	**PWM Counter Enable Bit 4** 0 = PWM Counter and clock prescaler Stop Running. 1 = PWM Counter and clock prescaler Start Running.
[3]	**Reserved**	Reserved.
[2]	**CNTEN2**	**PWM Counter Enable Bit 2** 0 = PWM Counter and clock prescaler Stop Running. 1 = PWM Counter and clock prescaler Start Running.
[1]	**Reserved**	Reserved.
[0]	**CNTEN0**	**PWM Counter Enable Bit 0** 0 = PWM Counter and clock prescaler Stop Running. 1 = PWM Counter and clock prescaler Start Running.

PWM Clear Counter Register (PWM_CNTCLR)

Register	Offset	R/W	Description	Reset Value
PWM_CNTCLR	PWMx_BA+0x24	R/W	PWM Clear Counter Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			CNTCLR4	Reserved	CNTCLR2	Reserved	CNTCLR0

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**CNTCLR4**	**Clear PWM Counter Control Bit 4** It is automatically cleared by hardware. 0 = No effect. 1 = Clear 16-bit PWM counter to 0000H.
[3]	**Reserved**	Reserved.
[2]	**CNTCLR2**	**Clear PWM Counter Control Bit 2** It is automatically cleared by hardware. 0 = No effect. 1 = Clear 16-bit PWM counter to 0000H.
[1]	**Reserved**	Reserved.
[0]	**CNTCLR0**	**Clear PWM Counter Control Bit 0** It is automatically cleared by hardware. 0 = No effect. 1 = Clear 16-bit PWM counter to 0000H.

PWM Period Register 0, 2, 4 (PWM_PERIOD0, 2, 4)

Register	Offset	R/W	Description	Reset Value
PWM_PERIOD0	PWMx_BA+0x30	R/W	PWM Period Register 0	0x0000_0000
PWM_PERIOD2	PWMx_BA+0x38	R/W	PWM Period Register 2	0x0000_0000
PWM_PERIOD4	PWMx_BA+0x40	R/W	PWM Period Register 4	0x0000_0000

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

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

PWM Comparator Register 0~5 (PWM_CMPDAT0~5)

Register	Offset	R/W	Description	Reset Value
PWM_CMPDAT0	PWMx_BA+0x50	R/W	PWM Comparator Register 0	0x0000_0000
PWM_CMPDAT1	PWMx_BA+0x54	R/W	PWM Comparator Register 1	0x0000_0000
PWM_CMPDAT2	PWMx_BA+0x58	R/W	PWM Comparator Register 2	0x0000_0000
PWM_CMPDAT3	PWMx_BA+0x5C	R/W	PWM Comparator Register 3	0x0000_0000
PWM_CMPDAT4	PWMx_BA+0x60	R/W	PWM Comparator Register 4	0x0000_0000
PWM_CMPDAT5	PWMx_BA+0x64	R/W	PWM Comparator Register 5	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CMP**	**PWM Comparator Register** CMP is used to compare with CNTR to generate PWM waveform, interrupt and trigger ADC. In independent mode, PWM_CMPDAT0~5 denote as 6 independent PWM_CH0~5 compared point. In complementary mode, PWM_CMPDAT0, 2, 4 denote as first compared point, and PWM_CMPDAT1, 3, 5 denote as second compared point for the corresponding 3 complementary pairs PWM_CH0 and PWM_CH1, PWM_CH2 and PWM_CH3, PWM_CH4 and PWM_CH5.

PWM Dead-time Control Register 0_1, 2_3, 4_5 (PWM_DTCTL0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
PWM_DTCTL0_1	PWMx_BA+0x70	R/W	PWM Dead-time Control Register 0/1	0x0000_0000
PWM_DTCTL2_3	PWMx_BA+0x74	R/W	PWM Dead-time Control Register 2/3	0x0000_0000
PWM_DTCTL4_5	PWMx_BA+0x78	R/W	PWM Dead-time Control Register 4/5	0x0000_0000

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

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**DTCKSEL**	**Dead-time Clock Select (Write Protect)** 0 = Dead-time clock source from PWM_CLK. 1 = Dead-time clock source from prescaler output. **Note:** This bit is write protected. Refer to REGWRPROT register.
[23:17]	**Reserved**	Reserved.
[16]	**DTEN**	**Enable Dead-time Insertion for PWM Pair (PWM_CH0, PWM_CH1) (PWM_CH2, PWM_CH3) (PWM_CH4, PWM_CH5) (Write Protect)** Dead-time insertion is only active when this pair of complementary PWM is enabled. If dead- time insertion is inactive, the outputs of pin pair are complementary without any delay. 0 = Dead-time insertion Disabled on the pin pair. 1 = Dead-time insertion Enabled on the pin pair. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[15:12]	**Reserved**	Reserved.
[11:0]	**DTCNT**	**Dead-time Counter (Write Protect)** The dead-time can be calculated from the following formula: DTCKSEL=0: Dead-time = (DTCNT[11:0]+1) * PWM_CLK period. DTCKSEL=1: Dead-time = (DTCNT[11:0]+1) * PWM_CLK period * (CLKPSC+1). **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

PWM Counter Register 0, 2, 4 (PWM_CNT0, 2, 4)

Register	Offset	R/W	Description	Reset Value
PWM_CNT0	PWMx_BA+0x90	R	PWM Counter Register 0	0x0000_0000
PWM_CNT2	PWMx_BA+0x98	R	PWM Counter Register 2	0x0000_0000
PWM_CNT4	PWMx_BA+0xA0	R	PWM Counter Register 4	0x0000_0000

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

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

PWM Generation Register 0 (PWM_WGCTL0)

Register	Offset	R/W	Description	Reset Value
PWM_WGCTL0	PWMx_BA+0xB0	R/W	PWM Generation Register 0	0x0000_0000

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

Bits	Description	
[31:28]	**Reserved**	Reserved.
[17+2n:16+2n] n=0,1..5	**PRDPCTLn**	**PWM Period (Center) Point Control** 00 = Do nothing. 01 = PWM period (center) point output Low. 10 = PWM period (center) point output High. 11 = PWM period (center) point output Toggle. **Note 1:** PWM can control output level when PWM counter counts to (PERIODn+1). **Note 2:** This bit is center point control when PWM counter operating in up-down counter type.
[15:12]	**Reserved**	Reserved.
[1+2n:2n] n=0,1..5	**ZPCTLn**	**PWM Zero Point Control** 00 = Do nothing. 01 = PWM zero point output Low. 10 = PWM zero point output High. 11 = PWM zero point output Toggle. **Note:** PWM can control output level when PWM counter counts to 0.

PWM Generation Register 1 (PWM_WGCTL1)

Register	Offset	R/W	Description	Reset Value
PWM_WGCTL1	PWMx_BA+0xB4	R/W	PWM Generation Register 1	0x0000_0000

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

Bits	Description	
[31:28]	**Reserved**	Reserved.
[17+2n:16+2n] n=0,1..5	**CMPDCTLn**	**PWM Compare Down Point Control** 00 = Do nothing. 01 = PWM compare down point output Low. 10 = PWM compare down point output High. 11 = PWM compare down point output Toggle. **Note 1:** PWM can control output level when PWM counter counts down to CMPDAT. **Note 2:** 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**	**PWM Compare Up Point Control** 00 = Do nothing. 01 = PWM compare up point output Low. 10 = PWM compare up point output High. 11 = PWM compare up point output Toggle. **Note 1:** PWM can control output level when PWM counter counts up to CMPDAT. **Note 2:** In complementary mode, CMPUCTL1, 3, 5 is used as another CMPUCTL for channel 0, 2, 4.

PWM Mask Enable Register (PWM_MSKEN)

Register	Offset	R/W	Description	Reset Value
PWM_MSKEN	PWMx_BA+0xB8	R/W	PWM Mask Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		MSKEN5	MSKEN4	MSKEN3	MSKEN2	MSKEN1	MSKEN0

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

PWM Mask DATA Register (PWM_MSK)

Register	Offset	R/W	Description	Reset Value
PWM_MSK	PWMx_BA+0xBC	R/W	PWM Mask Data Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		MSKDAT5	MSKDAT4	MSKDAT3	MSKDAT2	MSKDAT1	MSKDAT0

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

PWM Brake Noise Filter Register (PWM_BNF)

Register	Offset	R/W	Description	Reset Value
PWM_BNF	PWMx_BA+0xC0	R/W	PWM Brake Noise Filter Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							BK1SRC
23	22	21	20	19	18	17	16
Reserved							BK0SRC
15	14	13	12	11	10	9	8
BRK1PINV	BRK1FCNT			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 PWM0 setting: 0 = Brake 1 pin source come from PWM0_BRAKE1. 1 = Brake 1 pin source come from PWM1_BRAKE1. For PWM1 setting: 0 = Brake 1 pin source come from PWM1_BRAKE1. 1 = Brake 1 pin source come from PWM0_BRAKE1.
[23:17]	**Reserved**	Reserved.
[16]	**BK0SRC**	**Brake 0 Pin Source Select** For PWM0 setting: 0 = Brake 0 pin source come from PWM0_BRAKE0. 1 = Brake 0 pin source come from PWM1_BRAKE0. For PWM1 setting: 0 = Brake 0 pin source come from PWM1_BRAKE0. 1 = Brake 0 pin source come from PWM0_BRAKE0.
[15]	**BRK1PINV**	**Brake 1 Pin Inverse** 0 = The state of pin PWMx_BRAKE1 is passed to the negative edge detector. 1 = The inversed state of pin PWMx_BRAKE1 is passed to the negative edge detector.
[14:12]	**BRK1FCNT**	**Brake 1 Edge Detector Filter Count** The register bits control the Brake1 filter counter to count from 0 to BRK1FCNT.
[11:9]	**BRK1NFSEL**	**Brake 1 Edge Detector Filter Clock Selection** 000 = Filter clock = HCLK. 001 = Filter clock = HCLK/2. 010 = Filter clock = HCLK/4.

		011 = Filter clock = HCLK/8. 100 = Filter clock = HCLK/16. 101 = Filter clock = HCLK/32. 110 = Filter clock = HCLK/64. 111 = Filter clock = HCLK/128.
[8]	**BRK1NFEN**	**PWM Brake 1 Noise Filter Enable Bit** 0 = Noise filter of PWM Brake 1 Disabled. 1 = Noise filter of PWM Brake 1 Enabled.
[7]	**BRK0PINV**	**Brake 0 Pin Inverse** 0 = The state of pin PWMx_BRAKE0 is passed to the negative edge detector. 1 = The inversed state of pin PWMx_BRAKE10 is passed to the negative edge detector.
[6:4]	**BRK0FCNT**	**Brake 0 Edge Detector Filter Count** The register bits control the Brake0 filter counter to count from 0 to BRK1FCNT.
[3:1]	**BRK0NFSEL**	**Brake 0 Edge Detector Filter Clock Selection** 000 = Filter clock = HCLK. 001 = Filter clock = HCLK/2. 010 = Filter clock = HCLK/4. 011 = Filter clock = HCLK/8. 100 = Filter clock = HCLK/16. 101 = Filter clock = HCLK/32. 110 = Filter clock = HCLK/64. 111 = Filter clock = HCLK/128.
[0]	**BRK0NFEN**	**PWM Brake 0 Noise Filter Enable Bit** 0 = Noise filter of PWM Brake 0 Disabled. 1 = Noise filter of PWM Brake 0 Enabled.

PWM System Fail Brake Control Register (PWM_FAILBRK)

Register	Offset	R/W	Description	Reset Value
PWM_FAILBRK	PWMx_BA+0xC4	R/W	PWM System Fail Brake Control Register	0x0000_0000

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

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**CORBRKEN**	**Core Lockup Detection Trigger PWM Brake Function Enable Bit** 0 = Brake Function triggered by Core lockup detection Disabled. 1 = Brake Function triggered by Core lockup detection Enabled.
[2]	**RAMBRKEN**	**SRAM Parity Error Detection Trigger PWM Brake Function Enable Bit** 0 = Brake Function triggered by SRAM parity error detection Disabled. 1 = Brake Function triggered by SRAM parity error detection Enabled.
[1]	**BODBRKEN**	**Brown-out Detection Trigger PWM Brake Function Enable Bit** 0 = Brake Function triggered by BOD Disabled. 1 = Brake Function triggered by BOD Enabled.
[0]	**CSSBRKEN**	**Clock Security System Detection Trigger PWM Brake Function Enable Bit** 0 = Brake Function triggered by CSS detection Disabled. 1 = Brake Function triggered by CSS detection Enabled.

PWM Brake Edge Detect Control Register 0_1, 2_3, 4_5 (PWM_BRKCTL0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
PWM_BRKCTL0_1	PWMx_BA+0xC8	R/W	PWM Brake Edge Detect Control Register 0/1	0x0000_0000
PWM_BRKCTL2_3	PWMx_BA+0xCC	R/W	PWM Brake Edge Detect Control Register 2/3	0x0000_0000
PWM_BRKCTL4_5	PWMx_BA+0xD0	R/W	PWM Brake Edge Detect Control Register 4/5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				BRKAODD		BRKAEVEN	
15	14	13	12	11	10	9	8
SYSLBEN	Reserved	BRKP1LEN	BRKP0LEN	Reserved		CPO1LBEN	CPO0LBEN
7	6	5	4	3	2	1	0
SYSEBEN	Reserved	BRKP1EEN	BRKP0EEN	Reserved		CPO1EBEN	CPO0EBEN

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19:18]	**BRKAODD**	**PWM Brake Action Select for Odd Channel (Write Protect)** 00 = PWM odd channel level-detect brake function not affect channel output. 01 = PWM odd channel output tri-state when level-detect brake happened. 10 = PWM odd channel output low level when level-detect brake happened. 11 = PWM odd channel output high level when level-detect brake happened. **Note:** These bits are write protected. Refer to SYS_REGLCTL register.
[17:16]	**BRKAEVEN**	**PWM Brake Action Select for Even Channel (Write Protect)** 00 = PWM even channel level-detect brake function not affect channel output. 01 = PWM even channel output tri-state when level-detect brake happened. 10 = PWM even channel output low level when level-detect brake happened. 11 = PWM even channel output high level when level-detect brake happened. **Note:** These bits are write protected. Refer to SYS_REGLCTL register.
[15]	**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 = PWMx_BRAKE1 pin as level-detect brake source Disabled. 1 = PWMx_BRAKE1 pin as level-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

[12]	**BRKP0LEN**	**Enable BKP0 Pin As Level-detect Brake Source (Write Protect)** 0 = PWMx_BRAKE0 pin as level-detect brake source Disabled. 1 = PWMx_BRAKE0 pin as level-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[11:10]	**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 register 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 register 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 PWMx_BRAKE1 Pin As Edge-detect Brake Source (Write Protect)** 0 = BKP1 pin as edge-detect brake source Disabled. 1 = BKP1 pin as edge-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[4]	**BRKP0EEN**	**Enable PWMx_BRAKE0 Pin As Edge-detect Brake Source (Write Protect)** 0 = BKP0 pin as edge-detect brake source Disabled. 1 = BKP0 pin as edge-detect brake source Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[3:2]	**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 register 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 register is write protected. Refer to SYS_REGLCTL register.

PWM Pin Polar Inverse Control (PWM_POLCTL)

Register	Offset	R/W	Description	Reset Value
PWM_POLCTL	PWMx_BA+0xD4	R/W	PWM Pin Polar Inverse Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		PINV5	PINV4	PINV3	PINV2	PINV1	PINV0

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**PINVn**	**PWM PIN Polar Inverse Control** The register controls polarity state of PWM output. 0 = PWM output polar inverse Disabled. 1 = PWM output polar inverse Enabled.

PWM Output Enable Register (PWM_POEN)

Register	Offset	R/W	Description	Reset Value
PWM_POEN	PWMx_BA+0xD8	R/W	PWM Output Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		POEN5	POEN4	POEN3	POEN2	POEN1	POEN0

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**POENn**	**PWM Pin Output Enable Bits** 0 = PWM pin at tri-state. 1 = PWM pin in output mode.

PWM Software Brake Control Register (PWM_SWBRK)

Register	Offset	R/W	Description	Reset Value
PWM_SWBRK	PWMx_BA+0xDC	W	PWM Software Brake Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					BRKLTRG4	BRKLTRG2	BRKLTRG0
7	6	5	4	3	2	1	0
Reserved					BRKETRG4	BRKETRG2	BRKETRG0

Bits	Description	
[31:11]	**Reserved**	Reserved.
[8+n/2] n=0,2,4	**BRKLTRGn**	**PWM Level Brake Software Trigger (Write Only) (Write Protect)** Write 1 to this bit will trigger level brake, and set BRKLIFn to 1 in PWM_INTSTS1 register. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[7:3]	**Reserved**	Reserved.
[n/2] n=0,2,4	**BRKETRGn**	**PWM Edge Brake Software Trigger (Write Only) (Write Protect)** Write 1 to this bit will trigger Edge brake, and set BRKEIFn to 1 in PWM_INTSTS1 register. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

PWM Interrupt Enable Register 0 (PWM_INTEN0)

Register	Offset	R/W	Description	Reset Value
PWM_INTEN0	PWMx_BA+0xE0	R/W	PWM Interrupt Enable Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		CMPDIEN5	CMPDIEN4	CMPDIEN3	CMPDIEN2	CMPDIEN1	CMPDIEN0
23	22	21	20	19	18	17	16
Reserved		CMPUIEN5	CMPUIEN4	CMPUIEN3	CMPUIEN2	CMPUIEN1	CMPUIEN0
15	14	13	12	11	10	9	8
Reserved			PIEN4	Reserved	PIEN2	Reserved	PIEN0
7	6	5	4	3	2	1	0
Reserved			ZIEN4	Reserved	ZIEN2	Reserved	ZIEN0

Bits	Description	
[31:30]	**Reserved**	Reserved.
[24+n] n=0,1..5	**CMPDIENn**	**PWM 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**	**PWM Compare Up Count Interrupt Enable Bits** Each bit n controls the corresponding PWM channel n. 0 = Compare up count interrupt Disabled. 1 = Compare up count interrupt Enabled. **Note:** In complementary mode, CMPUIEN1, 3, 5 is used as another CMPUIEN for channel 0, 2, 4.
[15:13]	**Reserved**	Reserved.
[12]	**PIEN4**	**PWM Period Point Interrupt Enable Bit 4** 0 = Period point interrupt Disabled. 1 = Period point interrupt Enabled. **Note:** When up-down counter type, period point means center point.
[11]	**Reserved**	Reserved.
[10]	**PIEN2**	**PWM Period Point Interrupt Enable Bit 2** 0 = Period point interrupt Disabled. 1 = Period point interrupt Enabled. **Note:** When up-down counter type, period point means center point.
[9]	**Reserved**	Reserved.
[8]	**PIEN0**	**PWM Period Point Interrupt Enable Bit 0**

		0 = Period point interrupt Disabled. 1 = Period point interrupt Enabled. **Note:** When up-down counter type, period point means center point.
[7:5]	**Reserved**	Reserved.
[4]	**ZIEN4**	**PWM Zero Point Interrupt Enable Bit 4** 0 = Zero point interrupt Disabled. 1 = Zero point interrupt Enabled. **Note:** Odd channels will read always 0 at complementary mode.
[3]	**Reserved**	Reserved.
[2]	**ZIEN2**	**PWM Zero Point Interrupt Enable Bit 2** 0 = Zero point interrupt Disabled. 1 = Zero point interrupt Enabled. **Note:** Odd channels will read always 0 at complementary mode.
[1]	**Reserved**	Reserved.
[0]	**ZIEN0**	**PWM Zero Point Interrupt Enable Bit 0** 0 = Zero point interrupt Disabled. 1 = Zero point interrupt Enabled. **Note:** Odd channels will read always 0 at complementary mode.

PWM Interrupt Enable Register 1 (PWM_INTEN1)

Register	Offset	R/W	Description	Reset Value
PWM_INTEN1	PWMx_BA+0xE4	R/W	PWM Interrupt Enable Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					BRKLIEN4_5	BRKLIEN2_3	BRKLIEN0_1
7	6	5	4	3	2	1	0
Reserved					BRKEIEN4_5	BRKEIEN2_3	BRKEIEN0_1

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10]	**BRKLIEN4_5**	**PWM Level-detect Brake Interrupt Enable for Channel4/5 (Write Protect)** 0 = Level-detect Brake interrupt for channel4/5 Disabled. 1 = Level-detect Brake interrupt for channel4/5 Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[9]	**BRKLIEN2_3**	**PWM Level-detect Brake Interrupt Enable for Channel2/3 (Write Protect)** 0 = Level-detect Brake interrupt for channel2/3 Disabled. 1 = Level-detect Brake interrupt for channel2/3 Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[8]	**BRKLIEN0_1**	**PWM Level-detect Brake Interrupt Enable for Channel0/1 (Write Protect)** 0 = Level-detect Brake interrupt for channel0/1 Disabled. 1 = Level-detect Brake interrupt for channel0/1 Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.
[7:3]	**Reserved**	Reserved.
[2]	**BRKEIEN4_5**	**PWM Edge-detect Brake Interrupt Enable for Channel4/5 (Write Protect)** 0 = Edge-detect Brake interrupt for channel4/5 Disabled. 1 = Edge-detect Brake interrupt for channel4/5 Enabled. **Note:** This bitr is write protected. Refer to SYS_REGLCTL register.
[1]	**BRKEIEN2_3**	**PWM Edge-detect Brake Interrupt Enable for Channel2/3 (Write Protect)** 0 = Edge-detect Brake interrupt for channel2/3 Disabled. 1 = Edge-detect Brake interrupt for channel2/3 Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

[0]	**BRKEIEN0_1**	**PWM Edge-detect Brake Interrupt Enable for Channel0/1 (Write Protect)** 0 = Edge-detect Brake interrupt for channel0/1 Disabled. 1 = Edge-detect Brake interrupt for channel0/1 Enabled. **Note:** This bit is write protected. Refer to SYS_REGLCTL register.

PWM Interrupt Flag Register 0 (PWM_INTSTS0)

Register	Offset	R/W	Description	Reset Value
PWM_INTSTS0	PWMx_BA+0xE8	R/W	PWM Interrupt Flag Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		CMPDIF5	CMPDIF4	CMPDIF3	CMPDIF2	CMPDIF1	CMPDIF0
23	22	21	20	19	18	17	16
Reserved		CMPUIF5	CMPUIF4	CMPUIF3	CMPUIF2	CMPUIF1	CMPUIF0
15	14	13	12	11	10	9	8
Reserved			PIF4	Reserved	PIF2	Reserved	PIF0
7	6	5	4	3	2	1	0
Reserved			ZIF4	Reserved	ZIF2	Reserved	ZIF0

Bits	Description	
[31:30]	**Reserved**	Reserved.
[24+n] n=0,1..5	**CMPDIFn**	**PWM Compare Down Count Interrupt Flag** Flag is set by hardware when PWM counter down count and reaches PWM_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.
[23:22]	**Reserved**	Reserved.
[21:16]	**CMPUIFn**	**PWM Compare Up Count Interrupt Flag** Flag is set by hardware when PWM counter up count and reaches PWM_CMPDATn, software can clear this bit by writing 1 to it. **Note:** In complementary mode, CMPUIF1, 3, 5 is used as another CMPUIF for channel 0, 2, 4.
[15:13]	**Reserved**	Reserved.
[12]	**PIF4**	**PWM Period Point Interrupt Flag 4** This bit is set by hardware when PWM_CH4 counter reaches PWM_PERIOD4. **Note:** This bit can be cleared to 0 by software writing 1.
[11]	**Reserved**	Reserved.
[10]	**PIF2**	**PWM Period Point Interrupt Flag 2** This bit is set by hardware when PWM_CH2 counter reaches PWM_PERIOD2. **Note:** This bit can be cleared to 0 by software writing 1.
[9]	**Reserved**	Reserved.
[8]	**PIF0**	**PWM Period Point Interrupt Flag 0** This bit is set by hardware when PWM_CH0 counter reaches PWM_PERIOD0. **Note:** This bit can be cleared to 0 by software writing 1.
[7:5]	**Reserved**	Reserved.

[4]	**ZIF4**	**PWM Zero Point Interrupt Flag 4** This bit is set by hardware when PWM_CH4 counter reaches 0. **Note:** This bit can be cleared to 0 by software writing 1.
[3]	**Reserved**	Reserved.
[2]	**ZIF2**	**PWM Zero Point Interrupt Flag 2** This bit is set by hardware when PWM_CH2 counter reaches 0. **Note:** This bit can be cleared to 0 by software writing 1.
[1]	**Reserved**	Reserved.
[0]	**ZIF0**	**PWM Zero Point Interrupt Flag 0** This bit is set by hardware when PWM_CH0 counter reaches 0. **Note:** This bit can be cleared to 0 by software writing 1.

PWM Interrupt Flag Register 1 (PWM_INTSTS1)

Register	Offset	R/W	Description	Reset Value
PWM_INTSTS1	PWMx_BA+0xEC	R/W	PWM Interrupt Flag Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		BRKLSTS5	BRKLSTS4	BRKLSTS3	BRKLSTS2	BRKLSTS1	BRKLSTS0
23	22	21	20	19	18	17	16
Reserved		BRKESTS5	BRKESTS4	BRKESTS3	BRKESTS2	BRKESTS1	BRKESTS0
15	14	13	12	11	10	9	8
Reserved		BRKLIF5	BRKLIF4	BRKLIF3	BRKLIF2	BRKLIF1	BRKLIF0
7	6	5	4	3	2	1	0
Reserved		BRKEIF5	BRKEIF4	BRKEIF3	BRKEIF2	BRKEIF1	BRKEIF0

Bits	Description	
[31:30]	**Reserved**	Reserved.
[24+n] n=0,1..5	**BRKLSTSn**	**PWM Channel n Level-detect Brake Status (Read Only)** 0 = PWM channel n level-detect brake state is released. 1 = When PWM channel n level-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel n at brake state. **Note:** This bit is read only and auto cleared by hardware. When enabled brake source return to high level, PWM will release brake state until current PWM period finished. The PWM waveform will start output from next full PWM period.
[23:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**BRKESTSn**	**PWM Channel n Edge-detect Brake Status (Read Only)** 0 = PWM channel n edge-detect brake state is released. 1 = When PWM channel n edge-detect brake detects a falling edge of any enabled brake source; this flag will be set to indicate the PWM channel n at brake state. **Note:** This bit is read only and auto cleared by hardware. When edge-detect brake interrupt flag is cleared, PWM will release brake state until current PWM period finished. The PWM waveform will start output from next full PWM period.
[15:14]	**Reserved**	Reserved.
[8+n] n=0,1..5	**BRKLIFn**	**PWM Channel n Level-detect Brake Interrupt Flag (Write Protect)** 0 = PWM channel n level-detect brake event do not happened. 1 = When PWM 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**	**PWM Channel n Edge-detect Brake Interrupt Flag (Write Protect)** 0 = PWM channel n edge-detect brake event do not happened. 1 = When PWM 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.

PWM Trigger ADC Source Select Register 0 (PWM_ADCTS0)

Register	Offset	R/W	Description	Reset Value
PWM_ADCTS0	PWMx_BA+0xF8	R/W	PWM Trigger ADC Source Select Register 0	0x0000_0000

31	30	29	28	27	26	25	24
TRGEN3	Reserved			TRGSEL3			
23	22	21	20	19	18	17	16
TRGEN2	Reserved			TRGSEL2			
15	14	13	12	11	10	9	8
TRGEN1	Reserved			TRGSEL1			
7	6	5	4	3	2	1	0
TRGEN0	Reserved			TRGSEL0			

Bits	Description	
[31]	**TRGEN3**	**PWM_CH3 Trigger ADC Enable Bit** 0 = PWM_CH3 Trigger ADC function Disabled. 1 = PWM_CH3 Trigger ADC function Enabled.
[30:28]	**Reserved**	Reserved.
[27:24]	**TRGSEL3**	**PWM_CH3 Trigger ADC Source Select** 0000 = PWM_CH2 zero point. 0001 = PWM_CH2 period point. 0010 = PWM_CH2 zero or period point. 0011 = PWM_CH2 up-count CMPDAT point. 0100 = PWM_CH2 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = PWM_CH3 up-count CMPDAT point. 1001 = PWM_CH3 down-count CMPDAT point. Others = reserved.
[23]	**TRGEN2**	**PWM_CH2 Trigger ADC Enable Bit** 0 = PWM_CH2 Trigger ADC function Disabled. 1 = PWM_CH2 Trigger ADC function Enabled.
[22:20]	**Reserved**	Reserved.
[19:16]	**TRGSEL2**	**PWM_CH2 Trigger ADC Source Select** 0000 = PWM_CH2 zero point. 0001 = PWM_CH2 period point. 0010 = PWM_CH2 zero or period point. 0011 = PWM_CH2 up-count CMPDAT point. 0100 = PWM_CH2 down-count CMPDAT point.

		0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = PWM_CH3 up-count CMPDAT point. 1001 = PWM_CH3 down-count CMPDAT point. Others = reserved.
[15]	**TRGEN1**	**PWM_CH1 Trigger ADC Enable Bit** 0 = PWM_CH1 Trigger ADC function Disabled. 1 = PWM_CH1 Trigger ADC function Enabled.
[14:12]	**Reserved**	Reserved.
[11:8]	**TRGSEL1**	**PWM_CH1 Trigger ADC Source Select** 0000 = PWM_CH0 zero point. 0001 = PWM_CH0 period point. 0010 = PWM_CH0 zero or period point. 0011 = PWM_CH0 up-count CMPDAT point. 0100 = PWM_CH0 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = PWM_CH1 up-count CMPDAT point. 1001 = PWM_CH1 down-count CMPDAT point. Others = reserved.
[7]	**TRGEN0**	**PWM_CH0 Trigger ADC Enable Bit** 0 = PWM_CH0 Trigger ADC function Disabled. 1 = PWM_CH0 Trigger ADC function Enabled.
[6:4]	**Reserved**	Reserved.
[3:0]	**TRGSEL0**	**PWM_CH0 Trigger ADC Source Select** 0000 = PWM_CH0 zero point. 0001 = PWM_CH0 period point. 0010 = PWM_CH0 zero or period point. 0011 = PWM_CH0 up-count CMPDAT point. 0100 = PWM_CH0 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = PWM_CH1 up-count CMPDAT point. 1001 = PWM_CH1 down-count CMPDAT point. Others = reserved.

PWM Trigger ADC Source Select Register 1 (PWM_ADCTS1)

Register	Offset	R/W	Description	Reset Value
PWM_ADCTS1	PWMx_BA+0xFC	R/W	PWM Trigger ADC Source Select Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
TRGEN5	Reserved			TRGSEL5			
7	6	5	4	3	2	1	0
TRGEN4	Reserved			TRGSEL4			

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**TRGEN5**	**PWM_CH5 Trigger ADC Enable Bit** 0 = PWM_CH5 Trigger ADC function Disabled. 1 = PWM_CH5 Trigger ADC function Enabled.
[14:12]	**Reserved**	Reserved.
[11:8]	**TRGSEL5**	**PWM_CH5 Trigger ADC Source Select** 0000 = PWM_CH4 zero point. 0001 = PWM_CH4 period point. 0010 = PWM_CH4 zero or period point. 0011 = PWM_CH4 up-count CMPDAT point. 0100 = PWM_CH4 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = PWM_CH5 up-count CMPDAT point. 1001 = PWM_CH5 down-count CMPDAT point. Others = reserved.
[7]	**TRGEN4**	**PWM_CH4 Trigger ADC Enable Bit** 0 = PWM_CH4 Trigger ADC function Disabled. 1 = PWM_CH4 Trigger ADC function Enabled.
[6:4]	**Reserved**	Reserved.
[3:0]	**TRGSEL4**	**PWM_CH4 Trigger ADC Source Select** 0000 = PWM_CH4 zero point. 0001 = PWM_CH4 period point. 0010 = PWM_CH4 zero or period point.

		0011 = PWM_CH4 up-count CMPDAT point. 0100 = PWM_CH4 down-count CMPDAT point. 0101 = Reserved. 0110 = Reserved. 0111 = Reserved. 1000 = PWM_CH5 up-count CMPDAT point. 1001 = PWM_CH5 down-count CMPDAT point. Others = reserved.

PWM Synchronous Start Control Register (PWM_SSCTL)

Register	Offset	R/W	Description	Reset Value
PWM_SSCTL	PWMx_BA+0x110	R/W	PWM Synchronous Start Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**SSRC**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**SSEN4**	**Reserved**	**SSEN2**	**Reserved**	**SSEN0**

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:8]	**SSRC**	**PWM Synchronous Start Source Select Bits** 00 = Synchronous start source come from PWM0. 01 = Synchronous start source come from PWM1. 10 = Reserved. 11 = Reserved.
[7:5]	**Reserved**	Reserved.
[4]	**SSEN4**	**PWM Synchronous Start Function Enable Bit 4** When synchronous start function is enabled, the PWM_CH4 counter enable bit (CNTEN4) can be enabled by writing PWM synchronous start trigger bit (CNTSEN). 0 = PWM synchronous start function Disabled. 1 = PWM synchronous start function Enabled.
[3]	**Reserved**	Reserved.
[2]	**SSEN2**	**PWM Synchronous Start Function Enable Bit 2** When synchronous start function is enabled, the PWM_CH2 counter enable bit (CNTEN2) can be enabled by writing PWM synchronous start trigger bit (CNTSEN). 0 = PWM synchronous start function Disabled. 1 = PWM synchronous start function Enabled.
[1]	**Reserved**	Reserved.
[0]	**SSEN0**	**PWM Synchronous Start Function Enable Bit 0** When synchronous start function is enabled, the PWM_CH0 counter enable bit (CNTEN0) can be enabled by writing PWM synchronous start trigger bit (CNTSEN). 0 = PWM synchronous start function Disabled. 1 = PWM synchronous start function Enabled.

PWM Synchronous Start Trigger Register (PWM_SSTRG)

Register	Offset	R/W	Description	Reset Value
PWM_SSTRG	PWMx_BA+0x114	W	PWM Synchronous Start Trigger Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**CNTSEN**	**PWM Counter Synchronous Start Enable (Write Only)** PWM counter synchronous enable function is used to make selected PWM channels (include PWM0_CHx and PWM1_CHx) start counting at the same time. Writing this bit to 1 will also set the counter enable bit (CNTENn, n denotes channel 0 to 5) if correlated PWM channel counter synchronous start function is enabled.

PWM Status Register (PWM_STATUS)

Register	Offset	R/W	Description	Reset Value
PWM_STATUS	PWMx_BA+0x120	R/W	PWM Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved		ADCTRG5	ADCTRG4	ADCTRG3	ADCTRG2	ADCTRG1	ADCTRG0
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			CNTMAX4	Reserved	CNTMAX2	Reserved	CNTMAX0

Bits	Description	
[31:22]	**Reserved**	Reserved.
[16+n] n=0,1..5	**ADCTRGn**	**ADC Start of Conversion Status** 0 = Indicates no ADC start of conversion trigger event has occurred. 1 = An ADC start of conversion trigger event has occurred. **Note:** This bit can be cleared by software writing 1.
[15:5]	**Reserved**	Reserved.
[4]	**CNTMAX4**	**Time-base Counter 4 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.
[3]	**Reserved**	Reserved.
[2]	**CNTMAX2**	**Time-base Counter 2 Equal to 0xFFFF Latched Flag** 0 = indicates the time-base counter never reached its maximum value 0xFFFF. 1 = indicates the time-base counter reached its maximum value. **Note:** This bit can be cleared by software writing 1.
[1]	**Reserved**	Reserved.
[0]	**CNTMAX0**	**Time-base Counter 0 Equal to 0xFFFF Latched Flag** 0 = indicates the time-base counter never reached its maximum value 0xFFFF. 1 = indicates the time-base counter reached its maximum value. **Note:** This bit can be cleared by software writing 1.

PWM Capture Input Enable Register (PWM_CAPINEN)

Register	Offset	R/W	Description	Reset Value
PWM_CAPINEN	PWMx_BA+0x200	R/W	PWM Capture Input Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved		CAPINEN5	CAPINEN4	CAPINEN3	CAPINEN2	CAPINEN1	CAPINEN0

Bits	Description	
[31:6]	**Reserved**	Reserved.
[n] n=0,1..5	**CAPINENn**	**Capture Input Enable Bits** 0 = PWM Channel capture input path Disabled. The input of PWM channel capture function is always regarded as 0. 1 = PWM Channel capture input path Enabled. The input of PWM channel capture function comes from correlative multifunction pin.

PWM Capture Control Register (PWM_CAPCTL)

Register	Offset	R/W	Description	Reset Value
PWM_CAPCTL	PWMx_BA+0x204	R/W	PWM Capture Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		FCRLDEN5	FCRLDEN4	FCRLDEN3	FCRLDEN2	FCRLDEN1	FCRLDEN0
23	22	21	20	19	18	17	16
Reserved		RCRLDEN5	RCRLDEN4	RCRLDEN3	RCRLDEN2	RCRLDEN1	RCRLDEN0
15	14	13	12	11	10	9	8
Reserved		CAPINV5	CAPINV4	CAPINV3	CAPINV2	CAPINV1	CAPINV0
7	6	5	4	3	2	1	0
Reserved		CAPEN5	CAPEN4	CAPEN3	CAPEN2	CAPEN1	CAPEN0

Bits	Description	
[31:30]	**Reserved**	Reserved.
[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. RCAPDAT/FCAPDAT register will not be updated. 1 = Capture function Enabled. Capture latched the PWM counter value when detected rising or falling edge of input signal and saved to RCAPDAT (Rising latch) and FCAPDAT (Falling latch).

PWM Capture Status Register (PWM_CAPSTS)

Register	Offset	R/W	Description	Reset Value
PWM_CAPSTS	PWMx_BA+0x208	R	PWM Capture Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		CFLIFOV5	CFLIFOV4	CFLIFOV3	CFLIFOV2	CFLIFOV1	CFLIFOV0
7	6	5	4	3	2	1	0
Reserved		CRLIFOV5	CRLIFOV4	CRLIFOV3	CRLIFOV2	CRLIFOV1	CRLIFOV0

Bits	Description	
[31:14]	**Reserved**	Reserved.
[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 CFLIF is 1. **Note:** This bit will be cleared automatically when user clear corresponding CFLIF.
[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 CRLIF is 1. **Note:** This bit will be cleared automatically when user clear corresponding CRLIF.

PWM Rising Capture Data Register 0~5 (PWM_RCAPDAT 0~5)

Register	Offset	R/W	Description	Reset Value
PWM_RCAPDAT0	PWMx_BA+0x20C	R	PWM Rising Capture Data Register 0	0x0000_0000
PWM_RCAPDAT1	PWMx_BA+0x214	R	PWM Rising Capture Data Register 1	0x0000_0000
PWM_RCAPDAT2	PWMx_BA+0x21C	R	PWM Rising Capture Data Register 2	0x0000_0000
PWM_RCAPDAT3	PWMx_BA+0x224	R	PWM Rising Capture Data Register 3	0x0000_0000
PWM_RCAPDAT4	PWMx_BA+0x22C	R	PWM Rising Capture Data Register 4	0x0000_0000
PWM_RCAPDAT5	PWMx_BA+0x234	R	PWM Rising Capture Data Register 5	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**RCAPDAT**	**PWM Rising Capture Data Register (Read Only)** When rising capture condition happened, the PWM counter value will be saved in this register.

PWM Falling Capture Data Register 0~5 (PWM_FCAPDAT 0~5)

Register	Offset	R/W	Description	Reset Value
PWM_FCAPDAT0	PWMx_BA+0x210	R	PWM Falling Capture Data Register 0	0x0000_0000
PWM_FCAPDAT1	PWMx_BA+0x218	R	PWM Falling Capture Data Register 1	0x0000_0000
PWM_FCAPDAT2	PWMx_BA+0x220	R	PWM Falling Capture Data Register 2	0x0000_0000
PWM_FCAPDAT3	PWMx_BA+0x228	R	PWM Falling Capture Data Register 3	0x0000_0000
PWM_FCAPDAT4	PWMx_BA+0x230	R	PWM Falling Capture Data Register 4	0x0000_0000
PWM_FCAPDAT5	PWMx_BA+0x238	R	PWM Falling Capture Data Register 5	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**FCAPDAT**	**PWM Falling Capture Data Register (Read Only)** When falling capture condition happened, the PWM counter value will be saved in this register.

PWM PDMA Control Register (PWM_PDMACTL)

Register	Offset	R/W	Description	Reset Value
PWM_PDMACTL	PWMx_BA+0x23C	R/W	PWM PDMA Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			CHSEL4_5	CAPORD4_5	CAPMOD4_5		CHEN4_5
15	14	13	12	11	10	9	8
Reserved			CHSEL2_3	CAPORD2_3	CAPMOD2_3		CHEN2_3
7	6	5	4	3	2	1	0
Reserved			CHSEL0_1	CAPORD0_1	CAPMOD0_1		CHEN0_1

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20]	**CHSEL4_5**	**Select Channel 4/5 to Do PDMA Transfer** 0 = Channel4. 1 = Channel5. **Note:** If the PDMA function is not supported, this bit field will become invalid. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[19]	**CAPORD4_5**	**Capture Channel 4/5 Rising/Falling Order** Set this bit to determine whether the PWM_RCAPDAT4/5 or PWM_FCAPDAT4/5 is the first captured data transferred to memory through PDMA when CAPMOD4_5 =11. 0 = PWM_FCAPDAT4/5 is the first captured data to memory. 1 = PWM_RCAPDAT4/5 is the first captured data to memory. **Note:** If the PDMA function is not supported, this bit field will become invalid. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[18:17]	**CAPMOD4_5**	**Select PWM_RCAPDAT4/5 or PWM_FCAPDAT4/5 to Do PDMA Transfer** 00 = Reserved. 01 = PWM_RCAPDAT4/5. 10 = PWM_FCAPDAT4/5. 11 = Both PWM_RCAPDAT4/5 and PWM_FCAPDAT4/5. **Note:** If the PDMA function is not supported, this bit field will become invalid. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[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. **Note:** If the PDMA function is not supported, this bit field will become invalid. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed

		information.
[15:13]	**Reserved**	Reserved.
[12]	**CHSEL2_3**	**Select Channel 2/3 to Do PDMA Transfer** 0 = Channel2. 1 = Channel3. **Note:** If the PDMA function is not supported, this bit field will become invalid. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[11]	**CAPORD2_3**	**Capture Channel 2/3 Rising/Falling Order** Set this bit to determine whether the PWM_RCAPDAT2/3 or PWM_FCAPDAT2/3 is the first captured data transferred to memory through PDMA when CAPMOD2_3 =11. 0 = PWM_FCAPDAT2/3 is the first captured data to memory. 1 = PWM_RCAPDAT2/3 is the first captured data to memory. **Note:** If the PDMA function is not supported, this bit field will become invalid. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[10:9]	**CAPMOD2_3**	**Select PWM_RCAPDAT2/3 or PWM_FCAODAT2/3 to Do PDMA Transfer** 00 = Reserved. 01 = PWM_RCAPDAT2/3. 10 = PWM_FCAPDAT2/3. 11 = Both PWM_RCAPDAT2/3 and PWM_FCAPDAT2/3. **Note:** If the PDMA function is not supported, this bit field will become invalid. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[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. **Note:** If the PDMA function is not supported, this bit field will become invalid. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[7:5]	**Reserved**	Reserved.
[4]	**CHSEL0_1**	**Select Channel 0/1 to Do PDMA Transfer** 0 = Channel0. 1 = Channel1. **Note:** If the PDMA function is not supported, this bit field will become invalid. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[3]	**CAPORD0_1**	**Capture Channel 0/1 Rising/Falling Order** Set this bit to determine whether the PWM_RCAPDAT0/1 or PWM_FCAPDAT0/1 is the first captured data transferred to memory through PDMA when CAPMOD0_1 =11. 0 = PWM_FCAPDAT0/1 is the first captured data to memory. 1 = PWM_RCAPDAT0/1 is the first captured data to memory. **Note:** If the PDMA function is not supported, this bit field will become invalid. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[2:1]	**CAPMOD0_1**	**Select PWM_RCAPDAT0/1 or PWM_FCAPDAT0/1 to Do PDMA Transfer** 00 = Reserved. 01 = PWM_RCAPDAT0/1.

		10 = PWM_FCAPDAT0/1. 11 = Both PWM_RCAPDAT0/1 and PWM_FCAPDAT0/1. **Note:** If the PDMA function is not supported, this bit field will become invalid. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.
[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. **Note:** If the PDMA function is not supported, this bit field will become invalid. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.

PWM Capture Channel 0_1, 2_3, 4_5 PDMA Register (PWM_PDMACAP 0_1, 2_3, 4_5)

Register	Offset	R/W	Description	Reset Value
PWM_PDMACAP0_1	PWMx_BA+0x240	R	PWM Capture Channel 01 PDMA Register	0x0000_0000
PWM_PDMACAP2_3	PWMx_BA+0x244	R	PWM Capture Channel 23 PDMA Register	0x0000_0000
PWM_PDMACAP4_5	PWMx_BA+0x248	R	PWM Capture Channel 45 PDMA Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CAPBUF**	**PWM Capture PDMA Register (Read Only)** This register is used as a buffer to transfer PWM capture rising or falling data to memory by PDMA. **Note:** If the PDMA function is not supported, this bit field will become invalid. Please refer to section 3.2 NuMicro® M031/M032 Series Selection Guide for detailed information.

PWM Capture Interrupt Enable Register (PWM_CAPIEN)

Register	Offset	R/W	Description	Reset Value
PWM_CAPIEN	PWMx_BA+0x250	R/W	PWM Capture Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		CAPFIEN5	CAPFIEN4	CAPFIEN3	CAPFIEN2	CAPFIEN1	CAPFIEN0
7	6	5	4	3	2	1	0
Reserved		CAPRIEN5	CAPRIEN4	CAPRIEN3	CAPRIEN2	CAPRIEN1	CAPRIEN0

Bits	Description	
[31:14]	**Reserved**	Reserved.
[8+n] n=0,1..5	**CAPFIENn**	**PWM 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**	**PWM Capture Rising Latch Interrupt Enable Bits** 0 = Capture rising edge latch interrupt Disabled. 1 = Capture rising edge latch interrupt Enabled.

PWM Capture Interrupt Flag Register (PWM_CAPIF)

Register	Offset	R/W	Description	Reset Value
PWM_CAPIF	PWMx_BA+0x254	R/W	PWM Capture Interrupt Flag Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		CFLIF5	CFLIF4	CFLIF3	CFLIF2	CFLIF1	CFLIF0
7	6	5	4	3	2	1	0
Reserved		CRLIF5	CRLIF4	CRLIF3	CRLIF2	CRLIF1	CRLIF0

Bits	Description	
[31:14]	**Reserved**	Reserved.
[8+n] n=0,1..5	**CFLIFn**	**PWM 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, CAPIF corresponding channel CFLIF 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**	**PWM 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, CAPIF corresponding channel CRLIF will be cleared by hardware after PDMA transfer data. **Note 2:** This bit is cleared by writing 1 to it.

PWM Period Register Buffer 0, 2, 4 (PWM_PBUF0, 2, 4)

Register	Offset	R/W	Description	Reset Value
PWM_PBUF0	PWMx_BA+0x304	R	PWM PERIOD0 Buffer	0x0000_0000
PWM_PBUF2	PWMx_BA+0x30C	R	PWM PERIOD2 Buffer	0x0000_0000
PWM_PBUF4	PWMx_BA+0x314	R	PWM PERIOD4 Buffer	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PBUF**	**PWM Period Register Buffer (Read Only)** Used as PERIOD active register.

PWM Comparator Register Buffer 0~5 (PWM_CMPBUF0~5)

Register	Offset	R/W	Description	Reset Value
PWM_CMPBUF0	PWMx_BA+0x31C	R	PWM CMPDAT0 Buffer	0x0000_0000
PWM_CMPBUF1	PWMx_BA+0x320	R	PWM CMPDAT1 Buffer	0x0000_0000
PWM_CMPBUF2	PWMx_BA+0x324	R	PWM CMPDAT2 Buffer	0x0000_0000
PWM_CMPBUF3	PWMx_BA+0x328	R	PWM CMPDAT3 Buffer	0x0000_0000
PWM_CMPBUF4	PWMx_BA+0x32C	R	PWM CMPDAT4 Buffer	0x0000_0000
PWM_CMPBUF5	PWMx_BA+0x330	R	PWM CMPDAT5 Buffer	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CMPBUF**	**PWM Comparator Register Buffer (Read Only)** Used as CMP active register.

6.13 UART Interface Controller (UART)

6.13.1 Overview

The chip provides eight channels of Universal Asynchronous Receiver/Transmitters (UART). The UART controller performs Normal Speed UART and supports flow control function. The UART controller performs a serial-to-parallel conversion on data received from the peripheral and a parallel-to-serial conversion on data transmitted from the CPU. Each UART controller channel supports ten types of interrupts. The UART controller also supports IrDA SIR, RS-485 and Single-wire function modes and auto-baud rate measuring function.

6.13.2 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 (Only UART0 /UART1 /UART4 /UART5 with 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
  - Support 9600 bps for UART_CLK is selected LXT. (Only UART0 /UART1 /UART4 /UART5 with this feature)
- Supports break error, frame error, parity error and receive/transmit buffer overflow detection function
- Fully programmable serial-interface characteristics
  - Programmable number of data bit, 5-, 6-, 7-, 8- bit character
  - Programmable parity bit, even, odd, no parity or stick parity bit generation and detection
  - Programmable stop bit, 1, 1.5, or 2 stop bit generation
- Supports IrDA SIR function mode
  - Supports for 3/16 bit duration for normal mode
- Supports RS-485 function mode
  - Supports RS-485 9-bit mode
  - Supports hardware or software enables to program nRTS pin to control RS-485 transmission direction
- Supports PDMA transfer function
- Support Single-wire function mode.

UART Feature	UART0/ UART1/ UART4/ UART5	UART2/ UART3/ UART6/ UART7	USCI-UART
FIFO	16 Bytes	1 Bytes	TX: 1byte

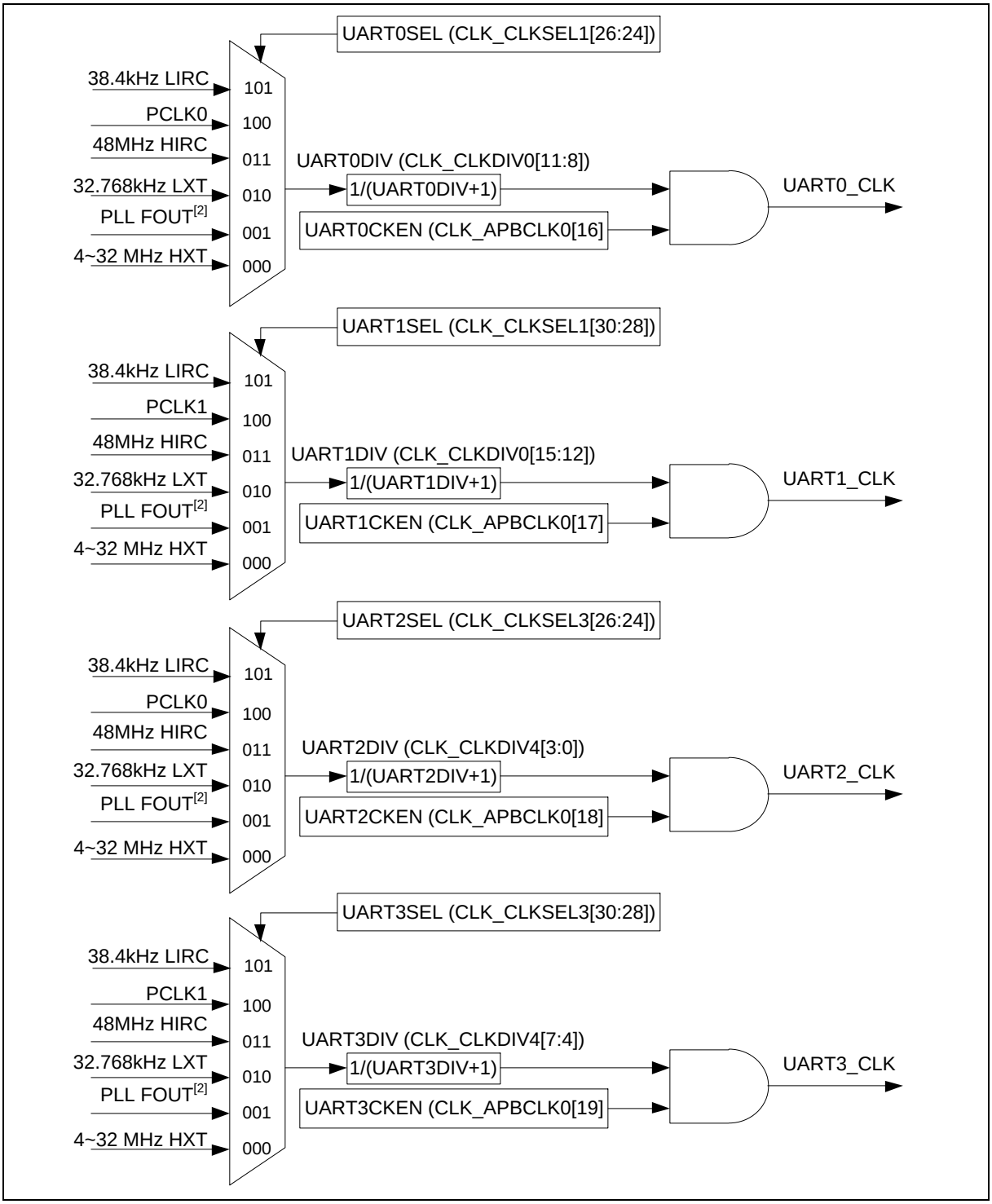
			RX: 2byte
Auto Flow Control (CTS/RTS)	√	√	√
IrDA	√	√	-
LIN	-	-	-
RS-485 Function Mode	√	√	√
nCTS Wake-up	√	√	√
Incoming Data Wake-up	√	√	√
Received Data FIFO reached threshold Wake-up	√	-	-
RS-485 Address Match (AAD mode) Wake-up	√	-	-
Auto-Baud Rate Measurement	√	√	√
STOP Bit Length	1, 1.5, 2 bit	1, 1.5, 2 bit	1, 2 bit
Word Length	5, 6, 7, 8 bits	5, 6, 7, 8 bits	6~13 bits
Even / Odd Parity	√	√	√
Stick Bit	√	√	-
Note: √= Supported			

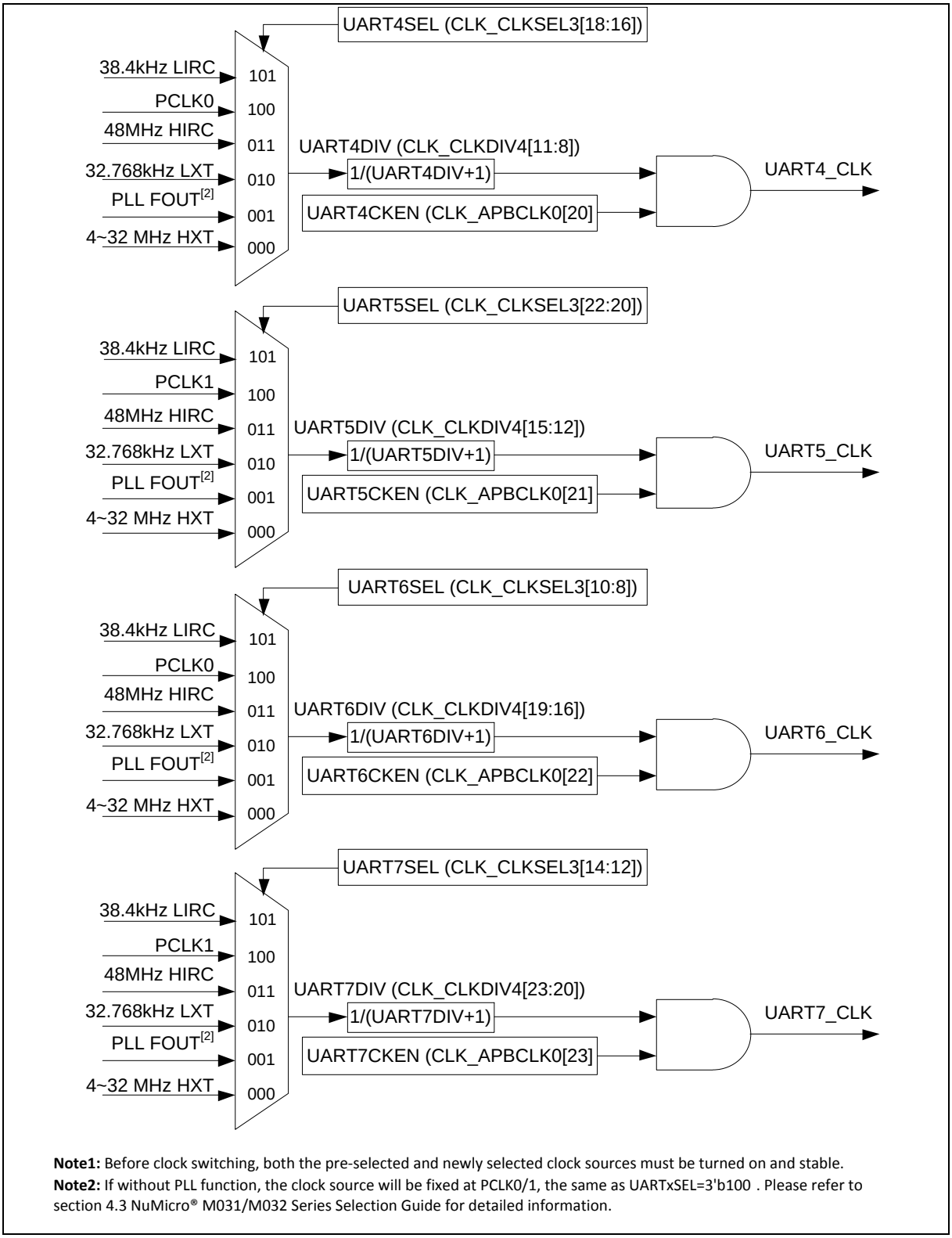
Table 6.13-1 NuMicro® M031/M032 Series UART Features

6.13.3 Block Diagram

The UART clock control and block diagram are shown in Figure 6.13-1 and Figure 6.13-2 respectively.

Note: The frequency of UARTx_CLK should not be greater than 30 times HCLK.


UART0SEL (CLK_CLKSEL1[26:24])
38.4kHz LIRC
PCLK0
48MHz HIRC
32.768kHz LXT
PLL FOUT[2]
4~32 MHz HXT
101
100
011
010
001
000
UART0DIV (CLK_CLKDIV0[11:8])
1/(UART0DIV+1)
UART0CKEN (CLK_APBCLK0[16]
UART0_CLK
UART1SEL (CLK_CLKSEL1[30:28])
38.4kHz LIRC
PCLK1
48MHz HIRC
32.768kHz LXT
PLL FOUT[2]
4~32 MHz HXT
101
100
011
010
001
000
UART1DIV (CLK_CLKDIV0[15:12])
1/(UART1DIV+1)
UART1CKEN (CLK_APBCLK0[17]
UART1_CLK
UART2SEL (CLK_CLKSEL3[26:24])
38.4kHz LIRC
PCLK0
48MHz HIRC
32.768kHz LXT
PLL FOUT[2]
4~32 MHz HXT
101
100
011
010
001
000
UART2DIV (CLK_CLKDIV4[3:0])
1/(UART2DIV+1)
UART2CKEN (CLK_APBCLK0[18]
UART2_CLK
UART3SEL (CLK_CLKSEL3[30:28])
38.4kHz LIRC
PCLK1
48MHz HIRC
32.768kHz LXT
PLL FOUT[2]
4~32 MHz HXT
101
100
011
010
001
000
UART3DIV (CLK_CLKDIV4[7:4])
1/(UART3DIV+1)
UART3CKEN (CLK_APBCLK0[19]
UART3_CLK



Note1: Before clock switching, both the pre-selected and newly selected clock sources must be turned on and stable.
Note2: If without PLL function, the clock source will be fixed at PCLK0/1, the same as UARTxSEL=3'b100 . Please refer to section 4.3 NuMicro® M031/M032 Series Selection Guide for detailed information.

Figure 6.13-1 UART Clock Control Diagram

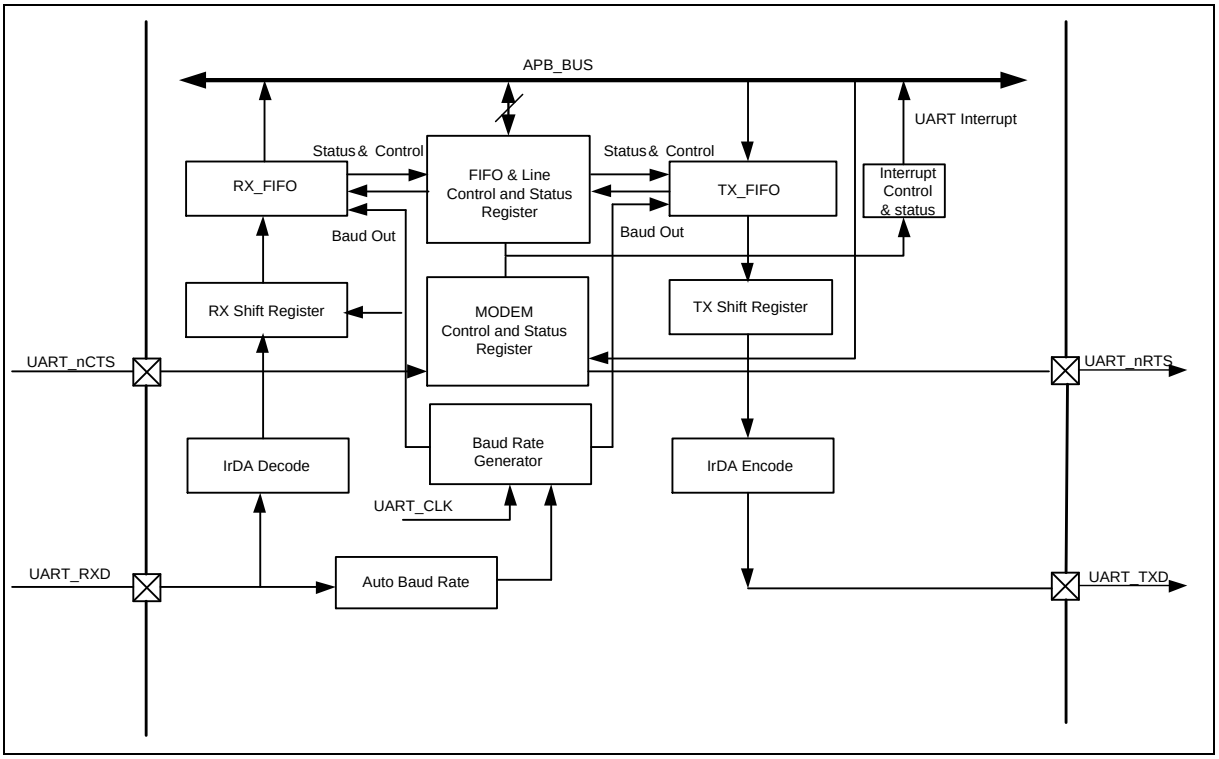


Figure 6.13-2 UART Block Diagram

Each block is described in detail as follows:

TX_FIFO

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

RX_FIFO

The receiver is buffered with a 16 bytes FIFO (plus three error bits, BIF (UART_FIFOSTS[6]), FEF (UART_FIFOSTS[5]), PEF (UART_FIFOSTS[4])) to reduce the number of interrupts presented to the CPU.

TX Shift Register

This block is responsible for shifting out the transmitting data serially.

RX Shift Register

This block is responsible for shifting in the receiving data serially.

Modem Control and Status Register

This register controls the interface to the MODEM or data set (or a peripheral device emulating a MODEM).

Baud Rate Generator

Divide the external clock by the divisor to get the desired baud rate clock. Refer to baud rate equation.

IrDA Encode

This block is IrDA encoding control block.

IrDA Decode

This block is IrDA decoding control block.

FIFO & Line Control and Status Register

This field is register set that including the FIFO control register (UART_FIFO), FIFO status register

(UART_FIFOSTS), and line control register (UART_LINE) for transmitter and receiver. The time-out register (UART_TOUT) identifies the condition of time-out interrupt.

Auto-Baud Rate Measurement

This block is responsible for auto-baud rate measurement.

Interrupt Control and Status Register

There are ten types of interrupts, Receive Data Available Interrupt (RDAINT), Transmit Holding Register Empty Interrupt (THERINT), Transmitter Empty Interrupt (TXENDINT), Receive Line Status Interrupt (parity error or framing error or break interrupt) (RLSINT), MODEM Status Interrupt (MODEMINT), Receiver Buffer Time-out Interrupt (RXTOINT), Buffer Error Interrupt (BUFERRINT), Wake-up Interrupt (WKINT), Auto-Baud Rate Interrupt (ABRINT) and Single-wire Bit Error Detect Interrupt (SWBEINT). 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.
WKINT	Wake-up Interrupt.
ABRINT	Auto-Baud Rate Interrupt.
SWBEINT	Single-wire Bit Error Detect Interrupt.

Table 6.13-2 UART Interrupt

6.13.4 Basic Configuration

The basic configurations of UART0 are as follows:

- Clock Source Configuration
  - Select the source of UART0 peripheral clock on UART0SEL (CLK_CLKSEL1[26:24]).
  - 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]).

The basic configurations of UART1 are as follows:

- Clock Source Configuration
  - Select the source of UART1 peripheral clock on UART1SEL (CLK_CLKSEL1[30:28]).
  - 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]).

The basic configurations of UART2 are as follows:

- Clock Source Configuration
  - Select the source of UART2 peripheral clock on UART2SEL (CLK_CLKSEL3[26:24]).
  - 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]).

The basic configurations of UART3 are as follows:

- Clock Source Configuration
  - Select the source of UART3 peripheral clock on UART3SEL (CLK_CLKSEL3[30:28]).
  - 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]).

The basic configurations of UART4 are as follows:

- Clock Source Configuration
  - Select the source of UART4 peripheral clock on UART4SEL (CLK_CLKSEL3[18:16]).
  - 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]).

The basic configurations of UART5 are as follows:

- Clock Source Configuration
  - Select the source of UART5 peripheral clock on UART5SEL (CLK_CLKSEL3[22:20]).
  - Select 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]).

The basic configurations of UART6 are as follows:

- Clock Source Configuration
  - Select the source of UART6 peripheral clock on UART6SEL (CLK_CLKSEL3[10:8]).
  - Select the clock divider number of UART6 peripheral clock on UART6DIV (CLK_CLKDIV4[19:16]).
  - Enable UART6 peripheral clock in UART6CKEN (CLK_APBCLK0[22]).
- Reset UART6 controller in UART6RST (SYS_IPRST1[22]).

The basic configurations of UART7 are as follows:

- Clock Source Configuration
  - Select the source of UART7 peripheral clock on UART7SEL (CLK_CLKSEL3[14:12]).
  - Select clock divider number of UART7 peripheral clock on UART7DIV (CLK_CLKDIV4[23:20]).
  - Enable UART7 peripheral clock in UART7CKEN (CLK_APBCLK0[23]).
- Reset UART7 controller in UART7RST (SYS_IPRST1[23]).

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 6.13-3 UART Interface Controller Pin

6.13.5 Functional Description

The UART controller supports four function modes including UART, IrDA and RS-485 mode. User can select a function by setting the UART_FUNCSEL register. The four function modes will be described in following section.

6.13.5.1 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 6.13-4 list the UART baud rate equations in the various conditions. Table 6.13-5 and Table 6.13-6 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 6.13-4 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 =13	BRD=24
230400	Not recommended	BRD =2, EDIVM1 =13	BRD =50
115200	Not recommended	BRD =6, EDIVM1 =13	BRD =102
57600	BRD =11	BRD =14, EDIVM1 =13	BRD =206
38400	BRD =18	BRD =22, EDIVM1 =13	BRD =311
19200	BRD =37	BRD =123, EDIVM1 =5	BRD =623
9600	BRD =76	BRD =123, EDIVM1 =10	BRD =1248
4800	BRD =154	BRD =248, EDIVM1 =10	BRD =2498

Table 6.13-5 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	0x2D00_0000	0x3000_0018
230400	Not recommended	0x2D00_0002	0x3000_0032
115200	Not recommended	0x2D00_0006	0x3000_0066
57600	0x0000_000B	0x2D00_000E	0x3000_00CE
38400	0x0000_0012	0x2D00_0016	0x3000_0137
19200	0x0000_0025	0x2500_007B	0x3000_026F
9600	0x0000_004C	0x2A00_007B	0x3000_04E0
4800	0x0000_009A	0x2A00_00F8	0x3000_09C2

Table 6.13-6 UART controller Baud Rate Register Setting Example Table

6.13.5.2 UART Controller Baud Rate Compensation

The UART controller supports baud rate compensation function. It is used to optimize the precision in each bit. The precision of the compensation is half of UART module clock because there is BRCOMDEC bit (UART_BRCOMP[31]) to define the positive or negative compensation in each bit. If the BRCOMPDEC (UART_BRCOMP[31]) = 0, it is positive compensation for each bit, one more module clock will be append in the compensated bit. If the BRCOMPDEC (UART_BRCOMP[31]) = 1, it is negative compensation for each bit, decrease one module clock in the compensated bit.

There is 9-bits location, BRCOMP[8:0] (UART_BRCOMP[8:0]), can be configured by user to define the relative bit is compensated or not. BRCOMP[7:0] is used to define the compensation of UART_DAT[7:0] and BRCOMP[8] is used to define the parity bit.

Example:

1. UART's peripheral clock = 32.768k and baud rate is 9600

Baud rate is 9600, UART peripheral clock is 32.768k → 3.413 peripheral clock/bit

if the baud divider is set 1 (3 peripheral clock/bit), the inaccuracy of each bit is -0.413 peripheral clock and BRCOMPDEC =0,

Bit	Name	Total INACCURACY	BRCOMP Compensated	Final Inaccuracy
0	Start	-0.413	x	-0.413
1	UART_DAT[0]	-0.826(-0.413-0.413)	1	0.174
2	UART_DAT[1]	-0.239(0.174-0.413)	0	-0.239
3	UART_DAT[2]	-0.652(-0.239-0.413)	1	0.348
4	UART_DAT[3]	-0.065(0.348-0.413)	0	-0.065
5	UART_DAT[4]	-0.478(-0.065-0.413)	0	-0.478
6	UART_DAT[5]	-0.891(-0.478-0.413)	1	0.109
7	UART_DAT[6]	-0.304(0.109-0.413)	0	-0.304
8	UART_DAT[7]	-0.717(-0.304-0.413)	1	0.283
9	Parity	-0.130(0.283-0.413)	0	-0.13

Table 6.13-7 Baud Rate Compensation Example Table 1

So that the BRCOMP (UART_BRCOMP[8:0]) can be set as 9'b010100101 = 0xa5.

2. UART's peripheral clock = 32.768k and baud rate is 4800

Baud rate is 4800, UART peripheral clock is 32.768k → 6.827 peripheral clock/bit

if the baud divider is set 5 (7 peripheral clock/bit), the inaccuracy of each bit is 0.173 peripheral clock and BRCOMPDEC =1,

Bit	Name	Total INACCURACY	BRCOMP Compensated	Final Inaccuracy
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 6.13-8 Baud Rate Compensation Example Table 2

So that the BRCOMP (UART_BRCOMP[8:0]) can be set as 9'b010000010 = 0x82.

UART Controller Auto-Baud Rate Function Mode

Auto-Baud Rate function can measure baud rate of receiving data from UART RX pin automatically. When the Auto-Baud Rate measurement is finished, the measuring baud rate is loaded to BRD (UART_BAUD[15:0]). Both of the BAUDM1 (UART_BAUD[29]) and BAUDM0 (UART_BAUD[28]) are set to 1 automatically. UART RX data from Start bit to 1st rising edge time is set by $2^{ABRDBITS}$ bit time in Auto-Baud Rate function detection frame.

$2^{ABRDBITS}$ bit time from Start bit to the 1st rising edge is calculated by setting ABRDBITS (UART_ALTCTL[20:19]). Setting ABRDEN (UART_ALTCTL[18]) is to enable auto-baud rate function. In beginning stage, the UART RX is kept at 1. Once falling edge is detected, START bit is received. The auto-baud rate counter is reset and starts counting. The auto-baud rate counter will be stop when the 1st rising edge is detected. Then, auto-baud rate counter value divided by ABRDBITS (UART_ALTCTL[20:19]) is loaded to BRD (UART_BAUD[15:0]) automatically. ABRDEN (UART_ALTCTL[18]) is cleared. The Auto-Baud is shown in Figure 6.13-3. 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. ABRDIF (UART_FIFOSTS[1]) or ABRDTOIF (UART_FIFOSTS[2]) cause the auto-baud rate flag ABRIF(UART_ALTCTL[17]) is generated. If the ABRIEN (UART_INTEN[18]) is enabled, ABRIF(UART_ALTCTL[17]) cause the auto-baud rate interrupt ABRINT (UART_INTSTS[31]) is generated.

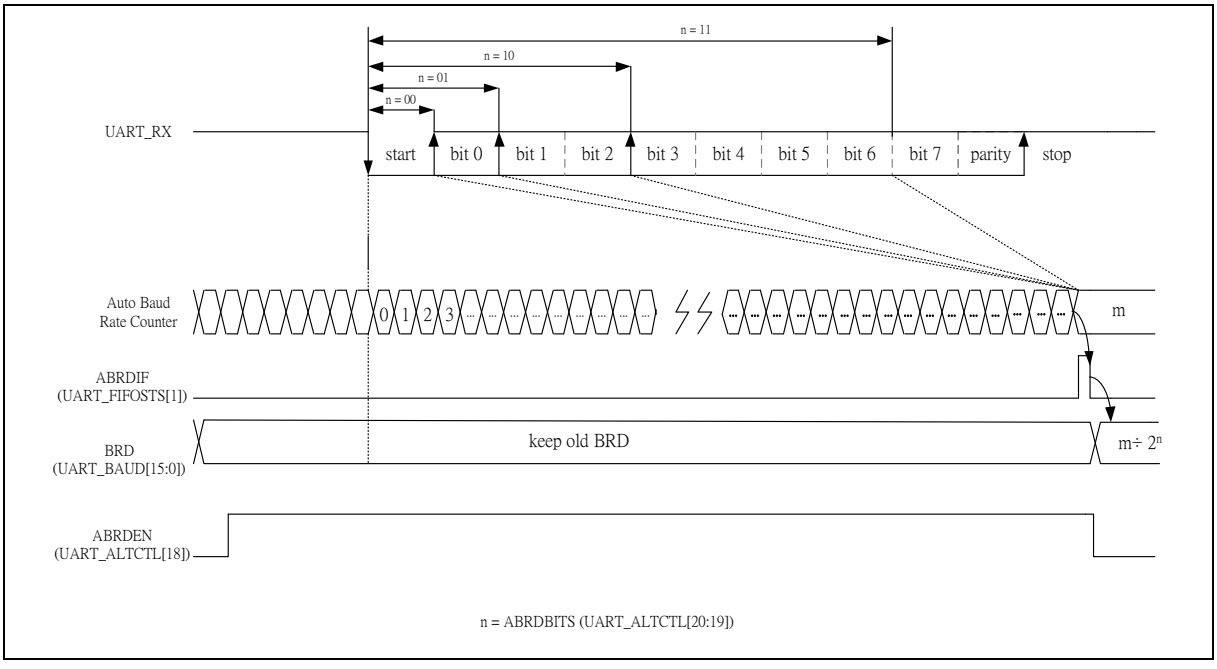


Figure 6.13-3 Auto-Baud Rate Measurement

6.13.5.3 Programming Sequence Example:

1. Program ABRDBITS (UART_ALTCTL[20:19]) to determines 2 ^{ABRDBITS} bit time from UART RX receive START bit falling edge to data 1st rising edge.
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.

6.13.5.4 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 6.13-4.

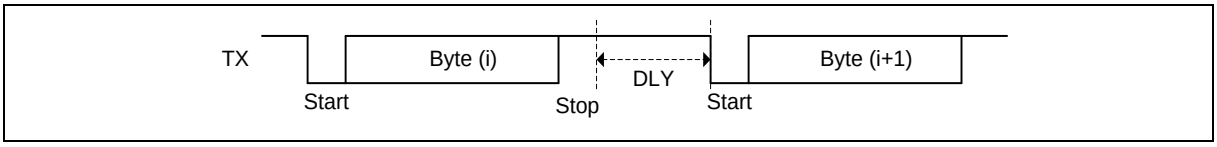


Figure 6.13-4 Transmit Delay Time Operation

6.13.5.5 UART Controller FIFO Control and Status

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

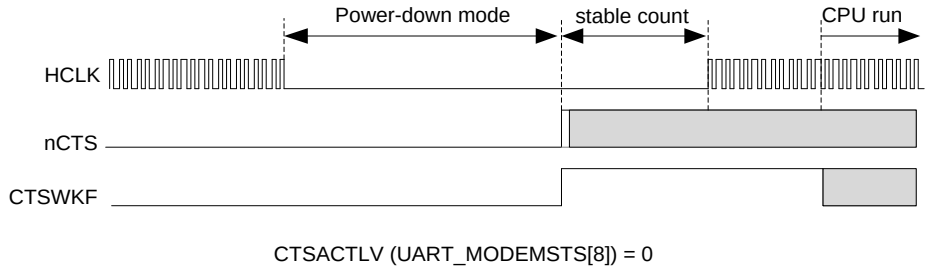
6.13.5.6 UART Controller Wake-up Function

The UART controller supports wake-up system function. The wake-up function includes nCTS pin, incoming data wake-up, Received Data FIFO reached threshold wake-up, RS-485 Address Match (AAD mode) wake-up and Received Data FIFO threshold time-out wake-up function. CTSWKF (UART_WKSTS[0]), DATWKF (UART_WKSTS[1]), RFRTWKF (UART_WKSTS[2]), RS485WKF (UART_WKSTS[3]) or TOUTWKF (UART_WKSTS[4]) cause the wake-up interrupt flag WKIF(UART_INTSTS[6]) is generated. If the WKIEN (UART_INTEN[6]) is enabled, the wake-up interrupt flag WKIF(UART_INTSTS[6]) cause the wake-up interrupt WKINT (UART_INTSTS[14]) is generated.

nCTS pin wake-up :

When the system is in Power-down mode and WKCTSEN (UART_WKCTL[0]) is set, the toggle of nCTS pin can wake-up system. If the WKCTSEN (UART_WKCTL[0]) is enabled, the toggle of nCTS pin cause the nCTS wake-up flag CTSWKF (UART_WKSTS[0]) is generated. The nCTS wake-up is shown in Figure 6.13-5 and Figure 6.13-6.

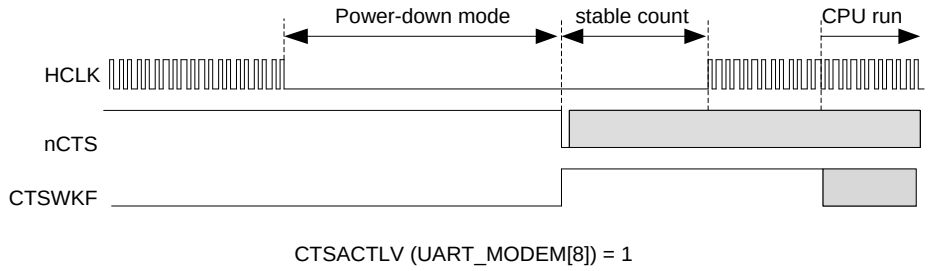
nCTS Wake-up Case 1 (nCTS transition from low to high)



Note: Stable count means HCLK source recovery stable count.

Figure 6.13-5 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 6.13-6 UART nCTS Wake-up Case2

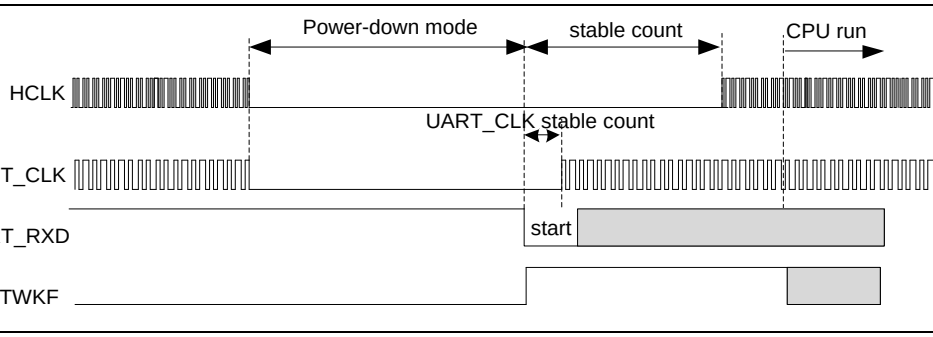
Incoming Data Wake-up

When system is in Power-down mode and the WKDATEN (UART_WKCTL [1]) is set, the toggle of incoming data (UART_RXD) pin can wake-up the system. In order to receive the incoming data after the system wake-up, the STCOMP (UART_DWKCOMP[15:0]) shall be set. These bits field of STCOMP indicate how many clock cycle selected by UART_CLK do the UART controller can get the 1st bit (start bit) when the system is wakeup from Power-down mode.

When incoming data wakes system up, the incoming data will be received and stored in FIFO. If the WKDATEN (UART_WKCTL[1]) is enabled, the toggle of incoming data (UART_RXD) pin cause the incoming data wake-up flag DATWKF (UART_WKSTS[1]) is generated. The incoming data wake-up is shown in Figure 6.13-7.

Note1: The UART controller clock source should be selected as HIRC and the compensation time for start bit is about 91.129us. It means that the value of STCOMP (UART_DWKCOMP[15:0]) can be set as 0x1116.

Note2: 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.

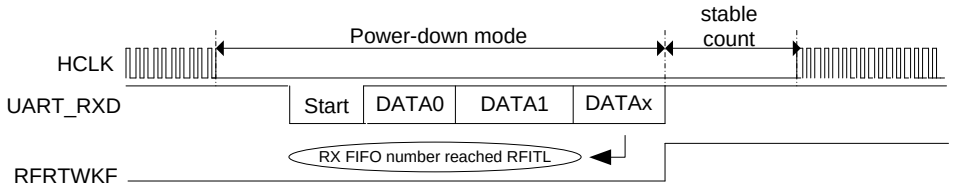
Note 2: UART_CLK stable count means UART clock source recovery stable count.

Figure 6.13-7 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 6.13-8.

Note: The UART controller clock source should be selected as LXT in Power-down mode to receive data.



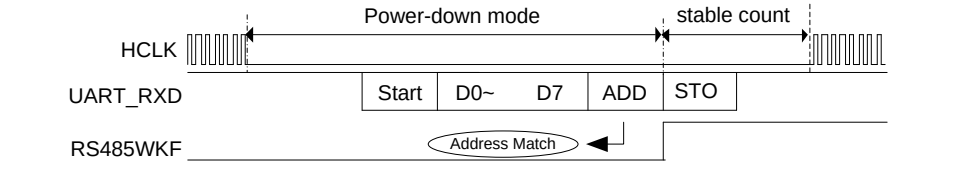
Note: Stable count means HCLK source recovery stable count.

Figure 6.13-8 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]) that cause 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 6.13-9.

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.

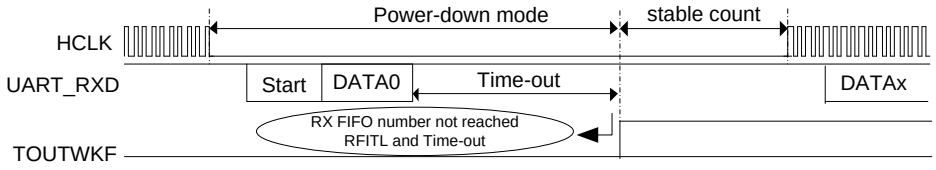
Figure 6.13-9 UART RS-485 AAD Mode Address Match Wake-up

Received Data FIFO Threshold Time-out Wake-up

The received data FIFO threshold time-out wake-up function is enabled by setting WKRFRTEN (UART_WKCTL[2]) and WKTOUTEN (UART_WKCTL[4]). Setting TOCNTEN (UART_INTEN[11]) to enable receiver buffer time-out counter. In Power-down mode, when the number of received data in

RX FIFO does not reach the threshold value RFITL (UART_FIFO[7:4]) and the time-out counter equals to the time-out value TOIC (UART_TOUT[7:0]), it can wake-up the system. If the WKTOUTEN (UART_WKCTL[4]) is enabled, when the time-out counter equals to the time-out value TOIC (UART_TOUT[7:0]) that cause the Received Data FIFO threshold time-out wake-up wake-up flag TOUTWKF (UART_WKSTS[4]) is generated. The Received Data FIFO threshold time-out wake-up is shown in Figure 6.13-10.

Note: The UART controller clock source should be selected as LXT or LIRC in Power-down mode to receive data.



Note: Stable count means HCLK source recovery stable count.

Figure 6.13-10 UART Received Data FIFO threshold time-out wake-up

6.13.5.7 UART Controller Interrupt and Status

Each UART controller supports ten types of interrupts including:

- Receive Data Available Interrupt (RDAINT)
- Transmit Holding Register Empty Interrupt (THERINT)
- Transmitter Empty Interrupt (TXENDIF)
- Receive Line Status Interrupt (RLSINT)
  - Break Interrupt Flag (BIF)
  - Framing Error Flag (FEF)
  - Parity Error Flag (PEF)
  - RS-485 Address Byte Detect Flag (ADDRDETF)
- MODEM Status Interrupt (MODEMINT)
  - Detect nCTS State Change Flag (CTSDETF)
- Receiver Buffer Time-out Interrupt (RXTOINT)
- Buffer Error Interrupt (BUFERRINT)
  - TX Overflow Error Interrupt Flag (TXOVIF)
  - RX Overflow Error Interrupt Flag (RXOVIF)
- 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 6.13-9 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
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 6.13-9 UART controller Interrupt Source and Flag List

6.13.5.8 UART Function Mode

The UART controller provides UART function (Setting FUNCSEL (UART_FUNCSEL [1:0]) to '00' to enable UART function mode).

The UART 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 6.13-10 and Table 6.13-11 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 6.13-10 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 6.13-11 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 6.13-11.

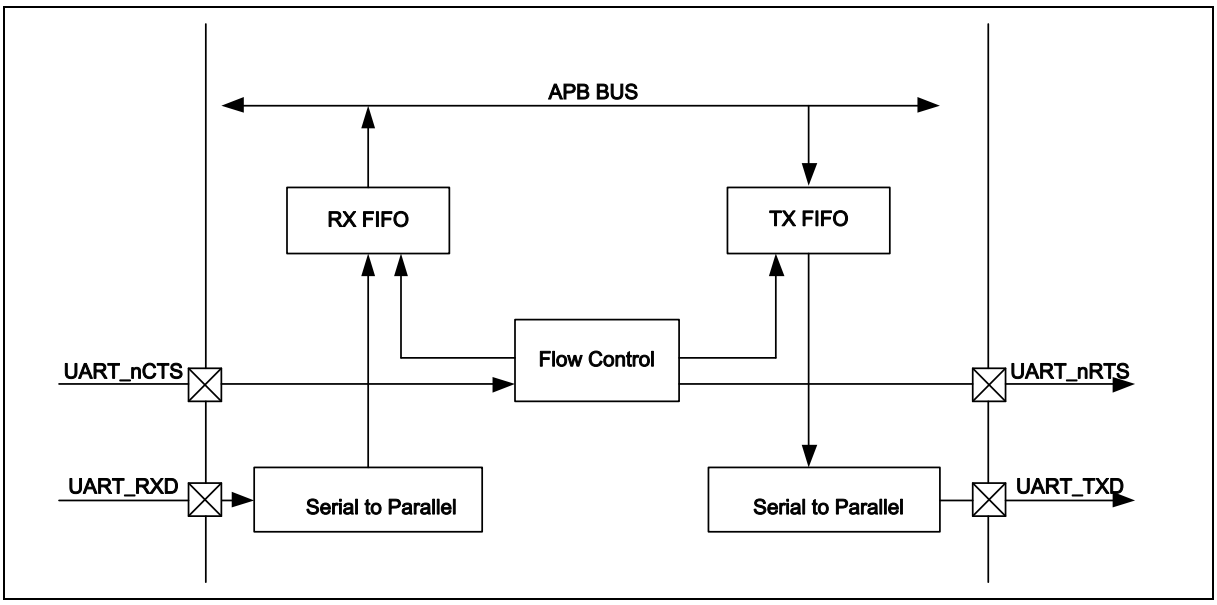


Figure 6.13-11 Auto-Flow Control Block Diagram

Figure 6.13-12 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.

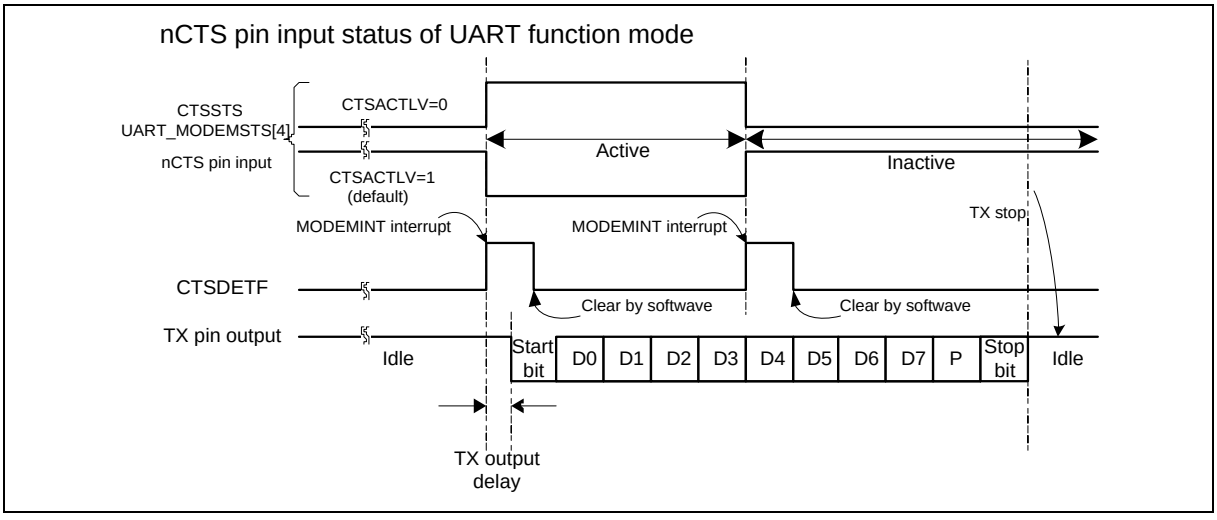


Figure 6.13-12 UART nCTS Auto-Flow Control Enabled

As shown in Figure 6.13-13, 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.

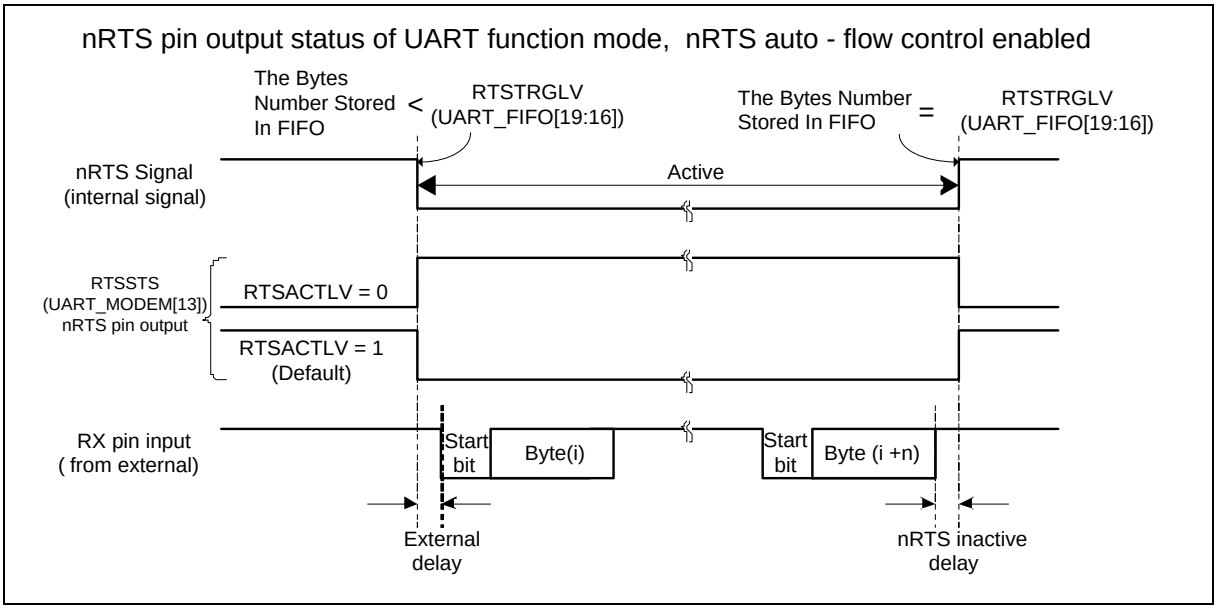


Figure 6.13-13 UART nRTS Auto-Flow Control Enabled

As shown in Figure 6.13-14, 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.

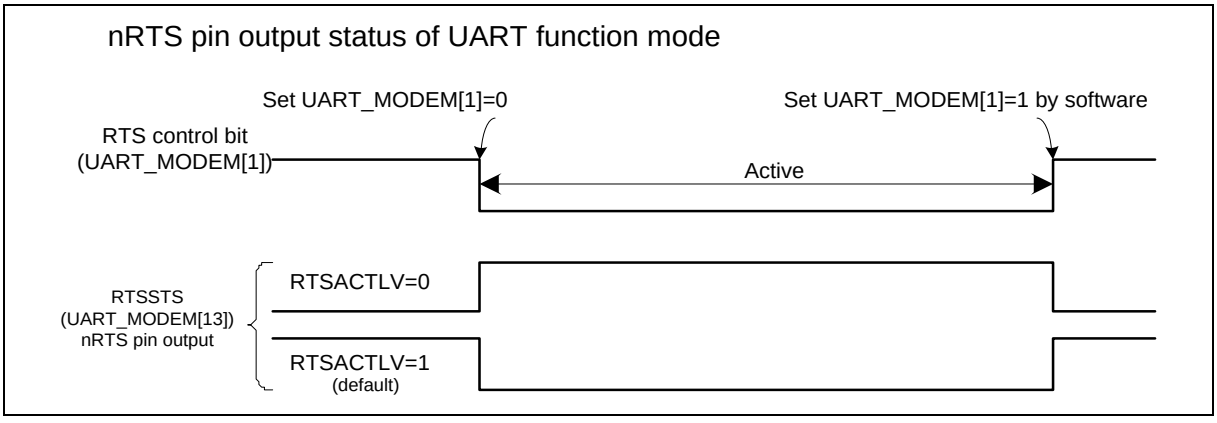


Figure 6.13-14 UART nRTS Auto-Flow with Software Control

6.13.5.9 IrDA Function Mode

The UART controller also provides Serial IrDA (SIR, Serial Infrared) function (Setting UART_FUNCSEL [1:0] to '10' to enable the IrDA function). The SIR specification defines a short-range infrared asynchronous serial transmission mode with one start bit, 8 data bits, and 1 stop bit. The maximum data rate is 115.2 kbps. The IrDA SIR block contains an IrDA SIR protocol encoder/decoder. The IrDA SIR protocol is half-duplex only. 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 6.13-15.

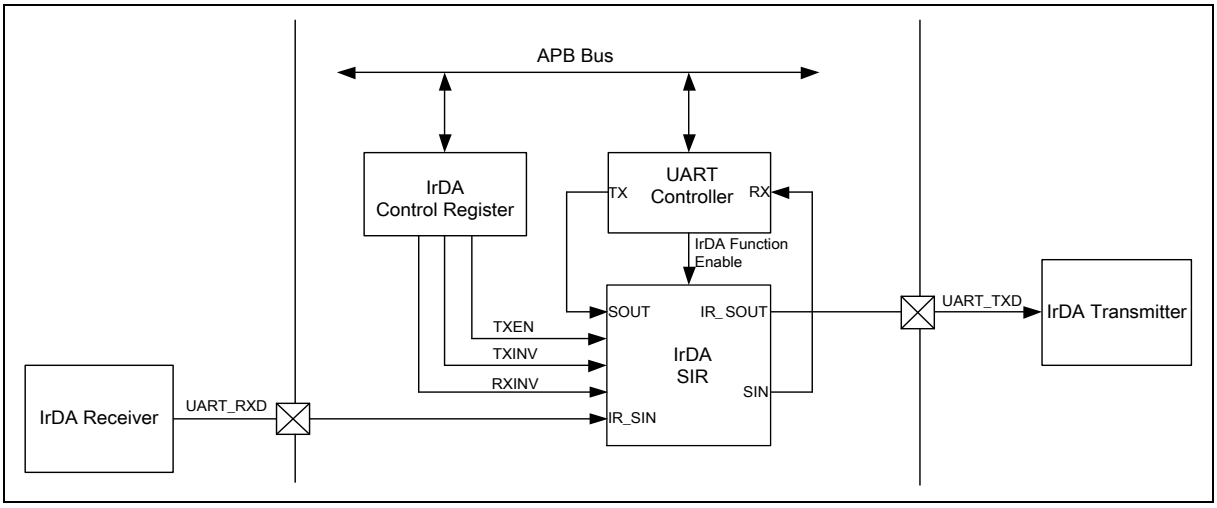


Figure 6.13-15 IrDA Control Block Diagram

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 6.13-16 is IrDA encoder/decoder waveform.

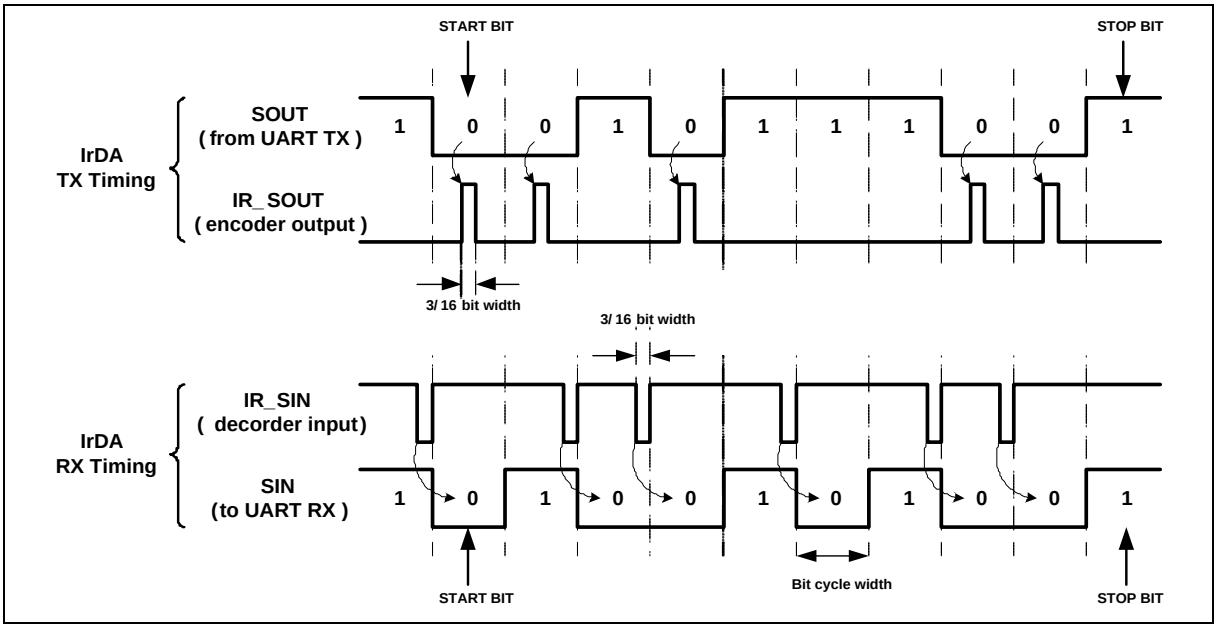


Figure 6.13-16 IrDA TX/RX Timing Diagram

6.13.5.10 RS-485 Function Mode

Another alternate function of UART controller is RS-485 function (user must set UART_FUNCSEL [1:0] to '11' to enable RS-485 function), and direction control provided by nRTS pin from an asynchronous serial port. The RS-485 transceiver control is implemented by using the nRTS control signal to enable the RS-485 driver. Many characteristics of the RX and TX are same as UART in RS-485 mode.

The UART controller can be configured as an RS-485 addressable slave and the RS-485 master transmitter will identify an address character by setting the parity (9-th bit) to 1. For data characters, the parity is set to 0. Software can use UART_LINE register to control the 9-th bit (When the PBE, EPE and SPE are set, the 9-th bit is transmitted 0 and when PBE and SPE are set and EPE is cleared, the 9-th bit is transmitted 1).

The controller supports three operation modes: RS-485 Normal Multidrop Operation Mode (NMM), RS-485 Auto Address Detection Operation Mode (AAD) and RS-485 Auto Direction Control Operation Mode (AUD). Software can choose any operation mode by programming the UART_ALTCTL register, and drive the transfer delay time between the last stop bit leaving the TX FIFO and the de-assertion of by setting DLY (UART_TOUT [15:8]) register.

RS-485 Normal Multidrop Operation Mode (NMM)

In RS-485 Normal Multidrop Operation Mode (RS485NMM (UART_ALTCTL[8]) = 1), in first, software must decide the data which before the address byte be detected will be stored in RX FIFO or not. If software wants to ignore any data before address byte detected, the flow is set RXOFF (UART_FIFO [8]) then enable RS485NMM (UART_ALTCTL [8]) and the receiver will ignore any data until an address byte is detected (bit 9 = 1) and the address byte data will be stored in the RX FIFO. If software wants to receive any data before address byte detected, the flow is disables RXOFF (UART_FIFO [8]) then enable RS485NMM (UART_ALTCTL [8]) and the receiver will received any data.

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

RS-485 Auto Address Detection Operation Mode (AAD)

In RS-485 Auto Address Detection Operation Mode (RS485AAD (UART_ALTCTL[9]) = 1), the receiver will ignore any data until an address byte is detected (bit 9 = 1) and the address byte data matches the ADDRMV (UART_ALTCTL[31:24]) value. The address byte data will be stored in the RX FIFO. The all received byte data will be accepted and stored in the RX FIFO until an address byte data not match the ADDRMV (UART_ALTCTL[31:24]) value.

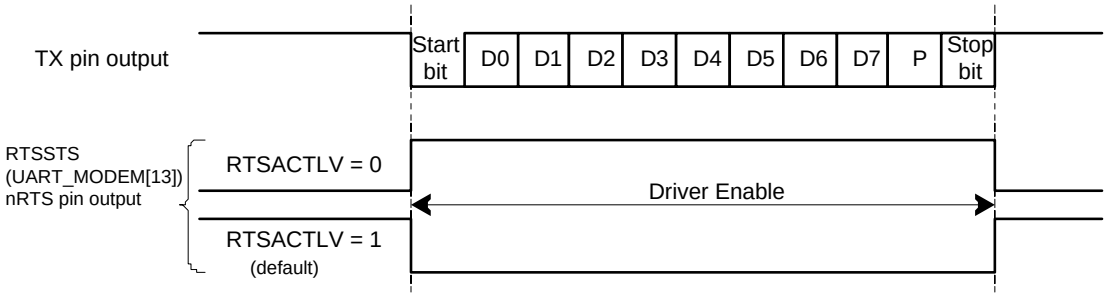
RS-485 Auto Direction Function (AUD)

Another option function of RS-485 controllers is RS-485 auto direction control function (RS485AUD (UART_ALTCTL[10) = 1). The RS-485 transceiver control is implemented by using the nRTS control signal from an asynchronous serial port. The nRTS line is connected to the RS-485 transceiver enable pin such that setting the nRTS line to high (logic 1) enables the RS-485 transceiver. Setting the nRTS line to low (logic 0) puts the transceiver into the tri-state condition to disabled. User can set RTSACTLV in UART_MODEM register to change the nRTS driving level.

Figure 6.13-17 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)

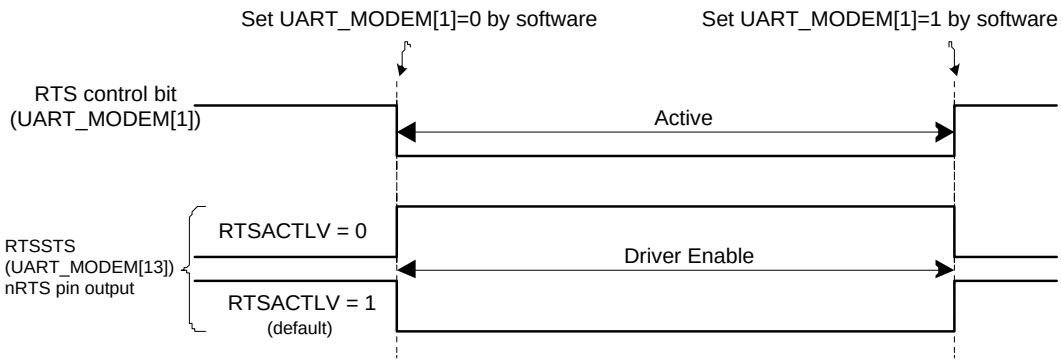


Note: RS485AUD(UART_ALTCTL[10]) = 1, the nRTS pin output by hardware control only.

Figure 6.13-17 RS-485 nRTS Driving Level in Auto Direction Mode

Figure 6.13-18 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 6.13-19.

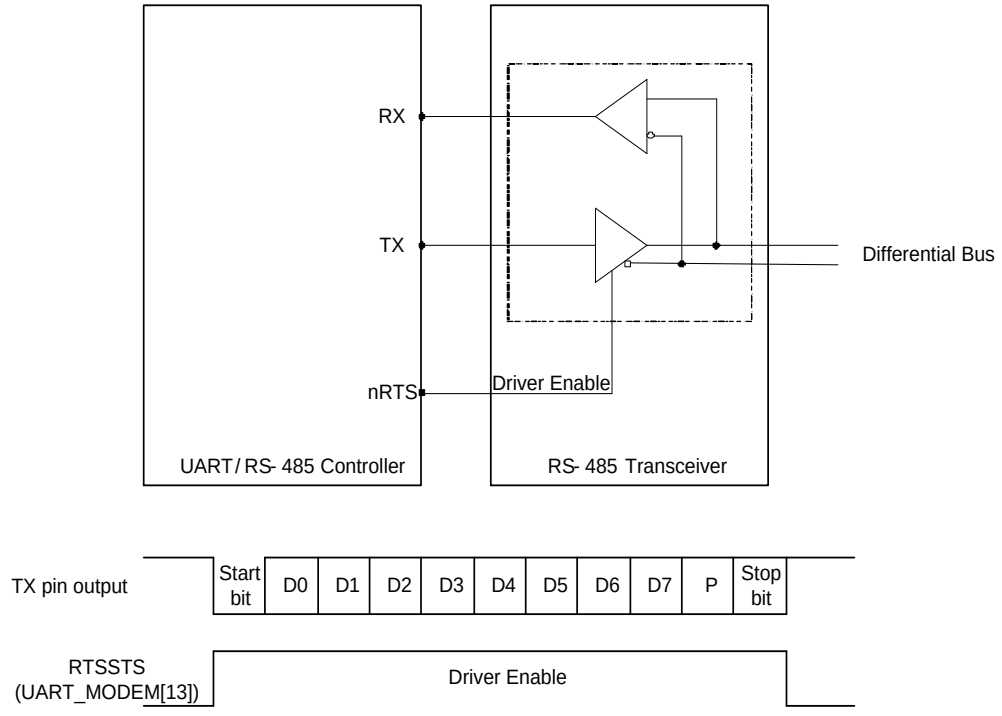


Note: RS485AUD (UART_ALTCTL[10]) = 0, support nRTS control bit by software control only.

Figure 6.13-18 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 6.13-19 Structure of RS-485 Frame

6.13.5.11 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 keeps at RX state during the single-wire bus is idle. By writing data to TX buffer DAT (UART_DAT[7:0]), the single wire transfers the bus state to TX state immediately. After end of trasmition, the single wire transfer bus type from TX state to RX state.

To reduce the bus conflict problem, the UART controller supports flow control and bit error detection. The nRTS is inactivated during the bus keeping in TX state. The default state of the UART is RX mode, and the UART only changes to TX mode to send data after the nCTS is inactivated when ATOCTSEN (UART_INTEN[13]) is enabled. Under the TX state, the UART controller will monitor bus state. If the bus state is not equal to the UART controller TX state, the SWBEIF(UART_INTSTS[16]) will be set.

Note1: Before writing data to TX buffer, the bus state should be checked in idle state by RXIDLE(UART_FIFOSTS[29]). And the bus confliction may cause RX receive broken data.

Note2: Single-wire does not support auto-flow control because the nRTS is actived automatic during TX transmitted.

6.13.5.12 PDMA Transfer Function

The UART controller supports PDMA transfer function.

By configuring PDMA parameter and set UART_DAT as the PDMA destination address. When TXPDMAEN (UART_INTEN[14]) is set to 1, the controller will issue request to PDMA controller to start the PDMA transmission process automatically.

By configuring PDMA parameter and set UART_DAT as the PDMA source address. When RXPDMAEN (UART_INTEN[15]) is set to 1, the controller will start the PDMA reception process. UART controller will issue request to PDMA controller automatically when there is data in the RX FIFO buffer.

Note: If STOPn (PDMA_STOP[n]) is set to stop UART RXPDMA task and the UART receive is not finish. UART controller will complete the transfer and stored current receive data in receive buffer. By reading RXEMPTY (UART_FIFOSTS[14]) to check there is valid data in receive buffer or not.

6.13.6 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=0,1,2,3,4,5,6,7**				
UART_DAT **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x00	R/W	UART Receive/Transmit Buffer Register	Undefined
UART_INTEN **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x04	R/W	UART Interrupt Enable Register	0x0000_0000
UART_FIFO **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x08	R/W	UART FIFO Control Register	0x0000_0101
UART_LINE **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x0C	R/W	UART Line Control Register	0x0000_0000
UART_MODEM **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x10	R/W	UART Modem Control Register	0x0000_0200
UART_MODEMSTS **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x14	R/W	UART Modem Status Register	0x0000_0110
UART_FIFOSTS **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x18	R/W	UART FIFO Status Register	0xB040_4000
UART_INTSTS **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x1C	R/W	UART Interrupt Status Register	0x0040_0002
UART_TOUT **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x20	R/W	UART Time-out Register	0x0000_0000
UART_BAUD **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x24	R/W	UART Baud Rate Divider Register	0x0F00_0000
UART_IRDA **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x28	R/W	UART IrDA Control Register	0x0000_0040
UART_ALTCTL **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x2C	R/W	UART Alternate Control/Status Register	0x0000_000C
UART_FUNCSEL **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x30	R/W	UART Function Select Register	0x0000_0000
UART_BRCOMP **x=0,1,4,5**	UARTx_BA+0x3C	R/W	UART Baud Rate Compensation Register	0x0000_0000
UART_WKCTL **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x40	R/W	UART Wake-up Control Register	0x0000_0000
UART_WKSTS **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x44	R/W	UART Wake-up Status Register	0x0000_0000
UART_DWKCOMP **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x48	R/W	UART Incoming Data Wake-up Compensation Register	0x0000_0000

6.13.7 Register Description

UART Receive/Transmit Buffer Register (UART_DAT)

Register	Offset	R/W	Description	Reset Value
UART_DAT **x=0,1,2,3,4,5,6,7**	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=0,1,2,3,4,5,6,7	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		
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 select 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.
[10:7]	**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=0,1,2,3,4,5,6,7	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]) 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]) be set.
[0]	**Reserved**	Reserved.

UART Line Control Register (UART_LINE)

Register	Offset	R/W	Description	Reset Value
UART_LINE **x=0,1,2,3,4,5,6,7**	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 waited for TXRXACT (UART_FIFOSTS[31]) is cleared. When the configuration is done, cleared TXRXDIS (UART_FUNCSEL[3]) to activate UART controller. **Note 2:** This bit is valid when FUNCSEL (UART_FUNCSEL[1:0]) is select UART, LIN or RS485 function.
[8]	**TXDINV**	**TX Data Inverted** 0 = Transmitted data signal inverted Disabled. 1 = Transmitted data signal inverted Enabled. **Note 1:** Before setting this bit, TXRXDIS (UART_FUNCSEL[3]) should be set then waited for TXRXACT (UART_FIFOSTS[31]) is cleared. When the configuration is done, cleared TXRXDIS (UART_FUNCSEL[3]) to activate UART controller. **Note 2:** This bit is valid when FUNCSEL (UART_FUNCSEL[1:0]) is select UART, LIN or RS485 function.
[7]	**PSS**	**Parity Bit Source Selection** The parity bit can be selected to be generated and checked automatically or by software. 0 = Parity bit is generated by EPE (UART_LINE[4]) and SPE (UART_LINE[5]) setting and checked automatically. 1 = Parity bit generated and checked by software. **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=0,1,2,3,4,5,6,7**	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 6.13-13 and Figure 6.13-14 for UART function mode. **Note 2:** Refer to Figure 6.13-17 and Figure 6.13-18 for RS-485 function mode. **Note 3:** Before setting this bit, TXRXDIS (UART_FUNCSEL[3]) should be set then waited for TXRXACT (UART_FIFOSTS[31]) is cleared. When the configuration is done, cleared TXRXDIS (UART_FUNCSEL[3]) to activate UART controller.
[8:2]	**Reserved**	Reserved.
[1]	**RTS**	**nRTS (Request-to-send) Signal Control** This bit is direct control internal nRTS signal active or not, and then drive the nRTS pin output with RTSACTLV bit configuration. 0 = nRTS signal is active. 1 = nRTS signal is inactive. **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 is support this feature.
[0]	**Reserved**	Reserved.

UART Modem Status Register (UART_MODEMSTS)

Register	Offset	R/W	Description	Reset Value
UART_MODEMSTS **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x14	R/W	UART Modem Status Register	0x0000_0110

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

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**CTSACTLV**	**nCTS Pin Active Level** This bit defines the active level state of nCTS pin input. 0 = nCTS pin input is high level active. 1 = nCTS pin input is low level active. (Default) **Note:** Before setting this bit, TXRXDIS (UART_FUNCSEL[3]) should be set then waited for TXRXACT (UART_FIFOSTS[31]) is cleared. When the configuration is done, cleared TXRXDIS (UART_FUNCSEL[3]) to activate UART controller.
[7:5]	**Reserved**	Reserved.
[4]	**CTSSTS**	**nCTS Pin Status (Read Only)** This bit mirror from nCTS pin input of voltage logic status. 0 = nCTS pin input is low level voltage logic state. 1 = nCTS pin input is high level voltage logic state. **Note:** This bit echoes when UART controller peripheral clock is enabled, and nCTS multi-function port is selected.
[3:1]	**Reserved**	Reserved.
[0]	**CTSDETF**	**Detect nCTS State Change Flag** This bit is set whenever nCTS input has change state, and it will generate Modem interrupt to CPU when MODEMIEN (UART_INTEN [3]) is set to 1. 0 = nCTS input has not change state. 1 = nCTS input has change state. **Note:** This bit can be cleared by writing “1” to it.

UART FIFO Status Register (UART_FIFOSTS)

Register	Offset	R/W	Description	Reset Value
UART_FIFOSTS x=0,1,2,3,4,5,6,7	UARTx_BA+0x18	R/W	UART FIFO Status Register	0xB040_4000

31	30	29	28	27	26	25	24
TXRXACT	Reserved	RXIDLE	TXEMPTYF	Reserved			TXOVIF
23	22	21	20	19	18	17	16
TXFULL	TXEMPTY	TXPTR					
15	14	13	12	11	10	9	8
RXFULL	RXEMPTY	RXPTR					
7	6	5	4	3	2	1	0
Reserved	BIF	FEF	PEF	ADDRDETF	ABRDTOIF	ABRDIF	RXOVIF

Bits	Description	
[31]	**TXRXACT**	**TX and RX Active Status (Read Only)** This bit indicates TX and RX are active or inactive. 0 = TX and RX are inactive. 1 = TX and RX are active. (Default) **Note:** When TXRXDIS (UART_FUNCSEL[3]) is set and both TX and RX are in idle state, this bit is cleared. The UART controller can not transmit or receive data at this moment. Otherwise this bit is set.
[30]	**Reserved**	Reserved.
[29]	**RXIDLE**	**RX Idle Status (Read Only)** This bit is set by hardware when RX is idle. 0 = RX is busy. 1 = RX is idle. (Default)
[28]	**TXEMPTYF**	**Transmitter Empty Flag (Read Only)** This bit is set by hardware when TX FIFO (UART_DAT) is empty and the STOP bit of the last byte has been transmitted. 0 = TX FIFO is not empty or the STOP bit of the last byte has been not transmitted. 1 = TX FIFO is empty and the STOP bit of the last byte has been transmitted. **Note:** This bit is cleared automatically when TX FIFO is not empty or the last byte transmission has not completed.
[27:25]	**Reserved**	Reserved.
[24]	**TXOVIF**	**TX Overflow Error Interrupt Flag** If TX FIFO (UART_DAT) is full, an additional write to UART_DAT will cause this bit to logic 1. 0 = TX FIFO is not overflow. 1 = TX FIFO is overflow. **Note:** This bit can be cleared by writing “1” to it.
[23]	**TXFULL**	**Transmitter FIFO Full (Read Only)**

		This bit indicates TX FIFO full or not. 0 = TX FIFO is not full. 1 = TX FIFO is full. **Note:** This bit is set when the number of usage in TX FIFO Buffer is equal to 16, otherwise it is cleared by hardware.
[22]	**TXEMPTY**	**Transmitter FIFO Empty (Read Only)** This bit indicates TX FIFO empty or not. 0 = TX FIFO is not empty. 1 = TX FIFO is empty. **Note:** When the last byte of TX FIFO has been transferred to Transmitter Shift Register, hardware sets this bit high. It will be cleared when writing data into UART_DAT (TX FIFO not empty).
[21:16]	**TXPTR**	**TX FIFO Pointer (Read Only)** This field indicates the TX FIFO Buffer Pointer. When CPU writes one byte into UART_DAT, TXPTR increases one. When one byte of TX FIFO is transferred to Transmitter Shift Register, TXPTR decreases one. The Maximum value shown in TXPTR is 15. When the using level of TX FIFO Buffer equal to 16, the TXFULL bit is set to 1 and TXPTR will show 0. As one byte of TX FIFO is transferred to Transmitter Shift Register, the TXFULL bit is cleared to 0 and TXPTR will show 15.
[15]	**RXFULL**	**Receiver FIFO Full (Read Only)** This bit initiates RX FIFO full or not. 0 = RX FIFO is not full. 1 = RX FIFO is full. **Note:** This bit is set when the number of usage in RX FIFO Buffer is equal to 16, otherwise it is cleared by hardware.
[14]	**RXEMPTY**	**Receiver FIFO Empty (Read Only)** This bit initiate RX FIFO empty or not. 0 = RX FIFO is not empty. 1 = RX FIFO is empty. **Note:** When the last byte of RX FIFO has been read by CPU, hardware sets this bit high. It will be cleared when UART receives any new data.
[13:8]	**RXPTR**	**RX FIFO Pointer (Read Only)** This field indicates the RX FIFO Buffer Pointer. When UART receives one byte from external device, RXPTR increases one. When one byte of RX FIFO is read by CPU, RXPTR decreases one. The Maximum value shown in RXPTR is 15. When the using level of RX FIFO Buffer equal to 16, the RXFULL bit is set to 1 and RXPTR will show 0. As one byte of RX FIFO is read by CPU, the RXFULL bit is cleared to 0 and RXPTR will show 15.
[7]	**Reserved**	Reserved.
[6]	**BIF**	**Break Interrupt Flag** This bit is set to logic 1 whenever the received data input (RX) is held in the “spacing state” (logic 0) for longer than a full word transmission time (that is, the total time of “start bit” + data bits + parity + stop bits). 0 = No Break interrupt is generated. 1 = Break interrupt is generated. **Note:** This bit can be cleared by writing “1” to it.
[5]	**FEF**	**Framing Error Flag** This bit is set to logic 1 whenever the received character does not have a valid “stop bit” (that is, the stop bit following the last data bit or parity bit is detected as logic 0). 0 = No framing error is generated.

		1 = Framing error is generated. **Note:** This bit can be cleared by writing “1” to it.
[4]	**PEF**	**Parity Error Flag** This bit is set to logic 1 whenever the received character does not have a valid “parity bit”. 0 = No parity error is generated. 1 = Parity error is generated. **Note:** This bit can be cleared by writing “1” to it.
[3]	**ADDRDETF**	**RS-485 Address Byte Detect Flag** 0 = Receiver detects a data that is not an address bit (bit 9 =’0’). 1 = Receiver detects a data that is an address bit (bit 9 =’1’). **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=0,1,2,3,4,5,6,7	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
Reserved	WKINT	BUFERRINT	RXTOINT	MODEMINT	RLSINT	THREINT	RDAINT
7	6	5	4	3	2	1	0
Reserved	WKIF	BUFERRIF	RXTOIF	MODEMIF	RLSIF	THREIF	RDAIF

Bits	Description	
[31]	**ABRINT**	**Auto-baud Rate Interrupt Indicator (Read Only)** This bit is set if ABRIEN (UART_INTEN[18]) and ABRIF (UART_ALTCTL[17]) are both set to 1. 0 = No Auto-baud Rate interrupt is generated. 1 = The Auto-baud Rate interrupt is generated.
[30]	**TXENDINT**	**Transmitter Empty Interrupt Indicator (Read Only)** This bit is set if TXENDIEN (UART_INTEN[22]) and TXENDIF(UART_INTSTS[22]) are both set to 1. 0 = No Transmitter Empty interrupt is generated. 1 = Transmitter Empty interrupt is generated.
[29]	**HWBUFEINT**	**PDMA Mode Buffer Error Interrupt Indicator (Read Only)** This bit is set if BUFERRIEN (UART_INTEN[5]) and HWBUFEIF (UART_INTSTS[21]) are both set to 1. 0 = No buffer error interrupt is generated in PDMA mode. 1 = Buffer error interrupt is generated in PDMA mode.
[28]	**HWTOINT**	**PDMA Mode RX Time-out Interrupt Indicator (Read Only)** This bit is set if RXTOIEN (UART_INTEN[4]) and HWTOIF(UART_INTSTS[20]) are both set to 1. 0 = No RX time-out interrupt is generated in PDMA mode. 1 = RX time-out interrupt is generated in PDMA mode.
[27]	**HWMODINT**	**PDMA Mode MODEM Status Interrupt Indicator (Read Only)** This bit is set if MODEMIEN (UART_INTEN[3]) and HWMODIF(UART_INTSTS[19]) are both set to 1. 0 = No Modem interrupt is generated in PDMA mode. 1 = Modem interrupt is generated in PDMA mode.
[26]	**HWRLSINT**	**PDMA Mode Receive Line Status Interrupt Indicator (Read Only)** This bit is set if RLSIEN (UART_INTEN[2]) and HWRLSIF(UART_INTSTS[18]) are both set to 1.

		0 = No RLS interrupt is generated in PDMA mode. 1 = RLS interrupt is generated in PDMA mode.
[25]	**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 select UART Single-wire mode. **Note 2:** This bit can be cleared by writing “1” to it.
[15]	**Reserved**	Reserved.
[14]	**WKINT**	**UART Wake-up Interrupt Indicator (Read Only)** This bit is set if WKIEN (UART_INTEN[6]) and WKIF (UART_INTSTS[6]) are both set to 1. 0 = No UART wake-up interrupt is generated. 1 = UART wake-up interrupt is generated.
[13]	**BUFERRINT**	**Buffer Error Interrupt Indicator (Read Only)** This bit is set if BUFERRIEN(UART_INTEN[5]) and BUFERRIF(UART_ INTSTS[5]) are both set to 1. 0 = No buffer error interrupt is generated. 1 = Buffer error interrupt is generated.
[12]	**RXTOINT**	**RX Time-out Interrupt Indicator (Read Only)** This bit is set if RXTOIEN (UART_INTEN[4]) and RXTOIF(UART_INTSTS[4]) are both set to 1. 0 = No RX time-out interrupt is generated. 1 = RX time-out interrupt is generated.
[11]	**MODEMINT**	**MODEM Status Interrupt Indicator (Read Only)** This bit is set if MODEMIEN(UART_INTEN[3]) and MODEMIF(UART_INTSTS[3]) are both set to 1 0 = No Modem interrupt is generated. 1 = Modem interrupt is generated..
[10]	**RLSINT**	**Receive Line Status Interrupt Indicator (Read Only)** This bit is set if RLSIEN (UART_INTEN[2]) and RLSIF(UART_INTSTS[2]) are both set to 1. 0 = No RLS interrupt is generated. 1 = RLS interrupt is generated.
[9]	**THREINT**	**Transmit Holding Register Empty Interrupt Indicator (Read Only)** This bit is set if THREIEN (UART_INTEN[1]) and THREIF(UART_INTSTS[1]) are both set to 1. 0 = No THRE interrupt is generated. 1 = THRE interrupt is generated.
[8]	**RDAINT**	**Receive Data Available Interrupt Indicator (Read Only)** This bit is set if RDAIEN (UART_INTEN[0]) and RDAIF (UART_INTSTS[0]) are both set to 1. 0 = No RDA interrupt is generated.

		1 = RDA interrupt is generated.
[7]	**Reserved**	Reserved.
[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** This bit is set when the last data of TX FIFO is transferred to Transmitter Shift Register. If THREIEN (UART_INTEN[1]) is enabled, the THRE interrupt will be generated.

		0 = No THRE interrupt flag is generated. 1 = THRE interrupt flag is generated. **Note:** This bit is read only and it will be cleared when writing data into UART_DAT (TX FIFO not empty).
[0]	**RDAIF**	**Receive Data Available Interrupt Flag** When the number of bytes in the RX FIFO equals the RFITL then the RDAIF(UART_INTSTS[0]) will be set. If RDAIEN (UART_INTEN [0]) is enabled, the RDA interrupt will be generated. 0 = No RDA interrupt flag is generated. 1 = RDA interrupt flag is generated. **Note:** This bit is read only and it will be cleared when the number of unread bytes of RX FIFO drops below the threshold level (RFITL(UART_FIFO[7:4]).

UART Time-out Register (UART_TOUT)

Register	Offset	R/W	Description	Reset Value
UART_TOUT **x=0,1,2,3,4,5,6,7**	UARTx_BA+0x20	R/W	UART Time-out Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**DLY**	**TX Delay Time Value** This field is used to programming the transfer delay time between the last stop bit and next start bit. The unit is bit time.
[7:0]	**TOIC**	**Time-out Interrupt Comparator** The time-out counter resets and starts counting (the counting clock = baud rate) whenever the RX FIFO receives a new data word if time out counter is enabled by setting TOCNTEN (UART_INTEN[11]). Once the content of time-out counter is equal to that of time-out interrupt comparator (TOIC (UART_TOUT[7:0])), a receiver time-out interrupt (RXTOINT(UART_INTSTS[12])) is generated if RXTOIEN (UART_INTEN [4]) enabled. A new incoming data word or RX FIFO empty will clear RXTOIF (UART_INTSTS[4]). In order to avoid receiver time-out interrupt generation immediately during one character is being received, TOIC value should be set between 40 and 255. 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=0,1,2,3,4,5,6,7	UARTx_BA+0x24	R/W	UART Baud Rate Divider Register	0x0F00_0000

31	30	29	28	27	26	25	24
Reserved		BAUDM1	BAUDM0	EDIVM1			
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
BRD							
7	6	5	4	3	2	1	0
BRD							

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29]	**BAUDM1**	**BAUD Rate Mode Selection Bit 1** This bit is baud rate mode selection bit 1. UART provides three baud rate calculation modes. This bit combines with BAUDM0 (UART_BAUD[28]) to select baud rate calculation mode. The detail description is shown in Table 6.13-4. **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 6.13-4.
[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 6.13-4.
[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 6.13-4.

UART IrDA Control Register (UART_IRDA)

Register	Offset	R/W	Description	Reset Value
UART_IRDA x=0,1,2,3,4,5,6,7	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 waited for TXRXACT (UART_FIFOSTS[31]) is cleared. When the configuration is done, cleared TXRXDIS (UART_FUNCSEL[3]) to activate UART controller. **Note 2:** This bit is valid when FUNCSEL (UART_FUNCSEL[1:0]) is select IrDA function.
[5]	**TXINV**	**IrDA Inverse Transmitting Output Signal** 0 = None inverse transmitting signal. (Default). 1 = Inverse transmitting output signal. **Note 1:** Before setting this bit, TXRXDIS (UART_FUNCSEL[3]) should be set then waited for TXRXACT (UART_FIFOSTS[31]) is cleared. When the configuration is done, cleared TXRXDIS (UART_FUNCSEL[3]) to activate UART controller. **Note 2:** This bit is valid when FUNCSEL (UART_FUNCSEL[1:0]) is select IrDA function.
[4:2]	**Reserved**	Reserved.
[1]	**TXEN**	**IrDA Receiver/Transmitter Selection Enable Bit** 0 = IrDA Transmitter Disabled and Receiver Enabled. (Default) 1 = IrDA Transmitter Enabled and Receiver Disabled.
[0]	**Reserved**	Reserved.

UART Alternate Control/Status Register (UART_ALTCTL)

Register	Offset	R/W	Description	Reset Value
UART_ALTCTL **x=0,1,2,3,4,5,6, 7**	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=0,1,2,3,4,5,6,7**	UARTx_BA+0x30	R/W	UART Function Select Register	0x0000_0000

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

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**TXRXDIS**	**TX and RX Disable Bit** Setting this bit can disable TX and RX. 0 = TX and RX Enabled. 1 = TX and RX Disabled. **Note:** The TX and RX will not disable immediately when this bit is set. The TX and RX compelet current task before disable TX and RX. When TX and RX disable, the TXRXACT (UART_FIFOSTS[31]) is cleared.
[2:0]	**FUNCSEL**	**Function Select** 000 = UART function. 010 = IrDA function. 011 = RS-485 function. 100 = UART Single-wire function. Others = Reserved.

UART Baud Rate Compensation Register (UART_BRCOMP)

Register	Offset	R/W	Description	Reset Value
UART_BRCOMP x=0,1,3,5	UARTx_BA+0x3C	R/W	UART Baud Rate Compensation Register	0x0000_0000

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

Bits	Description	
[31]	**BRCOMPDEC**	**Baud Rate Compensation Decrease** 0 = Positive (increase one module clock) compensation for each compensated bit. 1 = Negative (decrease one module clock) compensation for each compensated bit.
[30:9]	**Reserved**	Reserved.
[8:0]	**BRCOMP**	**Baud Rate Compensation Patten** These 9-bits are used to define the relative bit is compensated or not. BRCOMP[7:0] is used to define the compensation of UART_DAT[7:0] and BRCOM[8] is used to define the parity bit.

UART Wake-up Control Register (UART_WKCTL)

Register	Offset	R/W	Description	Reset Value
UART_WKCTL **x=0,1,2,3,4,5,6,7**	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. **Note 3:** This bit is valid in UART0, UART1, UART4 and UART5.
[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. **Note 3:** This bit is valid in UART0, UART1, UART4 and UART5.
[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. **Note 3:** This bit is valid in UART0, UART1, UART4 and UART5.
[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=0,1,2,3,4,5,6,7**	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. **Note 3:** This bit is valid in UART0, UART1, UART4 and UART5.
[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. **Note 3:** This bit is valid in UART0, UART1, UART4 and UART5.
[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. **Note 3:** This bit is valid in UART0, UART1, UART4 and UART5.
[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=0,1,2,3,4,5,6,7**	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 wake-up from Power-down mode. **Note:** It is valid only when WKDATEN (UART_WKCTL[1]) is set.

6.14 Serial Peripheral Interface (SPI)

6.14.1 Overview

The Serial Peripheral Interface (SPI) applies to synchronous serial data communication and allows full duplex transfer. Devices communicate in Master/Slave mode with the 4-wire bi-direction interface. The chip contains one set of SPI controller performing a serial-to-parallel conversion on data received from a peripheral device, and a parallel-to-serial conversion on data transmitted to a peripheral device. 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.

6.14.2 Features

- SPI Mode
  - Supports one set of SPI controller
  - Supports Master or Slave mode operation
  - Configurable bit length of a transaction word from 8 to 32-bit
  - Provides separate 4-level 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
  - Master mode up to 24 MHz and Slave mode up to 16 MHz (when chip works at V_{DD} = 1.8~3.6V)
  - Supports one data channel half-duplex transfer
  - Supports receive-only mode
  - Supports PDMA transfer
- I^2S Mode
  - Supports one set of I^2S by SPI controller
  - Interface with external audio CODEC
  - Supports Master or Slave mode
  - Capable of handling 8-, 16-, 24- and 32-bit word sizes
  - Supports monaural and stereo audio data
  - Supports PCM mode A, PCM mode B, I^2S and MSB justified data format
  - Each provides two 4-level FIFO data buffers, one for transmitting and the other for receiving
  - Supports two PDMA requests, one for transmitting and the other for receiving
  - Generates interrupt requests when buffer levels cross a programmable boundary

6.14.3 Block Diagram

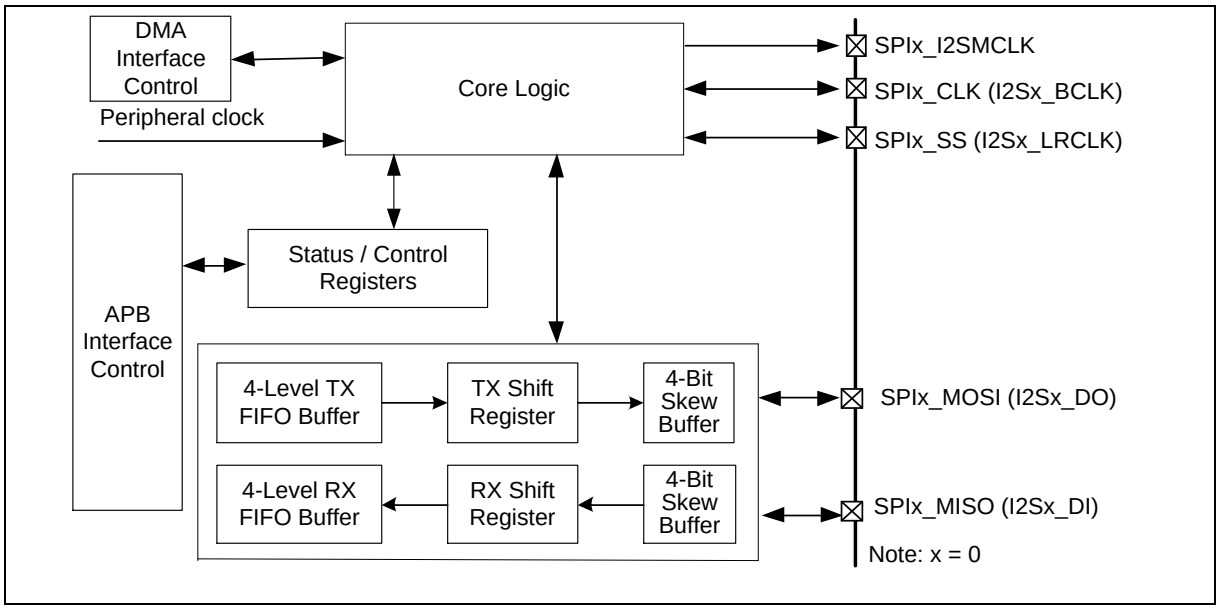


Figure 6.14-1 SPI Block Diagram

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 8~16 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 8~16 bits.

TX Shift Register:

The transmit shift register is a 32-bit wide register buffer. The transmit data is loaded from the TX FIFO buffer and shifted out bit-by-bit to the skew buffer.

RX Shift Register:

The receive shift register is also a 32-bit wide register buffer. The receive data is shift in bit-by-bit from the skew buffer and is loaded into RX FIFO buffer when a transaction done.

Skew Buffer:

The skew buffer is a 4-level 1-bit buffer. There are two skew buffers for transmitting and receiving side. For receiving side, it is used to shift bits into RX shift register from SPI bus. For transmitting side, it is used to shift bits into SPI bus from TX shift register.

6.14.4 Basic Configuration

6.14.4.1 SPI0 Basic 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]).

SPI/I^{2}S (SPI0) Interface Controller Pin description is shown as follows:

Pin	SPI Mode Description	I^{2}S Mode Description
SPIx_SS	SPI slave selection pin	I^{2}S left/right channel synchronization clock pin (I2Sx_LRCLK)
SPIx_CLK	SPI clock pin	I^{2}S bit clock pin (I2Sx_BCLK)
SPIx_MISO	SPI master input or slave output pin	I^{2}S data input pin (I2Sx_DI)
SPIx_MOSI	SPI master output or slave input pin	I^{2}S data output pin (I2Sx_DO)
SPIx_I2SMCLK	Not available	I^{2}S Master clock output pin

Table 6.14-1 SPI/I^{2}S Interface Controller Pin Description (SPI0)

6.14.5 Functional Description

6.14.5.1 Terminology

SPI Peripheral Clock and SPI Bus Clock

The SPI controller needs the peripheral clock to drive the SPI logic unit to perform the data transfer. The peripheral clock rate is determined by the settings of clock 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.

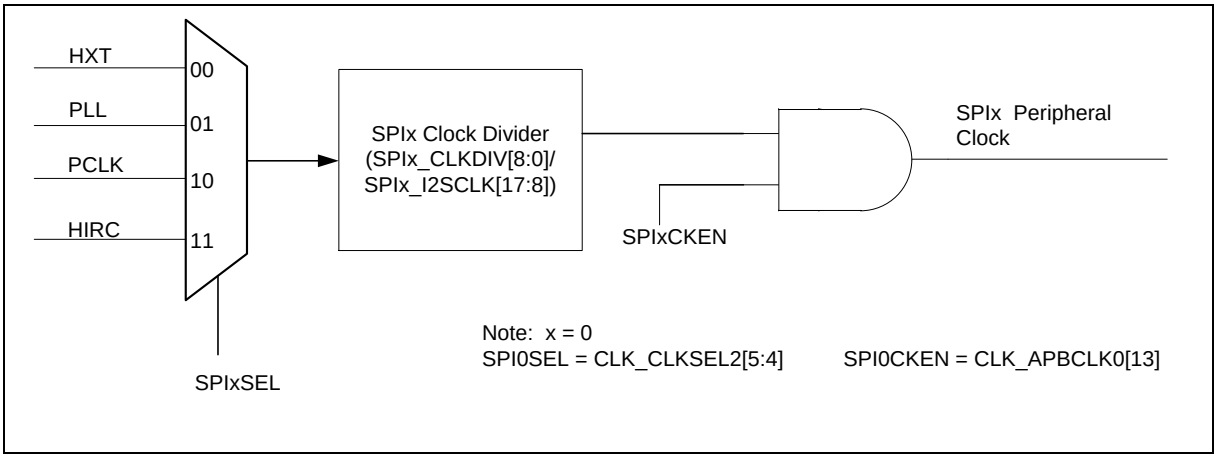


Figure 6.14-2 SPI Peripheral Clock

In Master mode, the frequency of the SPI bus clock is equal to the peripheral clock rate. In general, the SPI bus clock is denoted as SPI clock. In Slave mode, the SPI bus clock is provided by a master device. The frequency of SPI peripheral clock cannot be faster than the system clock rate regardless of Master or Slave mode. If the clock source of peripheral clock is not system clock, the frequency of SPI peripheral clock shall be slower than the system clock frequency regardless of Master or Slave mode.

In I^{2}S mode, the peripheral clock rate is equal to I^{2}S bit clock rate determined by 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.

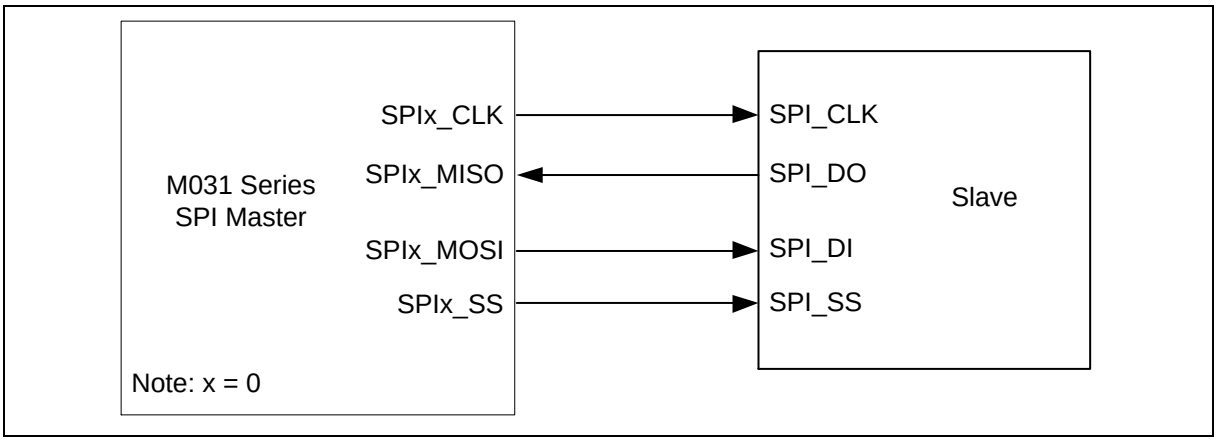


Figure 6.14-3 SPI Full-Duplex Master Mode Application Block Diagram

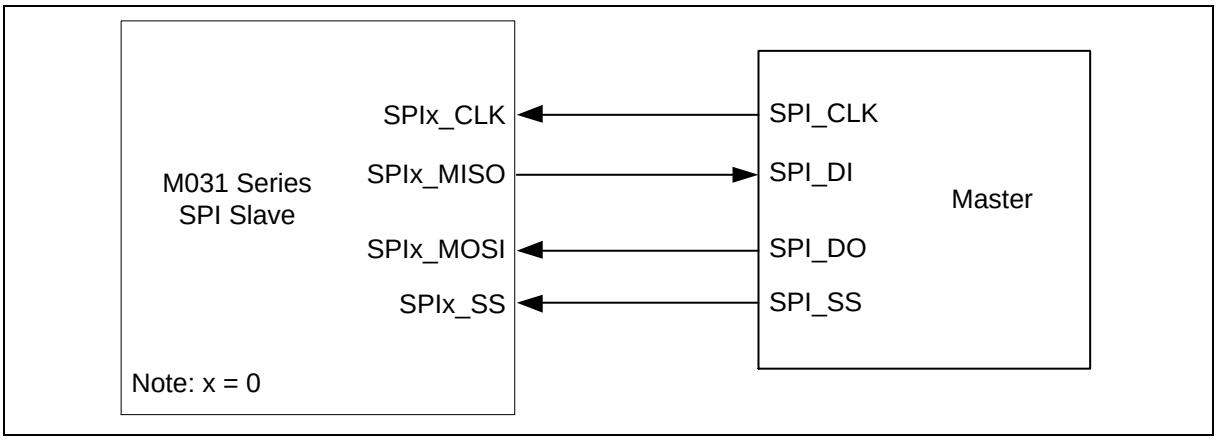


Figure 6.14-4 SPI Full-Duplex Slave Mode Application Block Diagram

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.

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.

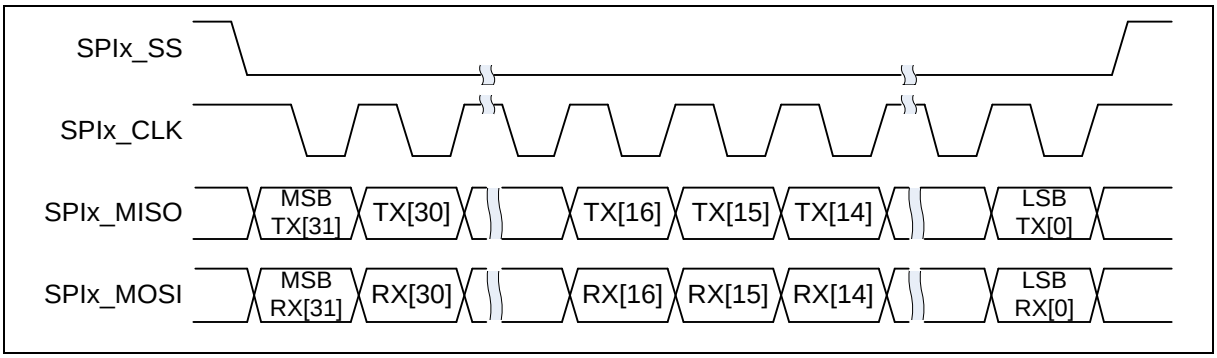


Figure 6.14-532-bit in One Transaction

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).

6.14.5.2 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.

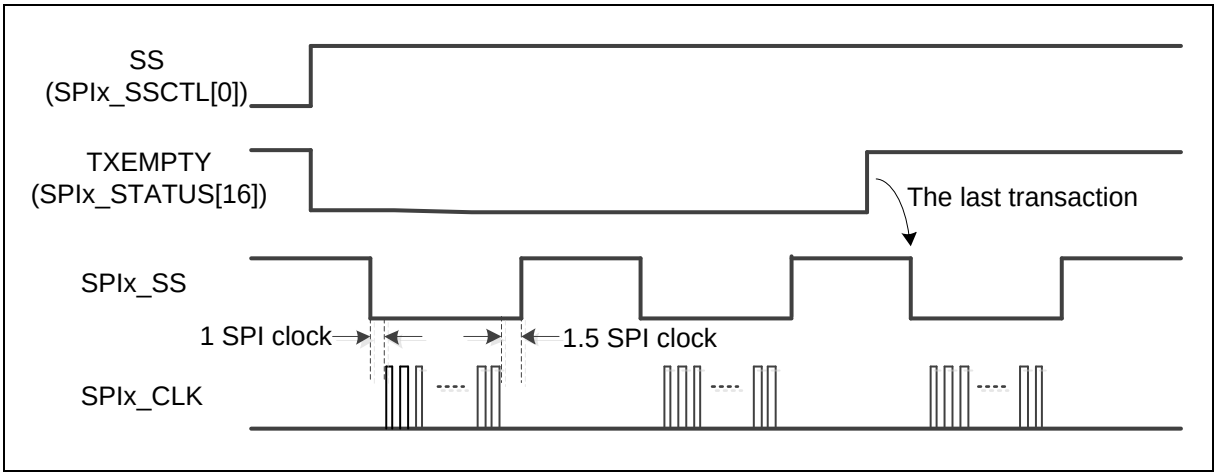


Figure 6.14-6 Automatic Slave Selection (SSACTPOL = 0, SUSPITV > 0x2)

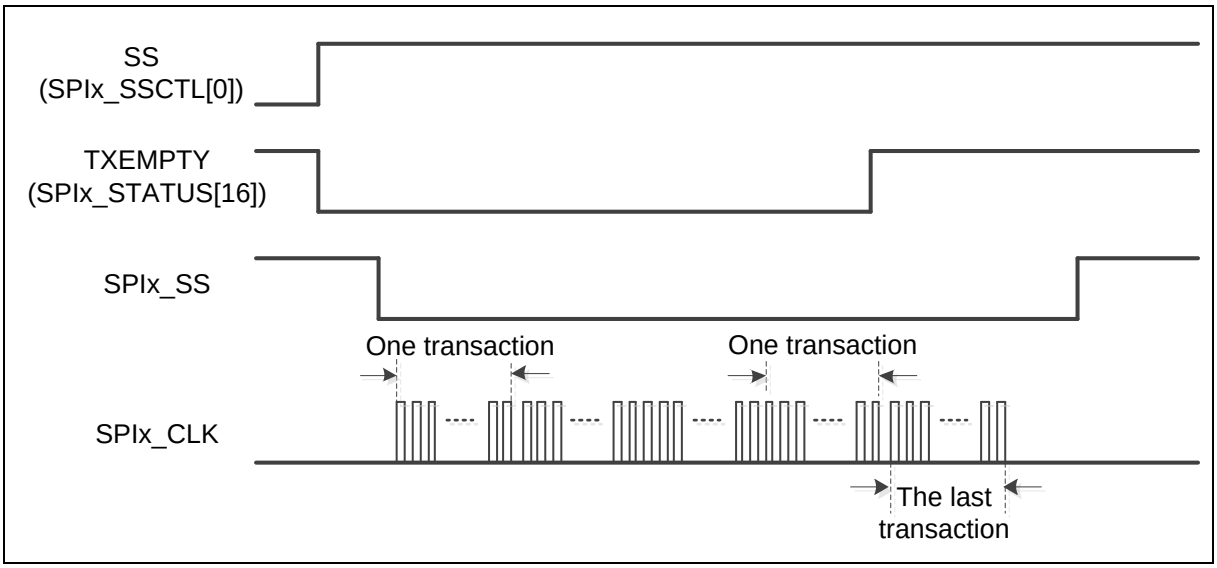


Figure 6.14-7 Automatic Slave Selection (SSACTPOL = 0, SUSPITV < 0x3)

6.14.5.3 Byte Reorder and Suspend Function

When the transfer is set as MSB first (LSB = 0) and the REORDER (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 Byte3. If the DWIDTH is set as 24-bit transfer mode, the data in TX buffer and RX buffer will be rearranged as [unknown byte, Byte0, Byte1, Byte2]. The SPI controller will transmit/receive data with the sequence of Byte0, Byte1 and then Byte2. Each byte will be transmitted/received with MSB first. The rule of 16-bit mode is the same as above. Byte Reorder function is only available when DWIDTH is configured as 16, 24, and 32 bits.

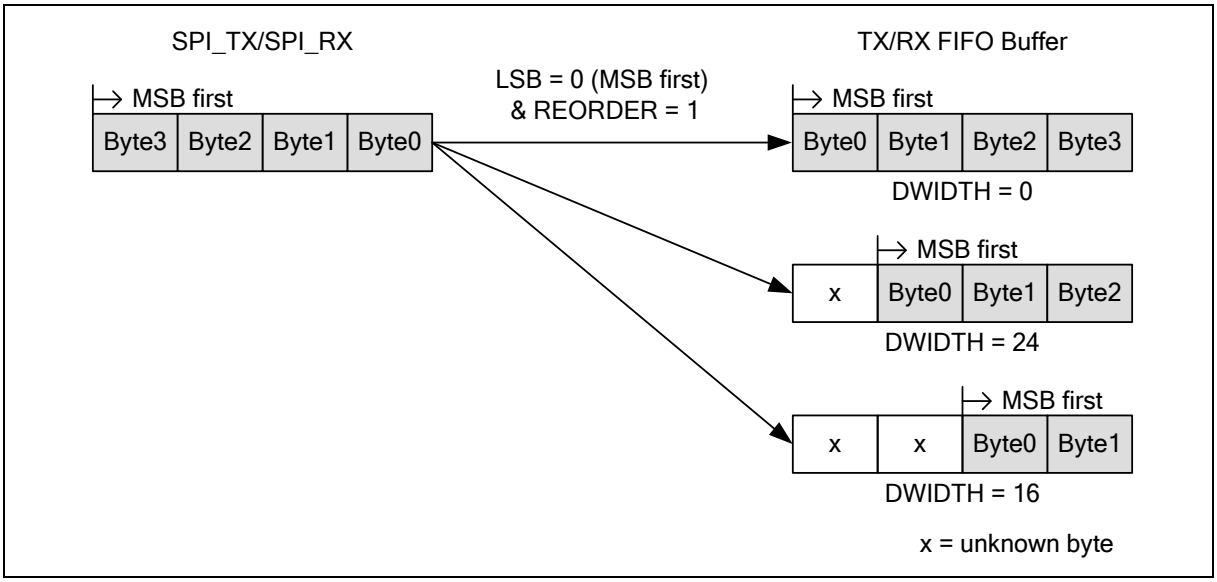


Figure 6.14-8 Byte Reorder Function

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]).

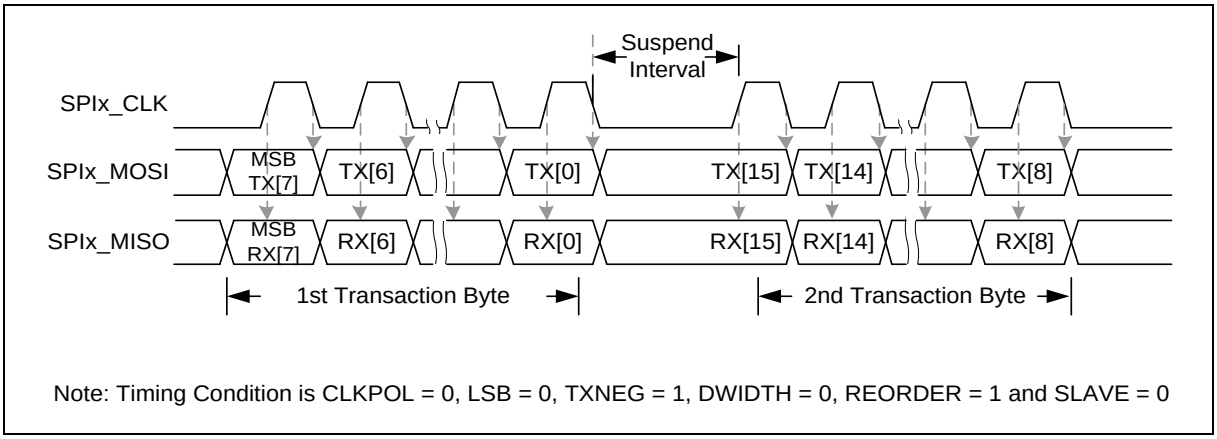


Figure 6.14-9 Timing Waveform for Byte Suspend

6.14.5.4 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.

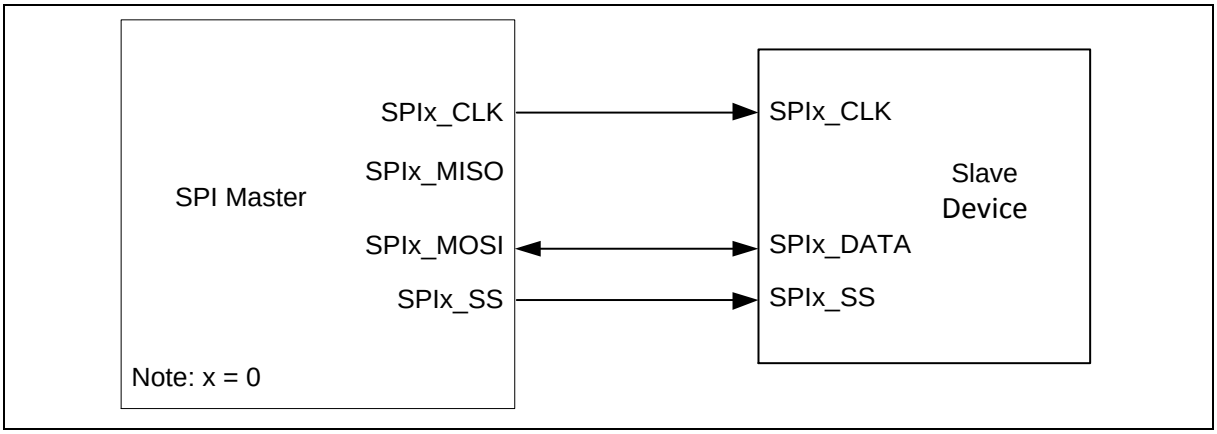


Figure 6.14-10 SPI Half-Duplex Master Mode Application Block Diagram

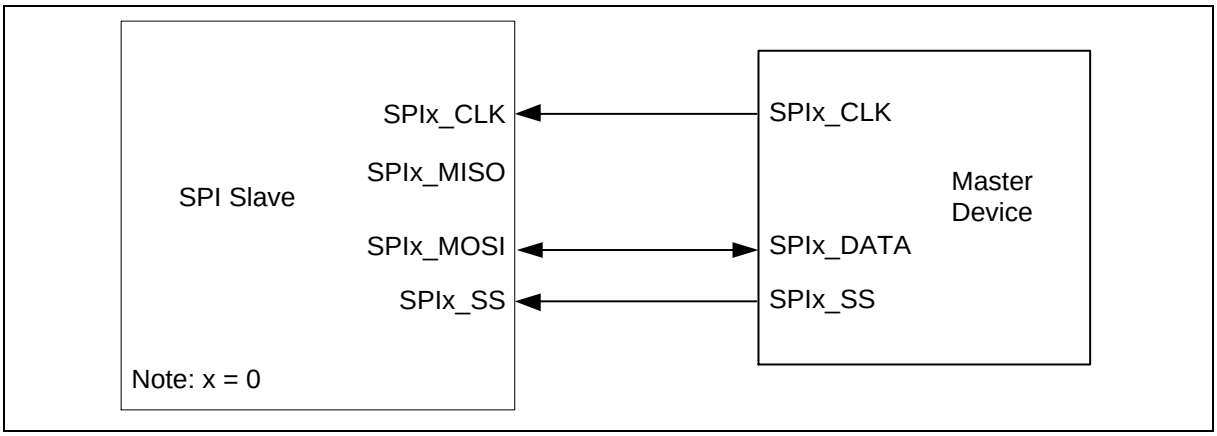


Figure 6.14-11 SPI Half-Duplex Slave Mode Application Block Diagram

6.14.5.5 Receive-Only Mode

In SPI Master device, it can communicate in receive-only mode by setting RXONLY (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]) enable, 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.

6.14.5.6 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.

When PDMA transfer is done, user needs to disable TXPDMAEN/RXPDMAEN. After re-setting control registers of PDMA, user enables TXPDMAEN/RXPDMAEN again.

Note: The SPI supports single request PDMA (Read/Write) only, burst request PDMA is not supported.

6.14.5.7 FIFO Buffer Operation

The SPI controllers are equipped 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.

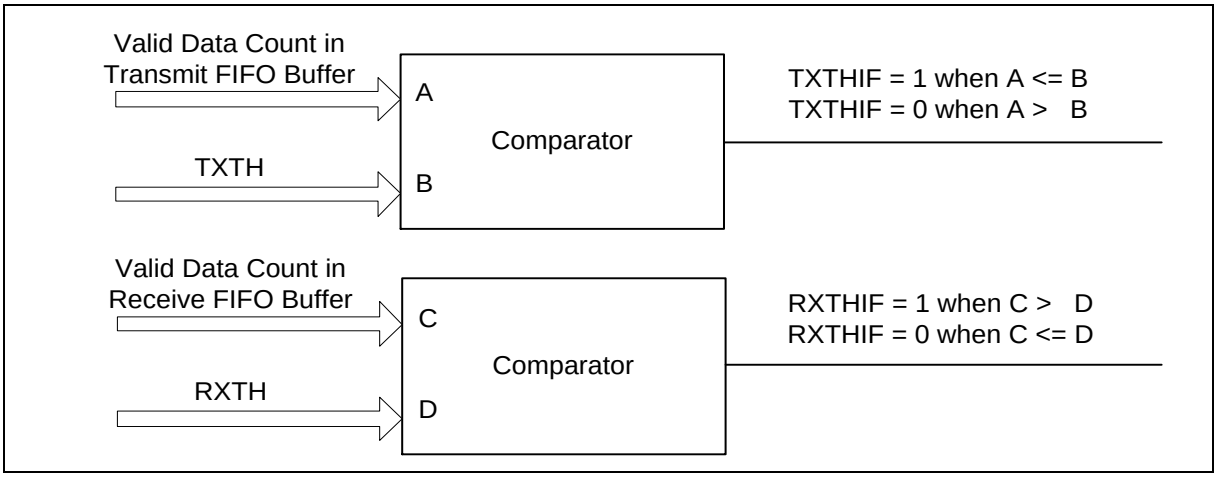


Figure 6.14-12 FIFO Threshold Comparator

In Master mode, when the first datum is written to the SPIx_TX register, the TXEMPTY (SPIx_STATUS[16]) flag 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.

When user sets DWIDTH (SPIx_CTL[12:8]) to 8 bits ~ 16 bits, the FIFO structure will be configured to eight 16-bit wide transmit and receive FIFO buffers automatically.

The example 1 of Figure 6.14-13 indicates the updated condition of TXEMPTY (SPIx_STATUS[16]) and the relationship among the FIFO buffer, shift register and the skew buffer for 8~16 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 to 1. The Data0 in shift register will be shift into skew buffer by bit for transmission until the transfer is done.

The Example 2 of Figure 6.14-13 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 TXFULL = 1.

The example 1 of Figure 6.14-14 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 6.14-14 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 TXFULL = 1.

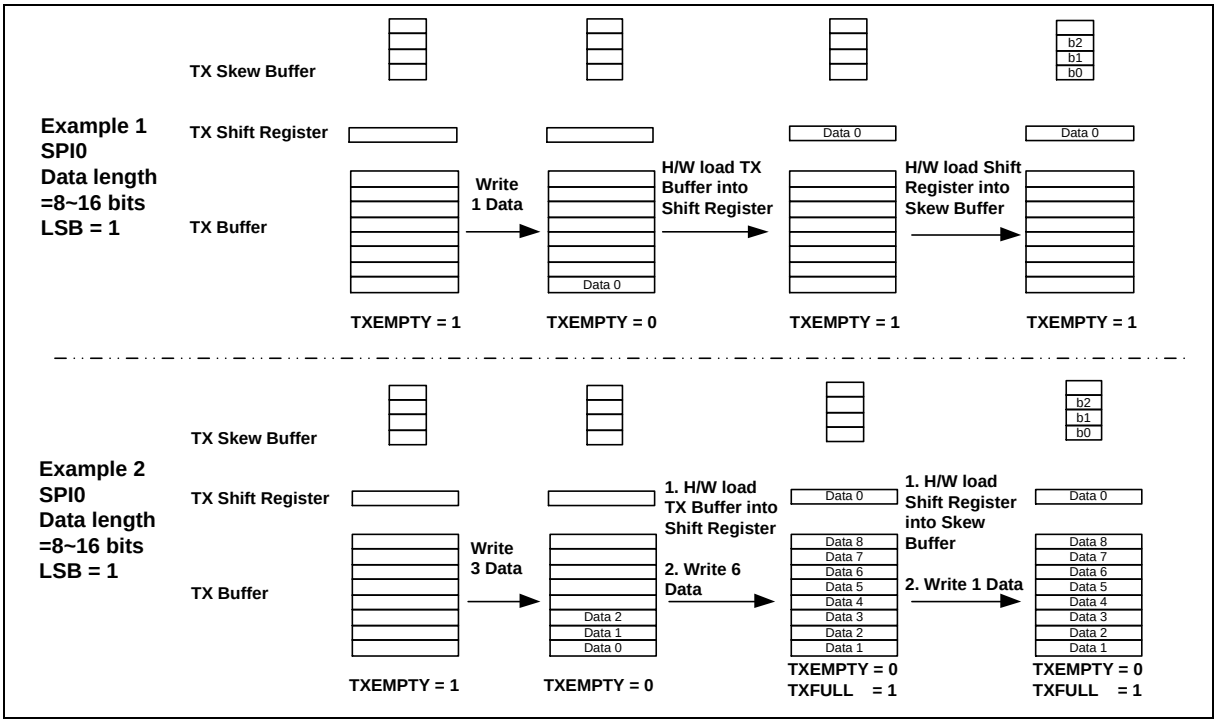


Figure 6.14-13 Transmit FIFO Buffer Example for 8~16 Bits of Data Length

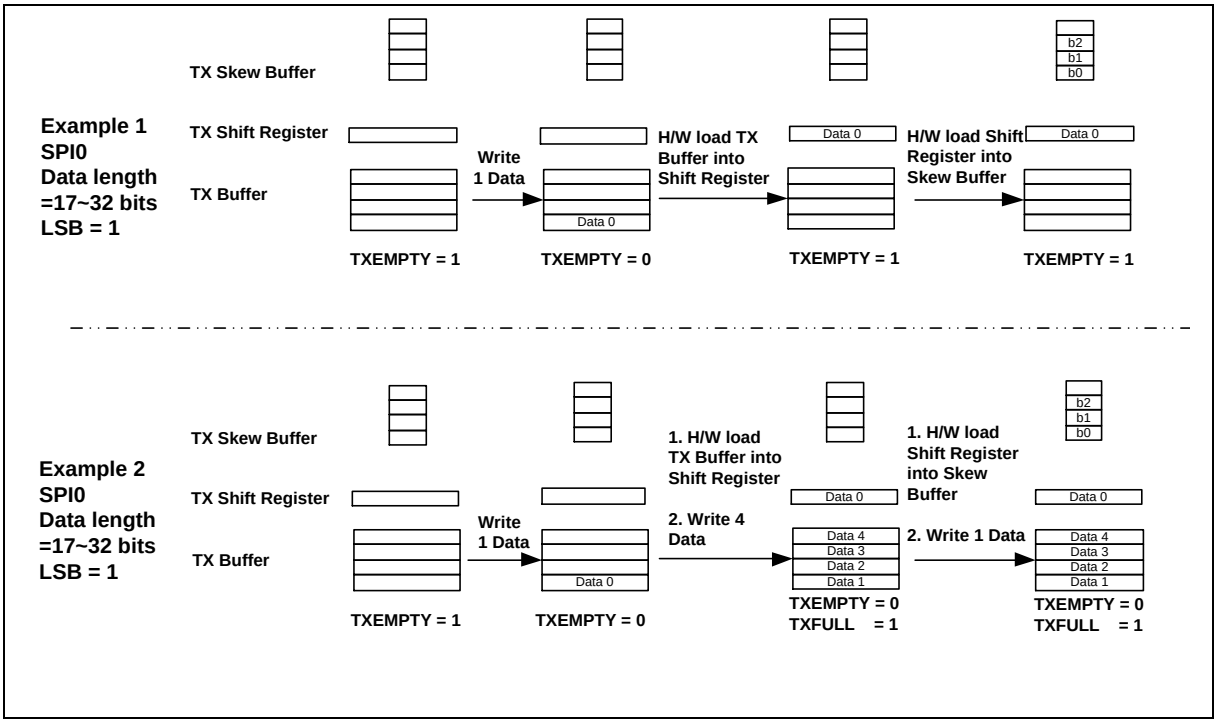


Figure 6.14-14 Transmit FIFO Buffer Example for 17~32 Bits of Data Length

The subsequent transactions will be triggered automatically if the transmitted data are updated in time. If the SPIx_TX register does not be updated after all data transfer are done, the transfer will stop.

In Master mode, during receiving operation, the serial data are received from 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 6.14-15). 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 6.14-15).

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 6.14-16). 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 6.14-16).

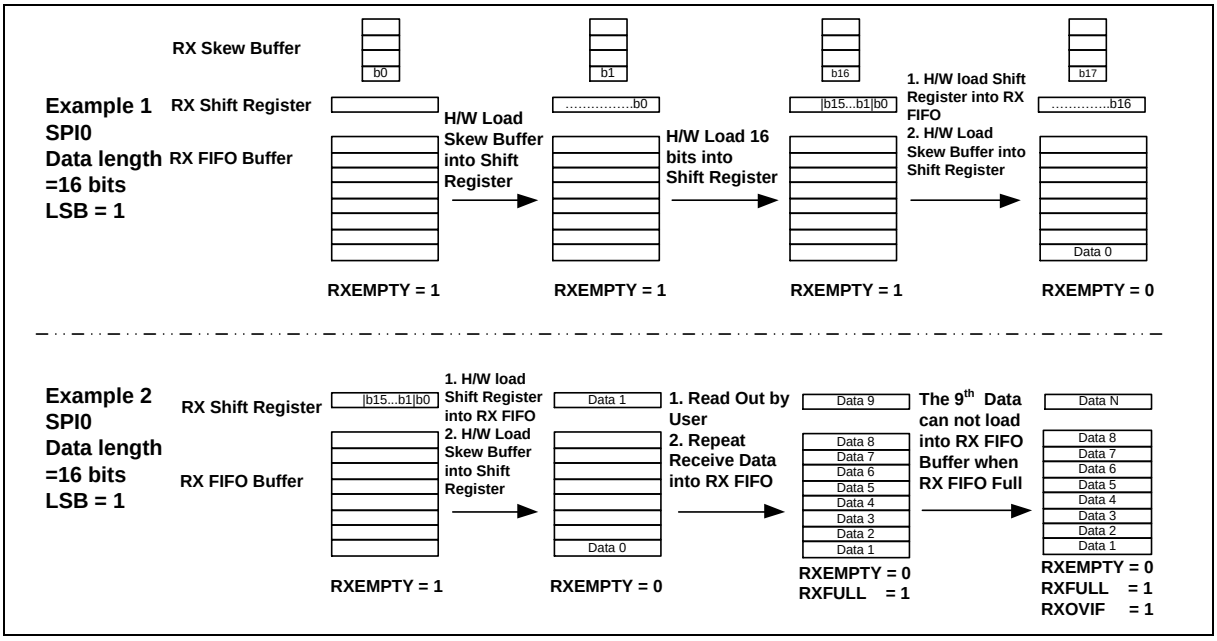


Figure 6.14-15 Receive FIFO Buffer Example for 16 Bits of Data Length

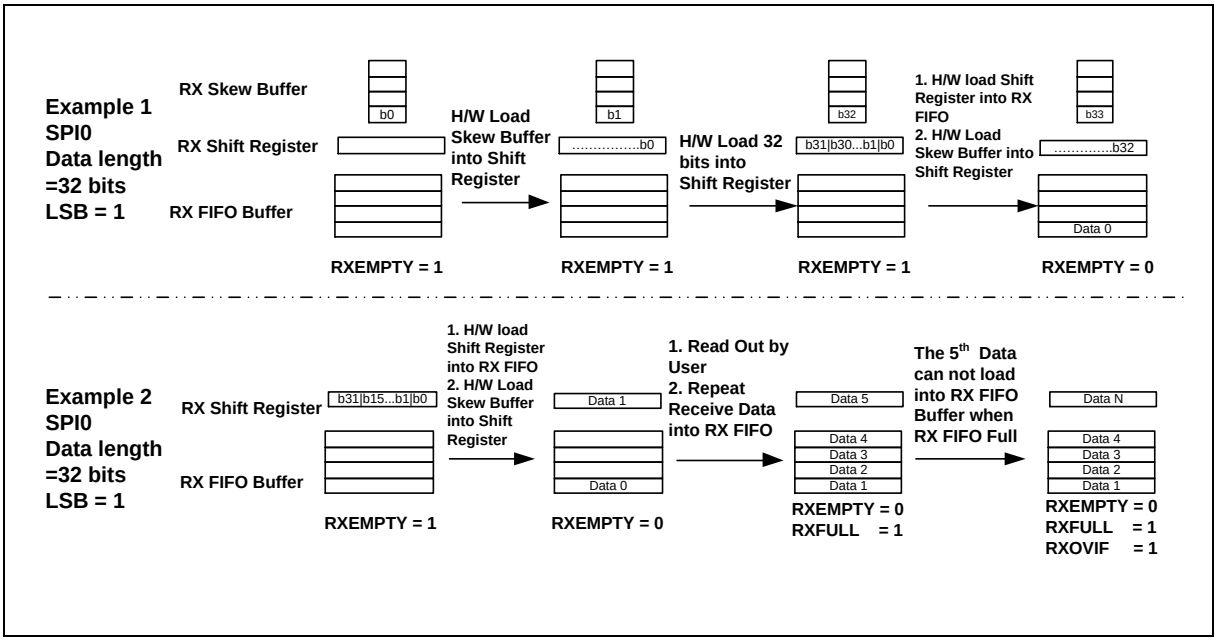


Figure 6.14-16 Receive FIFO Buffer Example for 32 Bits of Data Length

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 any 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.

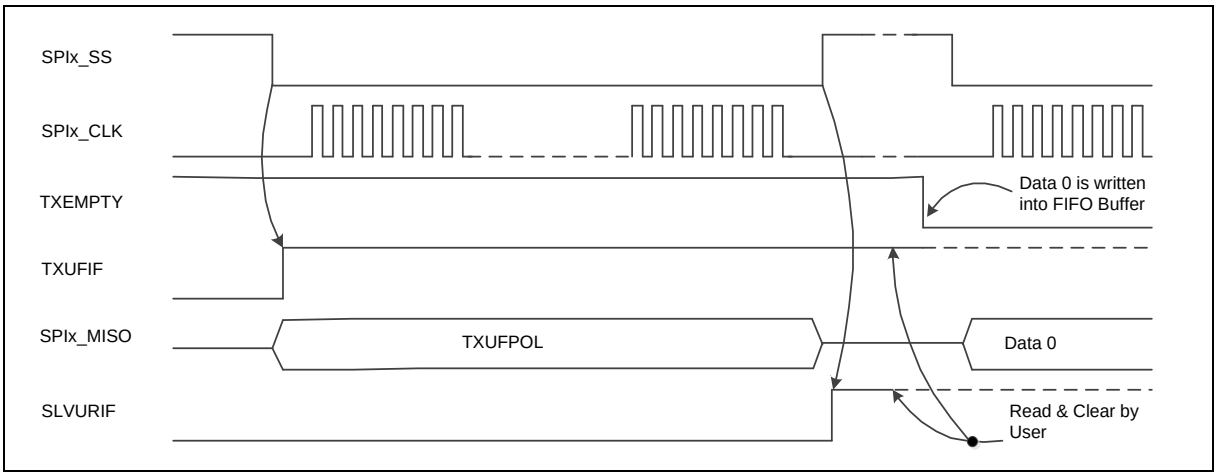


Figure 6.14-17 TX Underflow Event and Slave Under Run Event

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 (refer to the Receive FIFO Buffer Example figure). 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.

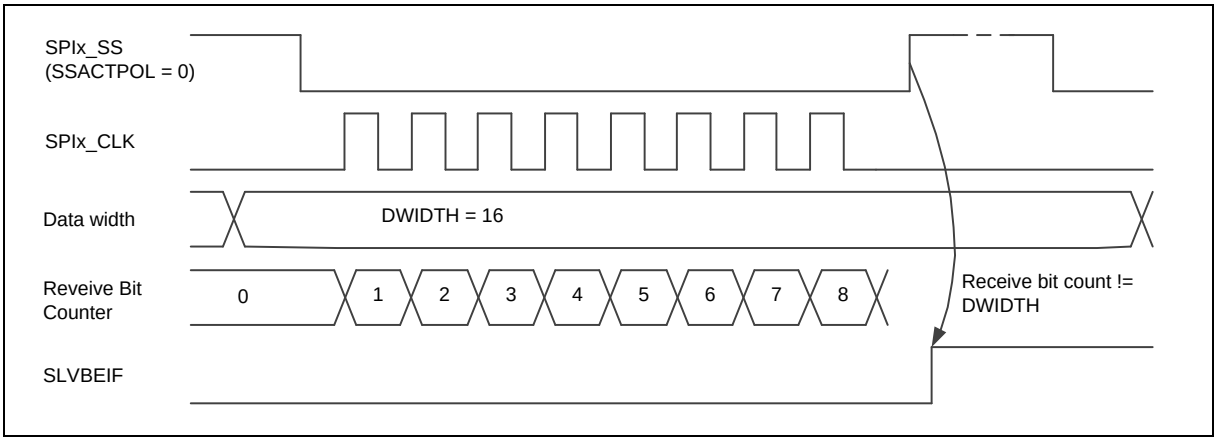


Figure 6.14-18 Slave Mode Bit Count Error

A receive time-out function is built-in in this controller. When the receive FIFO is not empty and no read operation in receive FIFO over 64 SPI 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.

6.14.5.8 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 and RX shift registers. 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.

- Receive Overrun interrupt

  In Slave mode, 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 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.

6.14.5.9 I²S Mode

The SPI0 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.

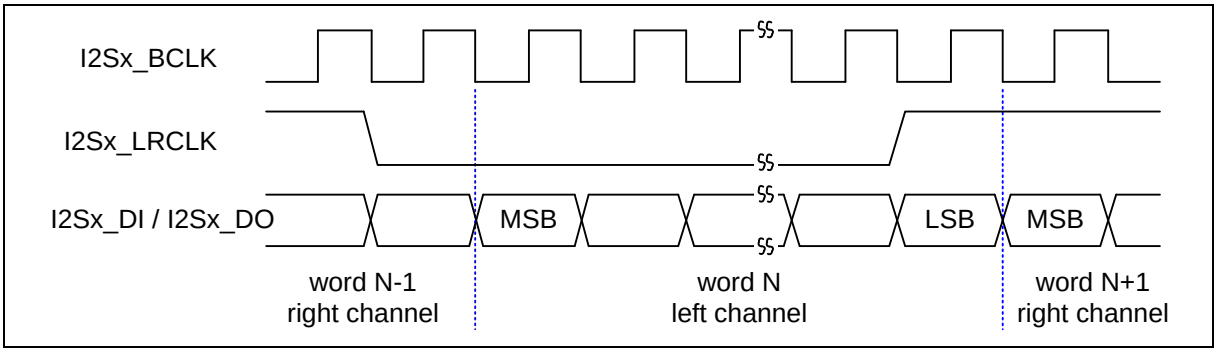


Figure 6.14-19 I²S Data Format Timing Diagram

In MSB justified data format, the MSB is sent and latched on the first clock of an audio channel.

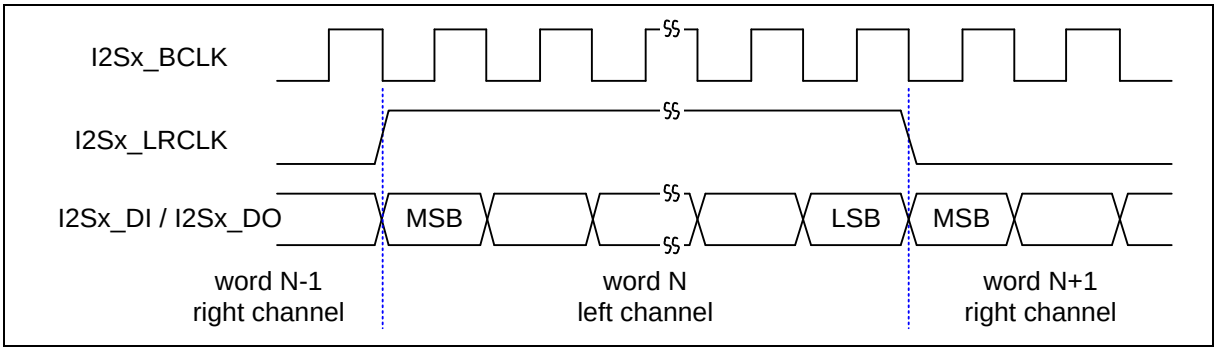


Figure 6.14-20 MSB Justified Data Format Timing Diagram

The I2Sx_LRCLK signal also supports PCM mode A and PCM mode B. The I2Sx_LRCLK signal in PCM mode indicates the beginning of an audio frame.

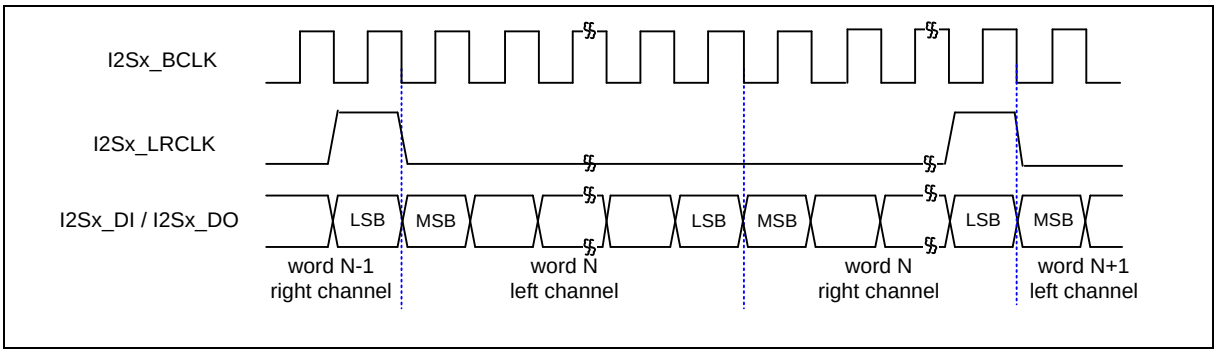


Figure 6.14-21 PCM Mode A Timing Diagram

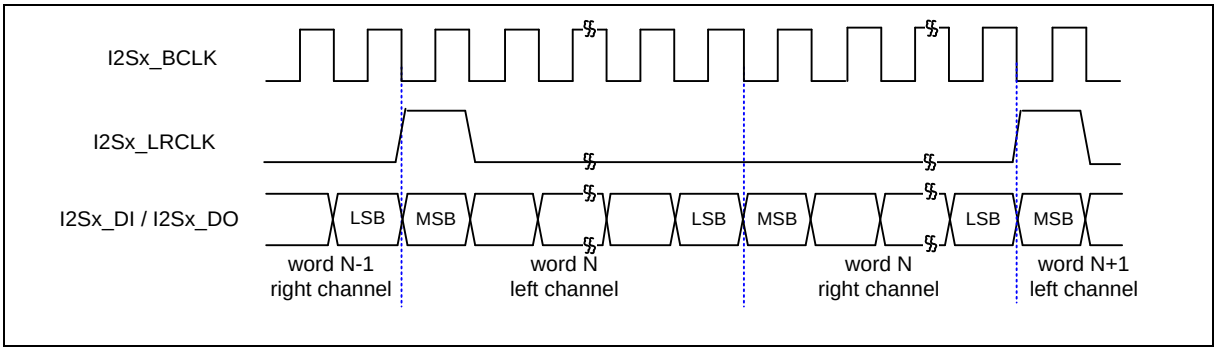


Figure 6.14-22 PCM Mode B Timing Diagram

6.14.5.10 I²S Mode FIFO Operation

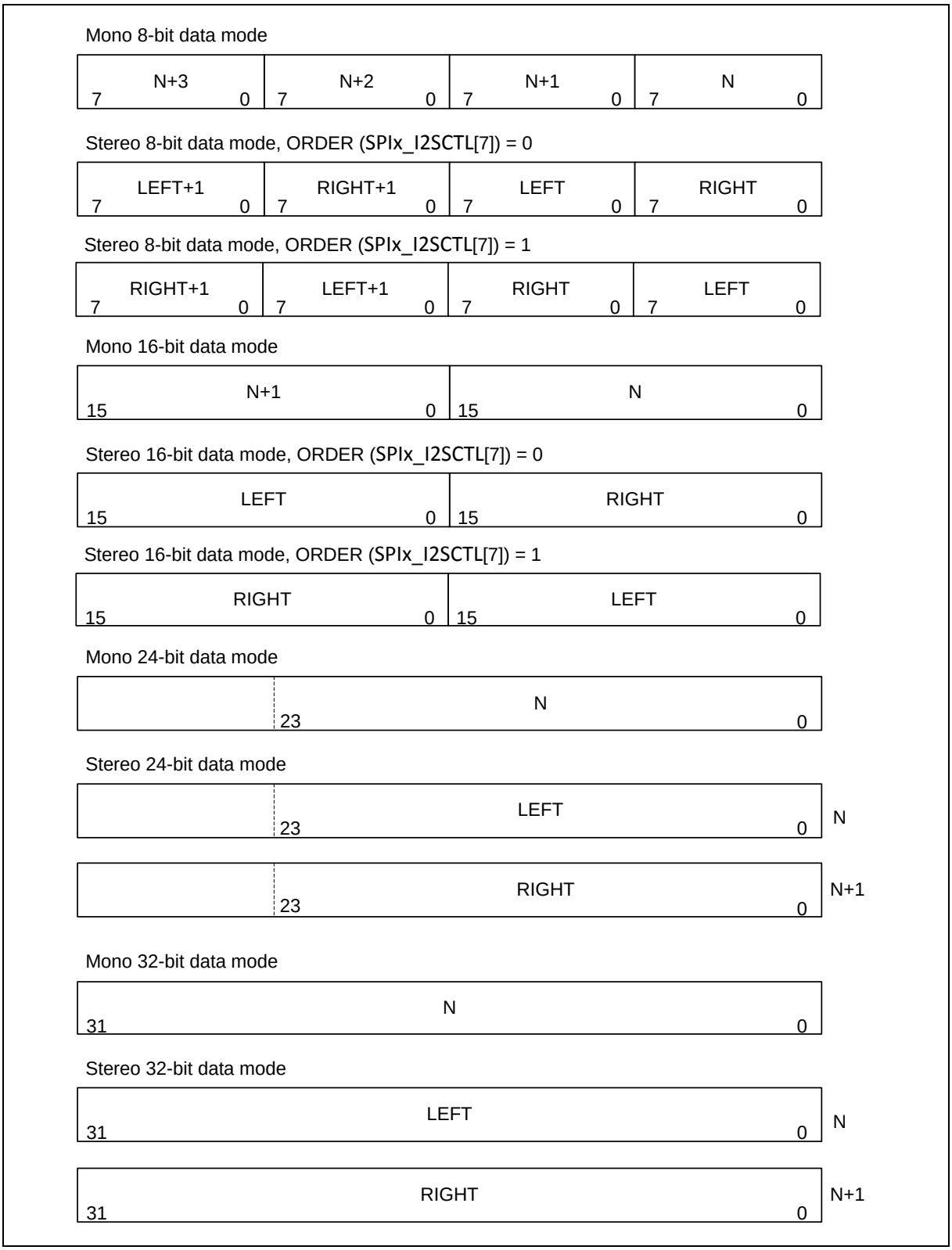


Figure 6.14-23 FIFO Contents for Various I²S Modes

6.14.5.11 Dummy Data Number for I²S / PCM Master mode and Monaural Mode

Before I²S / PCM master starts to send TX data to external slave device, we set 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 external slave device, master will send TX FIFO data to external slave device. Table 6.14-2 shows 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])	Dummy Data Number (Unit = L Channel + R Channel)
8 bits	0
16 bits	0
24 bits	1
32 bits	1

Table 6.14-2 Dummy Data Number for I²S / PCM Master Mode and Monaural Mode

6.14.5.12 Dummy Data Number for I²S / PCM Master mode and Stereo Mode

Before I²S / PCM master starts to send TX data to external slave device, we set 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 external slave device, master will send TX FIFO data to external slave device. Table 6.14-3 shows 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])	Dummy Data Number (Unit = L Channel + R Channel)
8 bits	0
16 bits	0
24 bits	1
32 bits	1

Table 6.14-3 Dummy Data Number for I²S / PCM Master Mode and Stereo Mode

6.14.5.13 Dummy Data Number for I²S Slave mode and Monaural Mode

Before I²S / PCM slave starts to send TX data to external master device, we set 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 external master device, slave will send TX FIFO data to external master device. Table 6.14-4 shows 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])	Dummy Data Number (Unit = L Channel + R Channel)
8 bits	3
16 bits	2
24 bits	2
32 bits	2

Table 6.14-4 Dummy Data Number for I²S Slave Mode and Monaural Mode

6.14.5.14 Dummy Data Number for PCM Slave mode and Monaural Mode

Before I²S / PCM slave starts to send TX data to external master device, we set 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 external master device, slave will send TX FIFO data to external master device. Table 6.14-5 shows 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])	Dummy Data Number (Unit = L Channel + R Channel)
8 bits	2
16 bits	1
24 bits	1
32 bits	1

Table 6.14-5 Dummy Data Number for PCM Slave Mode and Monaural Mode

6.14.5.15 Dummy Data Number for I²S Slave mode and Stereo Mode

Before I²S / PCM slave starts to send TX data to external master device, we set 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 external master device, slave will send TX FIFO data to external master device. Table 6.14-6 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])	Dummy Data Number (Unit = L Channel + R Channel)
8 bits	3
16 bits	2
24 bits	2
32 bits	2

Table 6.14-6 Dummy Data Number for I²S Slave Mode and Stereo Mode

6.14.5.16 Dummy Data Number for PCM Slave mode and Stereo Mode

Before I²S / PCM slave starts to send TX data to external master device, we set 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 external master device, slave will send TX FIFO data to external master device. Table 6.14-7 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])	Dummy Data Number (Unit = L Channel + R Channel)
8 bits	2
16 bits	1
24 bits	1
32 bits	1

Table 6.14-7 Dummy Data Number for PCM Slave Mode and Stereo Mode

6.14.6 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.

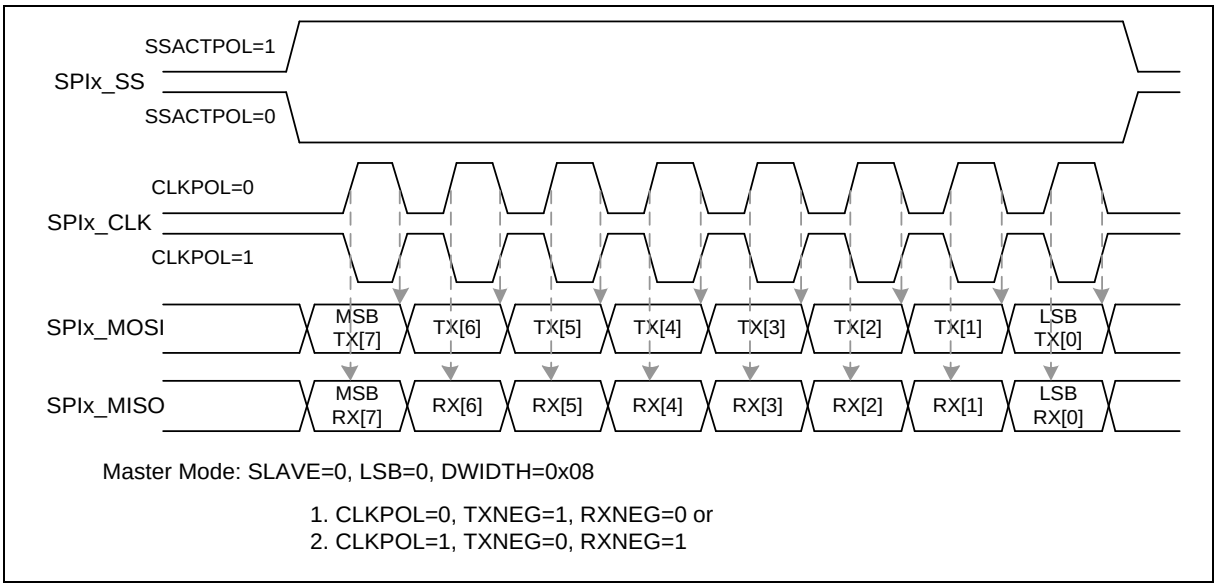


Figure 6.14-24 SPI Timing in Master Mode

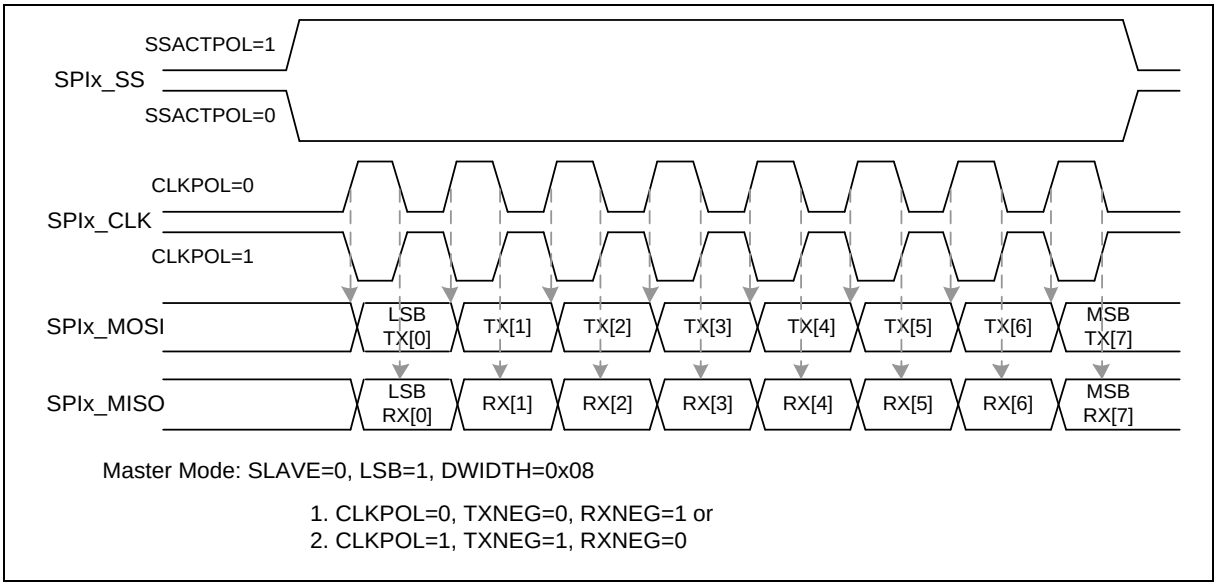


Figure 6.14-25 SPI Timing in Master Mode (Alternate Phase of SPIx_CLK)

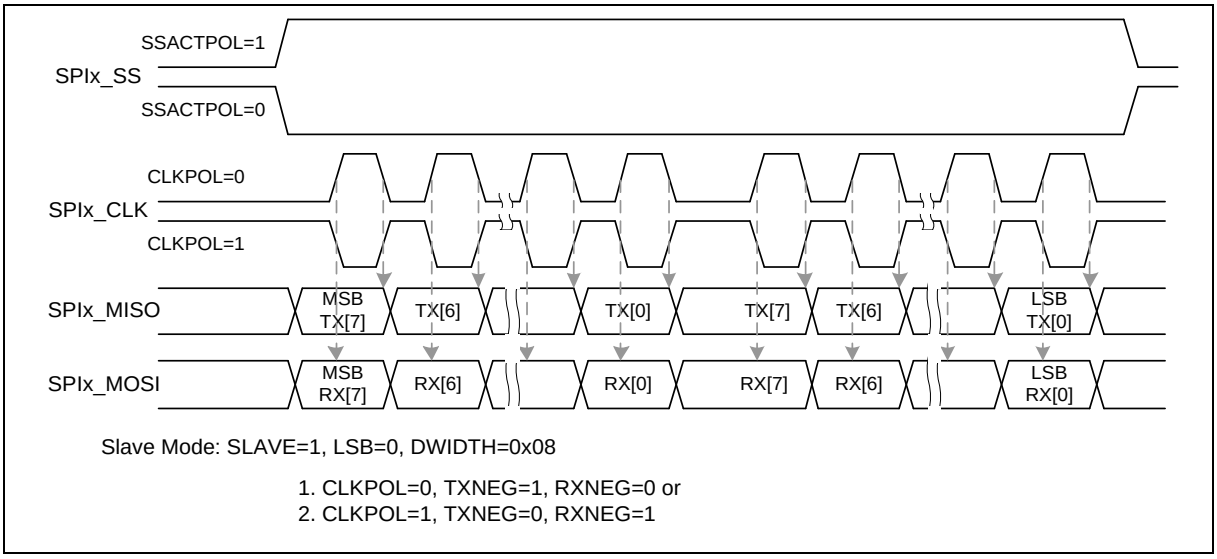


Figure 6.14-26 SPI Timing in Slave Mode

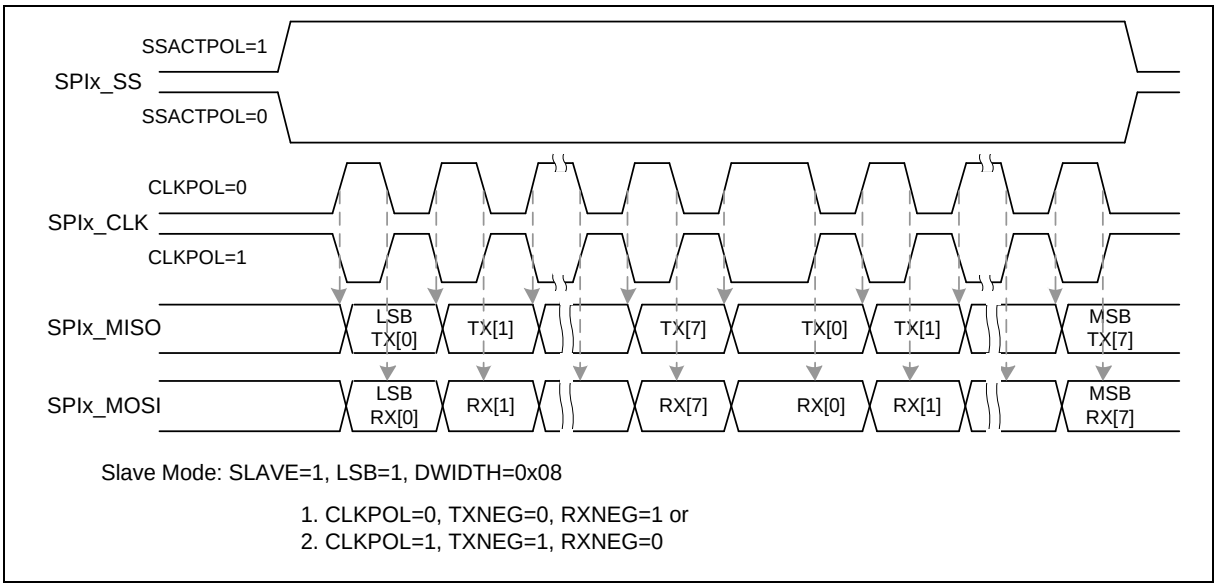


Figure 6.14-27 SPI Timing in Slave Mode (Alternate Phase of SPIx_CLK)

6.14.7 Programming Examples

Example 1:

The SPI controller is set as a full-duplex master to access an off-chip slave device with the following specifications:

- Data bit is latched on positive edge of SPI bus clock.
- Data bit is driven on negative edge of SPI bus clock.
- Data is transferred from MSB first.
- SPI bus clock is idle at low state.
- Only one byte of data to be transmitted/received in a transaction.
- Uses the first SPI slave select pin to connect with an off-chip slave device. The slave selection signal is active low.

The operation flow is as follows:

1. Set DIVIDER (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).

6) Set LSB transfer first by setting LSB (SPIx_CTL[13]) to 1.

4. 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.
5. 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.
6. 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.
7. 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]).
8. Read out the received one byte data from SPIx_RX register.
9. Go to 5 to continue another data transfer or stop data transfer.

6.14.8 Register Map

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

Register	Offset	R/W	Description	Reset Value
SPI Base Address: **SPIx_BA = 0x4006_1000**				
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_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

6.14.9 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 I^2S mode.

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			DATDIR	REORDER	SLAVE	UNITIEN	Reserved
15	14	13	12	11	10	9	8
RXONLY	HALFDPX	LSB	DWIDTH				
7	6	5	4	3	2	1	0
SUSPITV				CLKPOL	TXNEG	RXNEG	SPIEN

Bits	Description	
[31:21]	**Reserved**	Reserved.
[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 (Master Only)** This bit field is only available in Master mode. In receive-only mode, SPI Master will generate SPI bus clock continuously for receiving data bit from SPI slave device and assert the BUSY status. 0 = Receive-only mode Disabled. 1 = Receive-only mode Enabled.
[14]	**HALFDPX**	**SPI Half-duplex Transfer Enable Bit** This bit is used to select full-duplex or half-duplex for SPI transfer. The bit field DATDIR

		(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 8 bits and can be up to 32 bits. DWIDTH = 0x08 …. 8 bits. DWIDTH = 0x09 …. 9 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 (Master Only)** The four bits provide configurable suspend interval between two successive transmit/receive transaction in a transfer. The definition of the suspend interval is the interval between the last clock edge of the preceding transaction word and the first clock edge of the following transaction word. The default value is 0x3. The period of the suspend interval is obtained according to the following equation. (SUSPITV[3:0] + 0.5) * period of SPICLK clock cycle Example: SUSPITV = 0x0 …. 0.5 SPICLK clock cycle. SUSPITV = 0x1 …. 1.5 SPICLK clock cycle. …… SUSPITV = 0xE …. 14.5 SPICLK clock cycle. SUSPITV = 0xF …. 15.5 SPICLK clock cycle.
[3]	**CLKPOL**	**Clock Polarity** 0 = SPI bus clock is idle low. 1 = SPI bus clock is idle high.
[2]	**TXNEG**	**Transmit on Negative Edge** 0 = Transmitted data output signal is changed on the rising edge of SPI bus clock. 1 = Transmitted data output signal is changed on the falling edge of SPI bus clock.
[1]	**RXNEG**	**Receive on Negative Edge** 0 = Received data input signal is latched on the rising edge of SPI bus clock. 1 = Received data input signal is latched on the falling edge of SPI bus clock.
[0]	**SPIEN**	**SPI Transfer Control Enable Bit** In Master mode, the transfer will start when there is data in the FIFO buffer after this bit is set to 1. In Slave mode, this device is ready to receive data when this bit is set to 1. 0 = Transfer control Disabled. 1 = Transfer control Enabled. **Note:** Before changing the configurations of 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 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				**AUTOSS**	**SSACTPOL**	**Reserved**	**SS**

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13]	**SSINAIEN**	**Slave Select Inactive Interrupt Enable Bit** 0 = Slave select inactive interrupt Disabled. 1 = Slave select inactive interrupt Enabled.
[12]	**SSACTIEN**	**Slave Select Active Interrupt Enable Bit** 0 = Slave select active interrupt Disabled. 1 = Slave select active interrupt Enabled.
[11:10]	**Reserved**	Reserved.
[9]	**SLVURIEN**	**Slave Mode TX Under Run Interrupt Enable Bit** 0 = Slave mode TX under run interrupt Disabled. 1 = Slave mode TX under run interrupt Enabled.
[8]	**SLVBEIEN**	**Slave Mode Bit Count Error Interrupt Enable Bit** 0 = Slave mode bit count error interrupt Disabled. 1 = Slave mode bit count error interrupt Enabled.
[7:4]	**Reserved**	Reserved.
[3]	**AUTOSS**	**Automatic Slave Selection Function Enable Bit (Master Only)** 0 = Automatic slave selection function Disabled. Slave selection signal will be asserted/de-asserted according to SS (SPIx_SSCTL[0]). 1 = Automatic slave selection function Enabled.
[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 (Master Only)** 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]).

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. **Note:** In SPI Master mode with full duplex transfer, if both TX and RX PDMA functions are enabled, RX PDMA function cannot be enabled prior to TX PDMA function. User can enable TX PDMA function firstly or enable both functions simultaneously.

SPI FIFO Control Register (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						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 8~16 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 8~16 bits of data length.
[23:10]	**Reserved**	Reserved.
[9]	**TXFBCLR**	**Transmit FIFO Buffer Clear** 0 = No effect. 1 = Clear transmit FIFO pointer. The TXFULL bit will be cleared to 0 and the TXEMPTY bit will be set to 1. This bit will be cleared to 0 by hardware about 1 system clock after it is set to 1. **Note:** The TX shift register will not be cleared.
[8]	**RXFBCLR**	**Receive FIFO Buffer Clear** 0 = No effect. 1 = Clear receive FIFO pointer. The RXFULL bit will be cleared to 0 and the RXEMPTY bit will be set to 1. This bit will be cleared to 0 by hardware about 1 system clock after it is set to 1. **Note:** The RX shift register will not be cleared.
[7]	**TXUFIEN**	**TX Underflow Interrupt Enable Bit** When TX underflow event occurs in Slave mode, TXUFIF (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 I^2S mode.

31	30	29	28	27	26	25	24
TXCNT				RXCNT			
23	22	21	20	19	18	17	16
TXRXRST	Reserved			TXUFIF	TXTHIF	TXFULL	TXEMPTY
15	14	13	12	11	10	9	8
SPIENSTS	Reserved		RXTOIF	RXOVIF	RXTHIF	RXFULL	RXEMPTY
7	6	5	4	3	2	1	0
SLVURIF	SLVBEIF	Reserved	SSLINE	SSINAIF	SSACTIF	UNITIF	BUSY

Bits	Description	
[31:28]	**TXCNT**	**Transmit FIFO Data Count (Read Only)** This bit field indicates the valid data count of transmit FIFO buffer.
[27:24]	**RXCNT**	**Receive FIFO Data Count (Read Only)** This bit field indicates the valid data count of receive FIFO buffer.
[23]	**TXRXRST**	**TX or RX Reset Status (Read Only)** 0 = The reset function of TXRST or RXRST is done. 1 = Doing the reset function of TXRST or RXRST. **Note:** Both the reset operations of TXRST and RXRST need 3 system clock cycles + 2 peripheral clock cycles. User can check the status of this bit to monitor the reset function is doing or done.
[22:20]	**Reserved**	Reserved.
[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.

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 Clock Loss Interrupt Enable Bit for Slave Mode** Interrupt occurs if this bit is set to 1 and bit clock loss 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 left channel data sign bit changes or next shift data bits are all 0 then 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 right channel data sign bit change or next shift data bits are all 0 then 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							
23	22	21	20	19	18	17	16
Reserved						BCLKDIV	
15	14	13	12	11	10	9	8
BCLKDIV							
7	6	5	4	3	2	1	0
Reserved	MCLKDIV						

Bits	Description	
[31:18]	**Reserved**	Reserved.
[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,. $f_{MCLK} = \frac{f_{i2s\_clock\_src}}{2 \times \text{MCLKDIV}}$ If MCLKDIV = 0,. $f_{MCLK} = f_{i2s\_clock\_src}$ where $f_{i2s\_clock\_src}$ is the frequency of I²S peripheral clock source, which is defined in the clock control register CLK_CLKSEL2. In general, the master clock rate is 256 times sampling clock rate.

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 Clock Loss Interrupt Flag for Slave Mode** 0 = No bit clock loss event occurred. 1 = Bit clock loss 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.

6.15 Quad Serial Peripheral Interface (QSPI)

6.15.1 Overview

The Quad Serial Peripheral Interface (QSPI) 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 one QSPI controller performing a serial-to-parallel conversion on data received from a peripheral device, and a parallel-to-serial conversion on data transmitted to a peripheral device.

The QSPI controller supports 2-bit Transfer mode to perform full-duplex 2-bit data transfer and also supports Dual and Quad I/O Transfer mode and the controller supports the PDMA function to access the data buffer.

6.15.2 Features

- Supports one QSPI controller
- Supports Master or Slave mode operation
- Master mode up to 24 MHz and Slave mode up to 16 MHz (when chip works at V_{DD} = 1.8~3.6V)
- Supports 2-bit Transfer mode
- Supports Dual and Quad I/O Transfer mode
- Configurable bit length of a transaction word from 8 to 32-bit
- Provides separate 8-level depth transmit and receive FIFO buffers
- Supports MSB first or LSB first transfer sequence
- Supports 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

6.15.3 Block Diagram

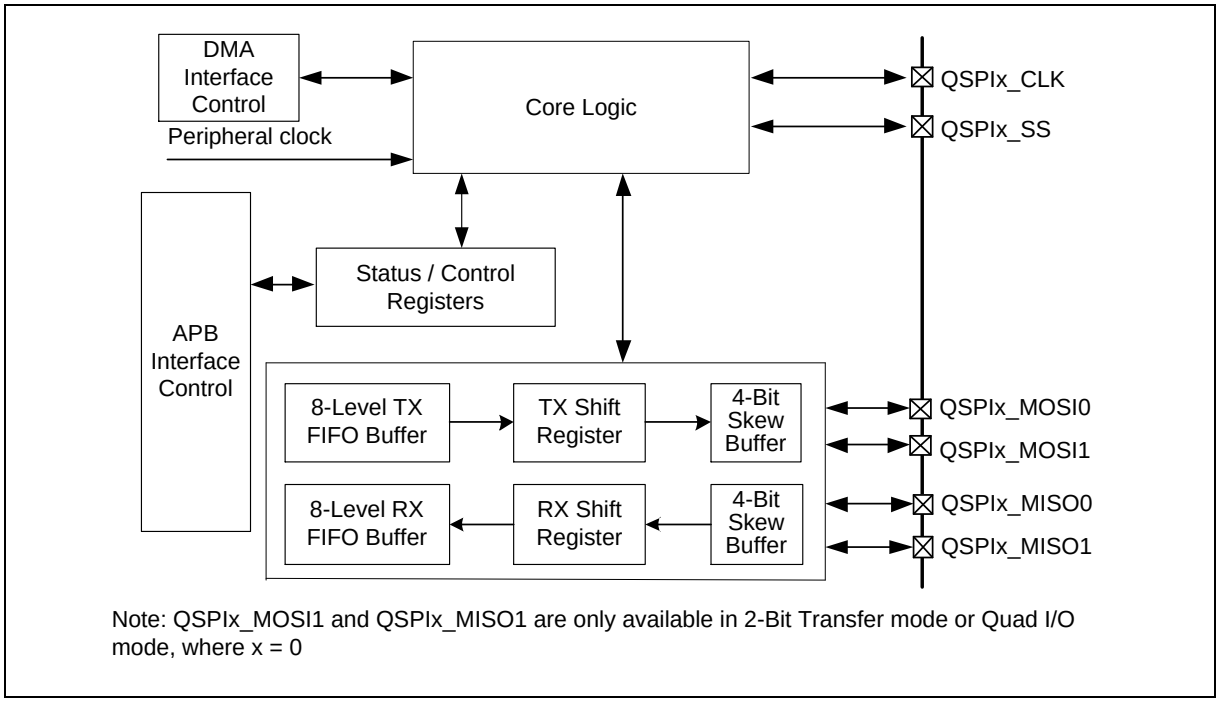


Figure 6.15-1 QSPI Block Diagram

TX FIFO Buffer:

The transmit FIFO buffer is a 8-level depth, 32-bit wide, first-in, first-out register buffer. The data can be written to the transmit FIFO buffer in advance through software by writing the QSPIx_TX register.

RX FIFO Buffer:

The receive FIFO buffer is also a 8-level depth, 32-bit wide, first-in, first-out register buffer. The receive control logic will store the receive data to this buffer. The FIFO buffer data can be read from QSPIx_RX register by software.

TX Shift Register:

The transmit shift register is a 32-bit wide register buffer. The transmit data is loaded from the TX FIFO buffer and shifted out bit-by-bit to the skew buffer.

RX Shift Register:

The receive shift register is also a 32-bit wide register buffer. The receive data is shift in bit-by-bit from the skew buffer and is loaded into RX FIFO buffer when a transaction done.

Skew Buffer:

The skew buffer is a 4-level 1-bit buffer. There are two skew buffers in transmitting and received side. In received side, it is used to shift bits into RX shift register from QSPI bus. In transmitting side, it is used to shift bits into QSPI bus from TX shift register.

6.15.4 Basic Configuration

6.15.4.1 QSPI0 Basic Configuration

- Clock source Configuration
  - Select the source of QSPI0 peripheral clock on QSPI0SEL (CLK_CLKSEL2[3:2]).
  - Enable QSPI0 peripheral clock in QSPI0CKEN (CLK_APBCLK0[12]).
- Reset Configuration
  - Reset QSPI0 controller in QSPI0RST (SYS_IPRST1[12]).

6.15.5 Functional Description

6.15.5.1 Terminology

QSPI Peripheral Clock and QSPI Bus Clock

The QSPI controller needs the peripheral clock to drive the QSPI logic unit to perform the data transfer. The peripheral clock rate is determined by the settings of clock divisor (QSPIx_CLKDIV) and the clock source which can be HXT, PLL, PCLK or HIRC. QSPIxSEL of CLK_CLKSEL2 register determines the clock source of the peripheral clock. The DIVIDER (QSPIx_CLKDIV[8:0]) setting determines the divisor of the clock rate calculation.

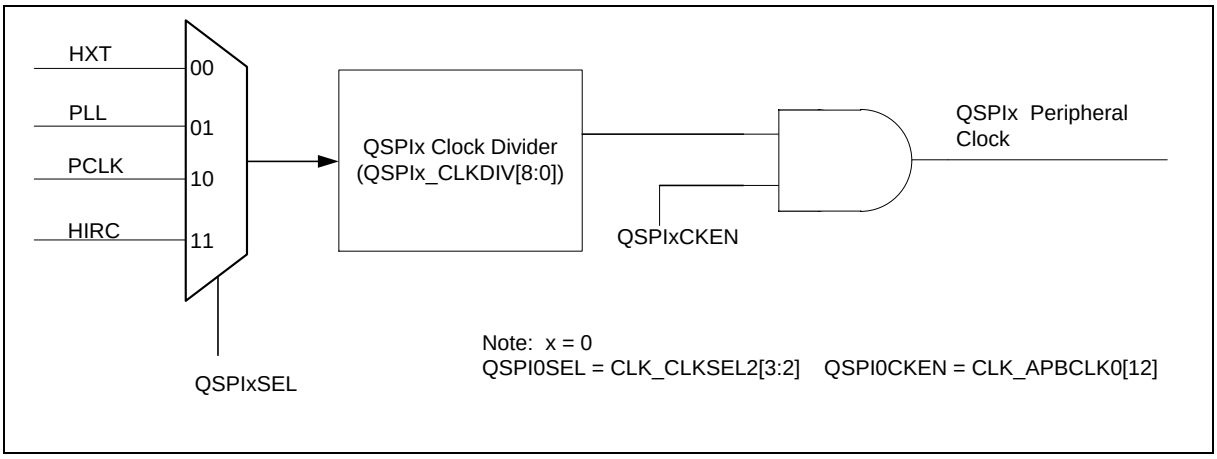


Figure 6.15-2 QSPI Peripheral Clock

In Master mode, the frequency of the QSPI bus clock is equal to the peripheral clock rate. In general, the QSPI bus clock is denoted as QSPI clock. In Slave mode, the QSPI bus clock is provided by a master device. The frequency of QSPI peripheral clock cannot be faster than the system clock rate regardless of Master or Slave mode.

Master/Slave mode

The QSPI controllers can be set as Master or Slave mode by setting the SLAVE (QSPIx_CTL[18]) to communicate with the off-chip SPI slave or master device. The HALFDPX (QSPIx_CTL[14]) can be used to select the full-duplex or half-duplex in QSPI transmission. The application block diagrams in Master and Slave mode are shown below.

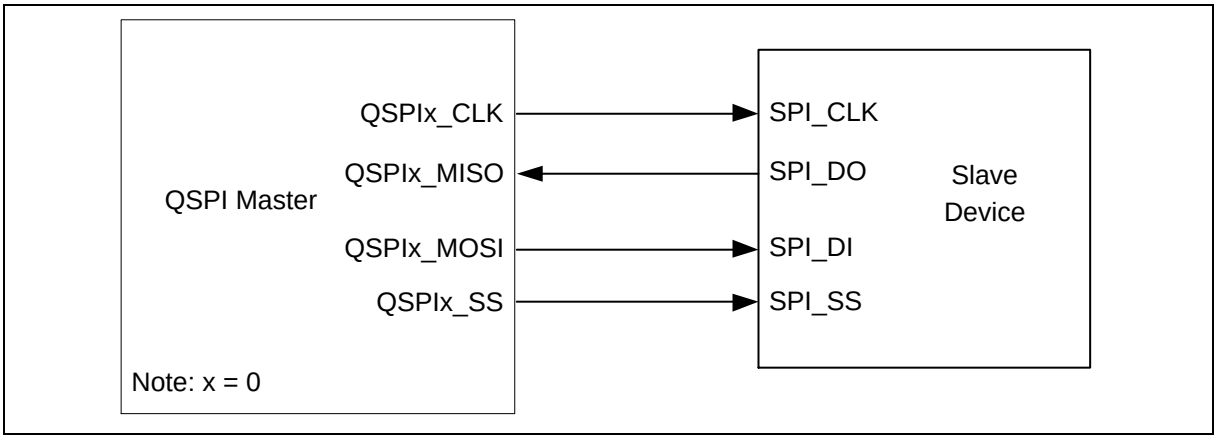


Figure 6.15-3 QSPI Full-Duplex Master Mode Application Block Diagram

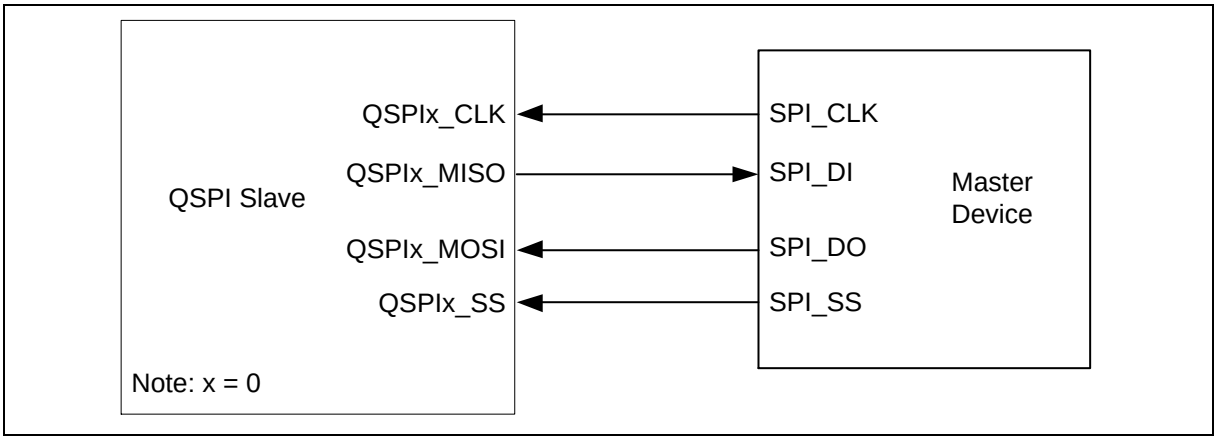


Figure 6.15-4 QSPI Full-Duplex Slave Mode Application Block Diagram

Slave Selection

In Master mode, the QSPI controller can drive off-chip slave device through the slave select output pin QSPIx_SS. In Slave mode, the off-chip master device drives the slave selection signal from the QSPIx_SS input port to this QSPI controller. The duration between the slave select active edge and the first QSPI clock input shall over 3 QSPI 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 (QSPIx_SSCTL[2]). The selection of slave select conditions depends on what type of device is connected. In Slave mode, to recognize the inactive state of the slave selection signal, the inactive period of the slave selection signal must be larger than or equal to 3 peripheral clock cycles between two successive transactions.

Timing Condition

The CLKPOL (QSPIx_CTL[3]) defines the QSPI clock idle state. If CLKPOL = 1, the output QSPI clock is idle at high state; if CLKPOL = 0, it is idle at low state.

TXNEG (QSPIx_CTL[2]) defines the data transmitted out either on negative edge or on positive edge of QSPI clock. RXNEG (QSPIx_CTL[1]) defines the data received either on negative edge or on positive edge of QSPI 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 (QSPIx_CTL[12:8]) and can be configured up to 32-bit length in a transaction word for transmitting and receiving.

When QSPI controller finishes a transaction, i.e. receives or transmits a specific count of bits defined in DWIDTH (QSPIx_CTL[12:8]), the unit transfer interrupt flag UNITIF (QSPIx_STATUS[1]) will be set to 1.

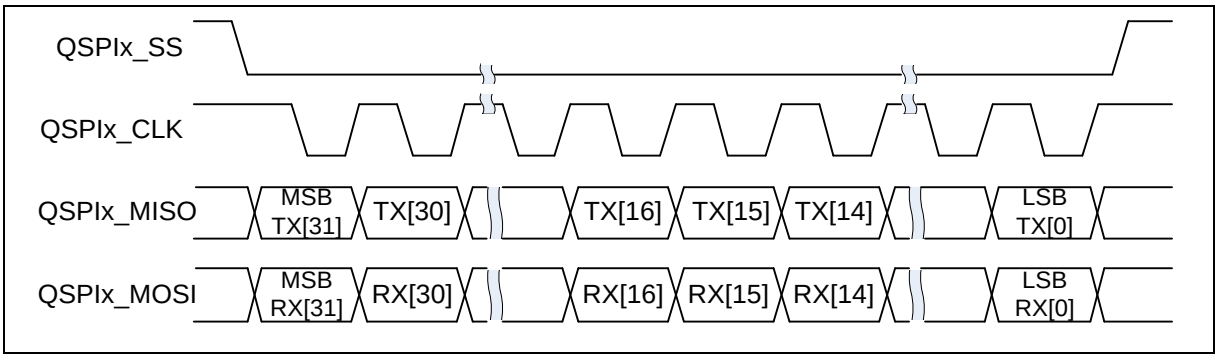


Figure 6.15-532-bit in One Transaction

LSB/MSB First

LSB (QSPIx_CTL[13]) defines the bit transfer sequence in a transaction. If the LSB (QSPIx_CTL[13]) is set to 1, the transfer sequence is LSB first. The bit 0 will be transferred firstly. If the LSB (QSPIx_CTL[13]) is cleared to 0, the transfer sequence is MSB first.

Suspend Interval

SUSPITV (QSPIx_CTL[7:4]) provides a configurable suspend interval, 0.5 ~ 15.5 QSPI 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 QSPI clock cycles).

6.15.5.2 Automatic Slave Selection

In Master mode, if AUTOSS (QSPIx_SSCTL[3]) is set, the slave selection signal will be generated automatically and output to the QSPIx_SS pin according to whether SS (QSPIx_SSCTL[0]) is enabled or not. The slave selection signal will be set to active state by the QSPI controller when the QSPI data transfer is started by writing to FIFO. It will be set to inactive state when QSPI bus is idle. If QSPI 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 (QSPIx_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 (QSPIx_SSCTL[2]).

The duration between the slave selection signal active edge and the first QSPI bus clock edge is 1 QSPI bus clock cycle and the duration between the last QSPI bus clock and the slave selection signal inactive edge is 1.5 QSPI bus clock cycle.

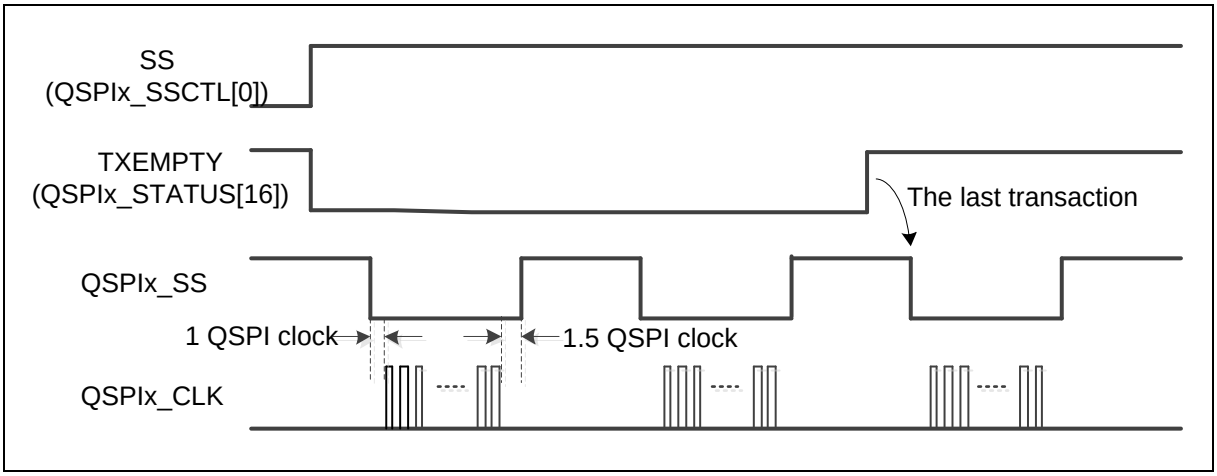


Figure 6.15-6 Automatic Slave Selection (SSACTPOL = 0, SUSPITV > 0x2)

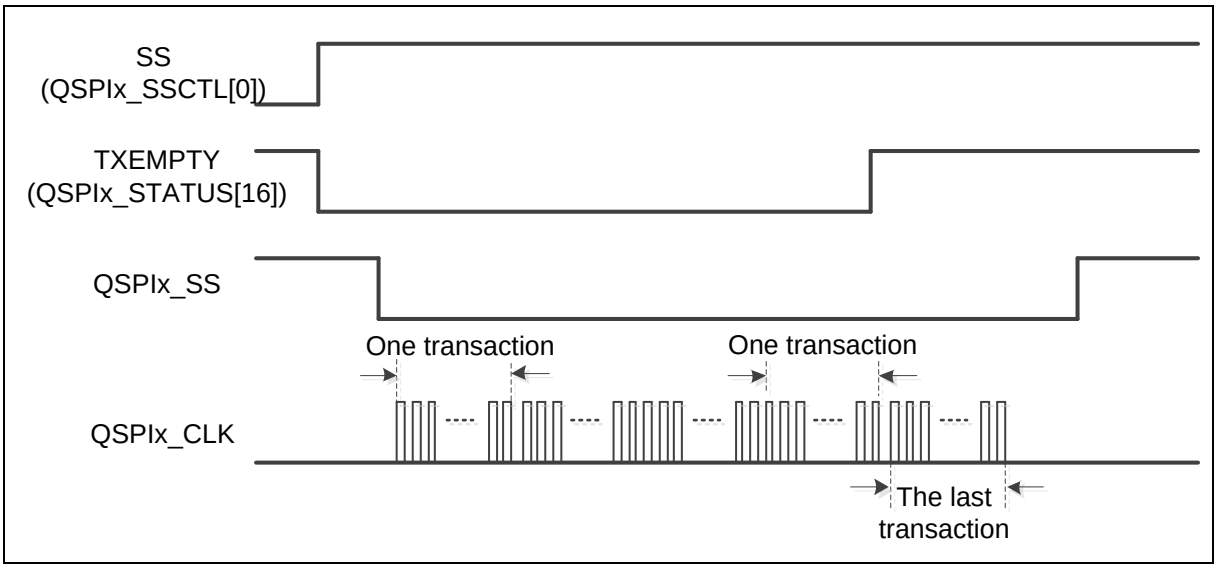


Figure 6.15-7 Automatic Slave Selection (SSACTPOL = 0, SUSPITV < 0x3)

6.15.5.3 Byte Reorder and Suspend Function

When the transfer is set as MSB first (LSB = 0) and the REORDER (QSPIx_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 QSPI 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.

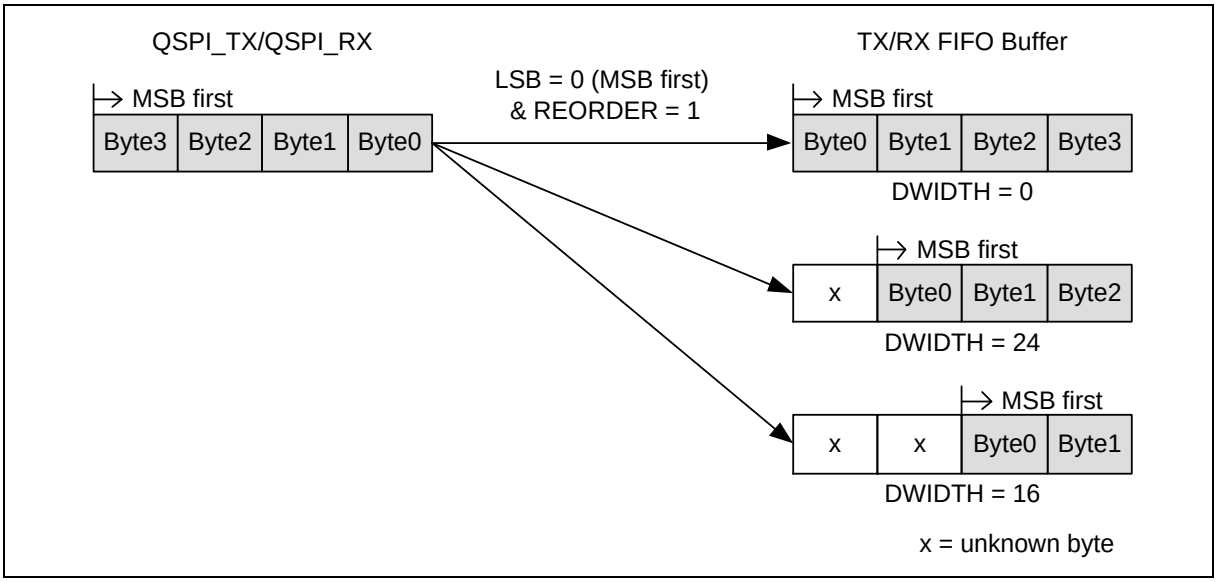


Figure 6.15-8 Byte Reorder Function

In Master mode, if REORDER (QSPIx_CTL[19]) is set to 1, a suspend interval of 0.5 ~ 15.5 QSPI clock periods will be inserted by hardware between two successive bytes in a transaction word. The suspend interval is configured in SUSPITV (QSPIx_CTL[7:4]).

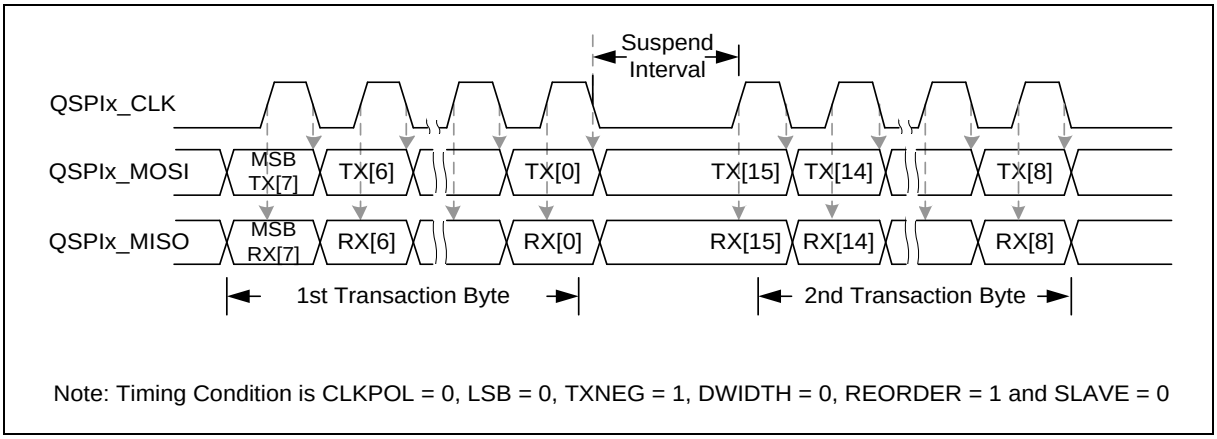


Figure 6.15-9 Timing Waveform for Byte Suspend

6.15.5.4 Half-Duplex Communication

The QSPI controller can communicate in half-duplex mode by setting HALFDPX (QSPIx_CTL[14]) bit. In half-duplex mode, there is only one data line for receiving or transmitting data direction which is defined by DATDIR (QSPIx_CTL[20]). In half-duplex configuration, the QSPIx_MISO pin is free for other applications and it can be configured as GPIO. Enabling or disabling the control bit HALFDPX (QSPIx_CTL[14]) will produce TXFBCLR (QSPIx_FIFOCTL[9]) and RXFBCLR (QSPIx_FIFOCTL[8]) at the same time automatically.

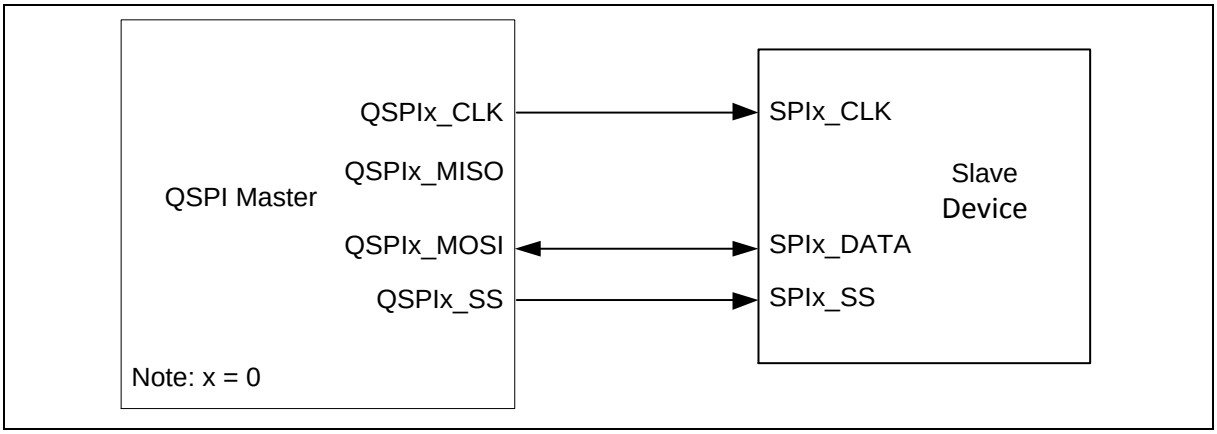


Figure 6.15-10 QSPI Half-Duplex Master Mode Application Block Diagram

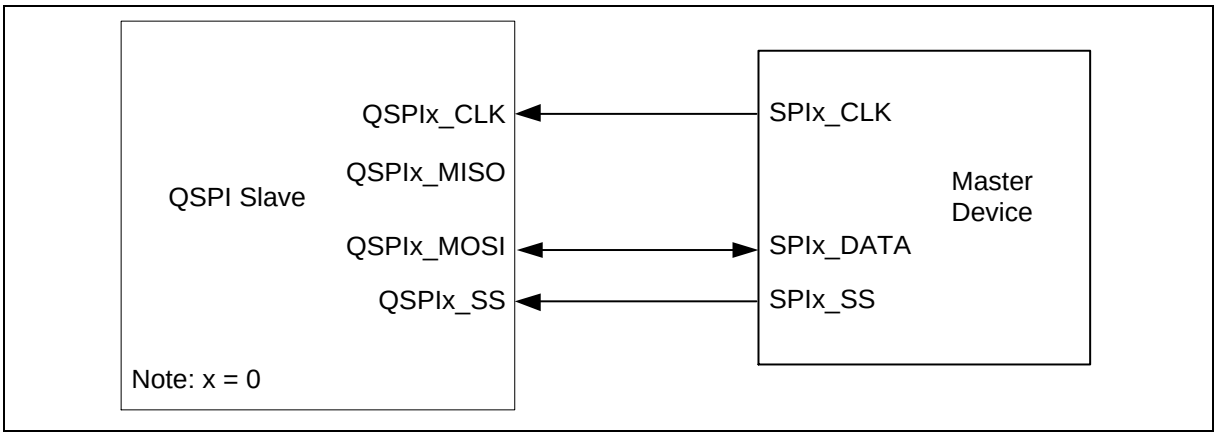


Figure 6.15-11 QSPI Half-Duplex Slave Mode Application Block Diagram

6.15.5.5 Receive-Only Mode

In QSPI Master device, it can communicate in receive-only mode by setting RXONLY (QSPIx_CTL[15]). In this configuration, the QSPI Master device will generate QSPI bus clock continuously as long as the receive-only mode is enabled for receiving data bit from SPI slave device. If AUTOSS (QSPIx_SSCTL[3]) is enabled in receive-only mode, QSPI Master will keep activating the slave select signal.

The remaining QSPIx_MOSI pin of QSPI Master device is not used for communication and can be configured as GPIO. The status BUSY (QSPIx_STATUS[0]) will be asserted in receive-only mode due to the generation of QSPI bus clock. Entering this mode will produce the TXFBCLR (QSPIx_FIFOCTL[9]) and RXFBCLR (QSPIx_FIFOCTL[8]) at the same time automatically. When user enables this mode, the output QSPI 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 (QSPIx_CTL[15]) enable, QSPI RX data with data bit width of DWIDTH (QSPIx_CTL[12:8]) will be received into RX FIFO and QSPI clock will be sent to SPI slave device until RX FIFO is full.

For two-bit transfer mode TWOBIT(QSPIx_CTL[16]) enable, the QSPI master will send QSPI output clock to SPI slave and receive RX data until RX FIFO is full. After user reads RX data from RX FIFO, QSPI master will send QSPI output clock to SPI slave and receive RX data again when RX FIFO counter RXCNT is less than or equal to 4.

6.15.5.6 Slave 3-Wire Mode

When SLV3WIRE (QSPIx_SSCTL[4]) is set by software to enable the Slave 3-Wire mode, the QSPI controller can work with no slave selection signal in Slave mode. The SLV3WIRE (QSPIx_SSCTL[4]) only takes effect in Slave mode. Only three pins, QSPIx_CLK, QSPIx_MISO, and QSPIx_MOSI, are required to communicate with a SPI master. The QSPIx_SS pin can be configured as a GPIO. When the SLV3WIRE (QSPIx_SSCTL[4]) is set to 1, the QSPI slave will be ready to transmit/receive data after the SPIEN (QSPIx_CTL[0]) is set to 1.

6.15.5.7 PDMA Transfer Function

QSPI controller supports PDMA transfer function.

When TXPDMAEN (QSPIx_PDMACTL[0]) is set to 1, the controller will issue request to PDMA controller to start the PDMA transmission process automatically.

When RXPDMAEN (QSPIx_PDMACTL[1]) is set to 1, the controller will start the PDMA reception process. QSPI controller will issue request to PDMA controller automatically when there is data in the RX FIFO buffer.

When PDMA transfer is done, user needs to disable TXPDMAEN/RXPDMAEN. After re-setting control registers of PDMA, user enables TXPDMAEN/RXPDMAEN again.

Note: QSPI supports single request PDMA (Read/Write) only, burst request PDMA is not supported.

6.15.5.8 Two-bit Transfer Mode

The QSPI controller also supports 2-bit Transfer mode when setting TWOBIT (QSPIx_CTL[16]) to 1. In 2-bit Transfer mode, the QSPI controller performs full duplex data transfer. In other words, the two serial data bits can be transmitted and received simultaneously.

For example, in Master mode, the even data (TX Data (n)) stored in the TX FIFO of QSPIx will be transmitted through the QSPIx_MOSI0 pin and the odd data (TX Data (n+1)) stored in the TX FIFO of QSPIx will be transmitted through the QSPIx_MOSI1 pin respectively. In the meanwhile, the even data received from QSPIx_MISO0 pin will be written to RX FIFO prior to the odd data received from QSPIx_MISO1 pin.

In Slave mode, the even and odd data stored in the TX FIFO of QSPIx will be transmitted through the QSPIx_MISO0 pin and QSPIx_MISO1 pin respectively. In the meanwhile, the RX FIFO of QSPIx will store the even data received from the QSPIx_MOSI0 pin and the odd data from QSPIx_MOSI1 pin respectively. The data sequence of FIFO buffers is the same as the Master mode.

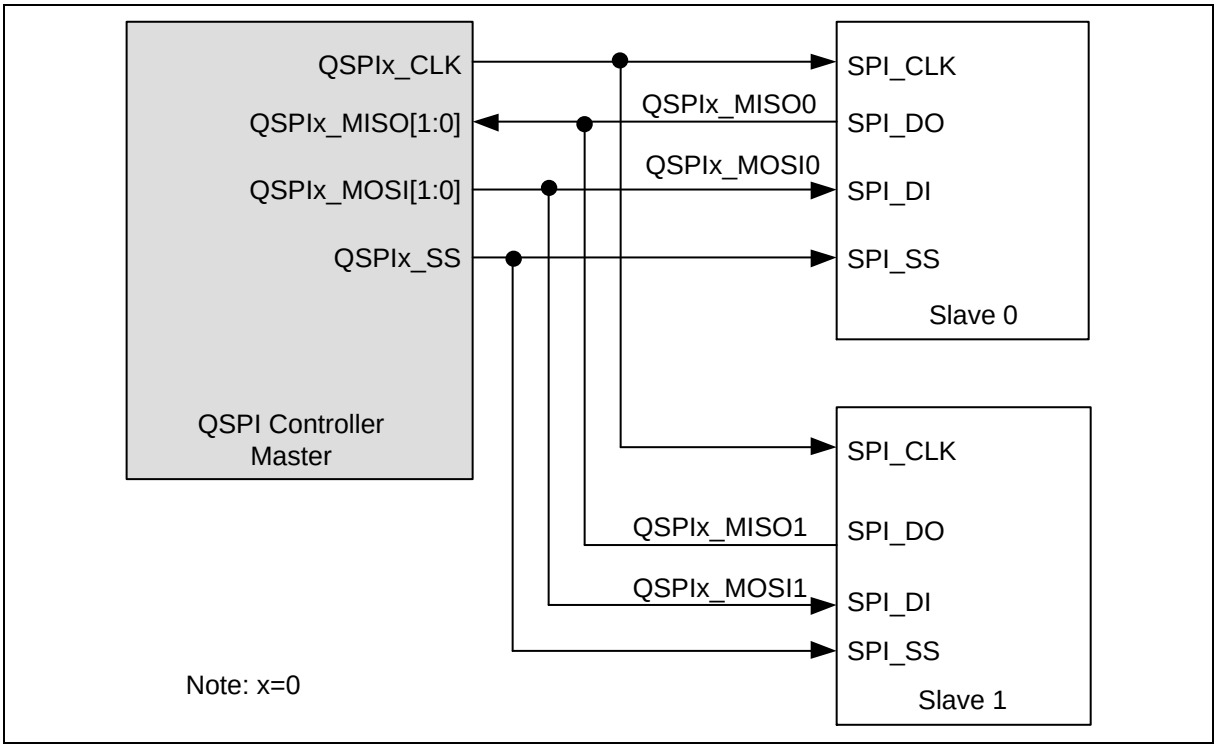


Figure 6.15-12 Two-bit Transfer Mode System Architecture

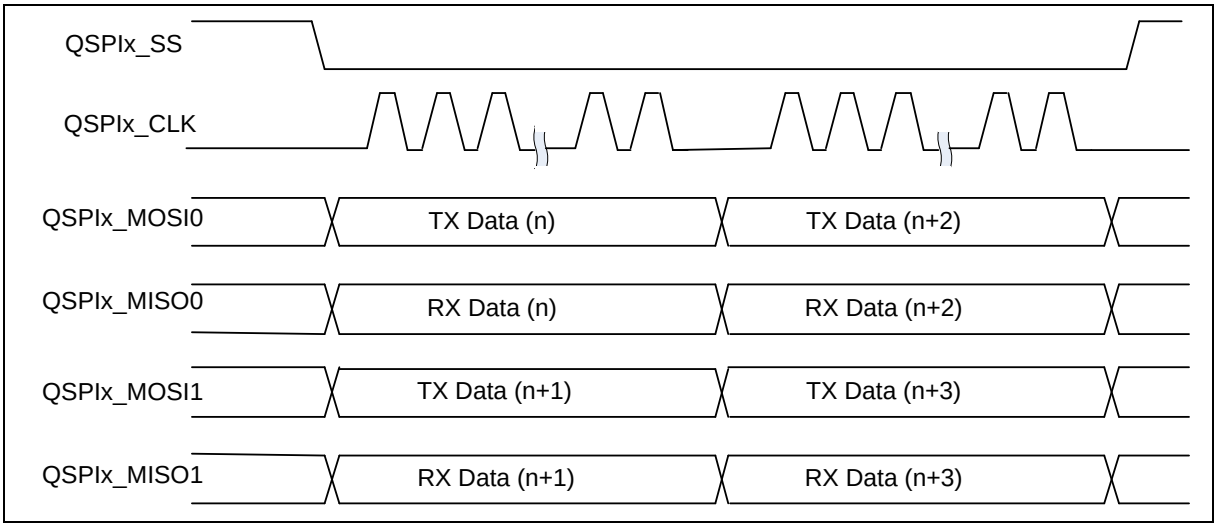


Figure 6.15-13 Two-bit Transfer Mode Timing (Master Mode)

6.15.5.9 Dual I/O Mode

The QSPI controller also supports Dual I/O transfer when setting the DUALIOEN ((QSPIx_CTL[21]) to 1. Many general SPI Flashes support Dual I/O transfer. The DATDIR (QSPIx_CTL[20]) is used to define the direction of the transfer data. When the DATDIR bit is set to 1, the controller will send the data to external device. When the DATDIR bit is set to 0, the controller will read the data from the external device. This function supports 8, 16, 24, and 32 bits of length.

The Dual I/O mode is not supported when the Slave 3-Wire mode or the Byte Reorder function is enabled.

For Dual I/O mode, if both the DUALIOEN (QSPIx_CTL[21]) and DATDIR (QSPIx_CTL[20]) are set as

1, the QSPIx_MOSI0 is the even bit data output and the QSPIx_MISO0 will be set as the odd bit data output. If the DUALIOEN (QSPIx_CTL[21]) is set as 1 and DATDIR (QSPIx_CTL[20]) is set as 0, both the QSPIx_MISO0 and QSPIx_MOSI0 will be set as data input ports.

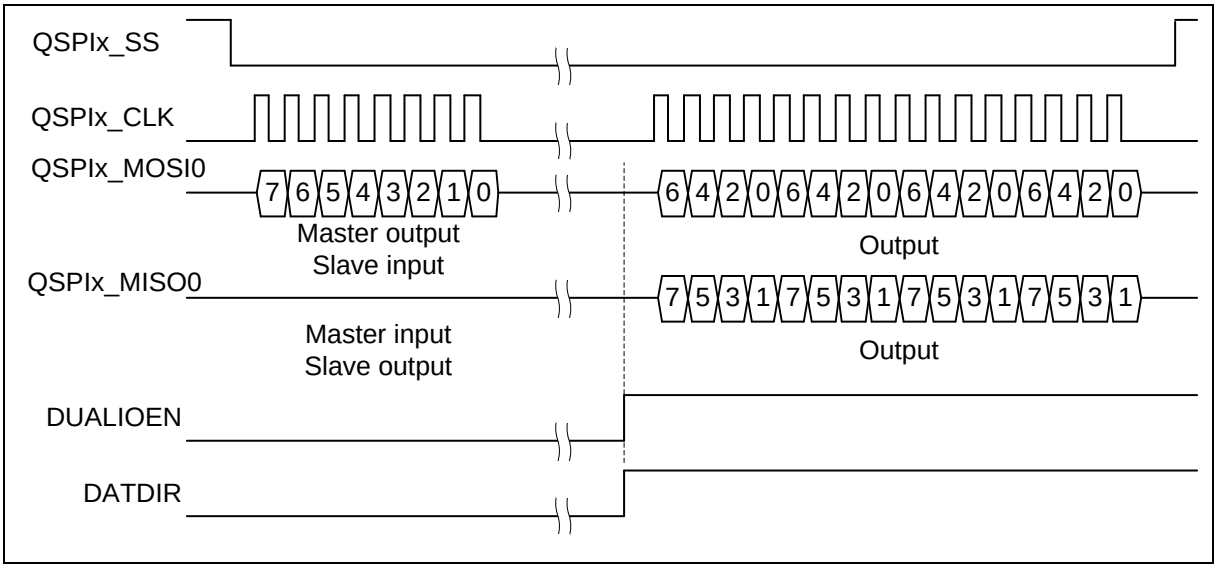


Figure 6.15-14 Bit Sequence of Dual Output Mode

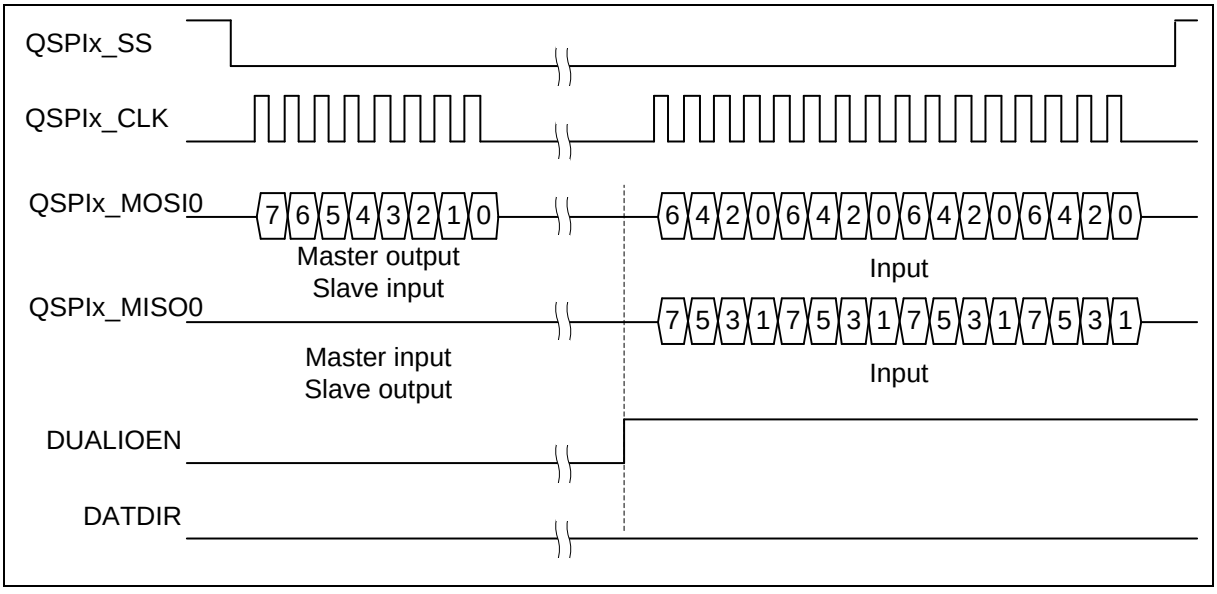


Figure 6.15-15 Bit Sequence of Dual Input Mode

6.15.5.10 Quad I/O Mode

The QSPI controller also supports Quad I/O transfer when setting the QUADIOEN (QSPIx_CTL[22]) to 1. Many general SPI Flashes support Quad I/O transfer. The DATDIR bit (QSPIx_CTL[20]) is used to define the direction of the transfer data. When the DATDIR (QSPIx_CTL[20]) is set to 1, the controller will send the data to external device. When the DATDIR (QSPIx_CTL[20]) is set to 0, the controller will read the data from the external device. This function supports 8, 16, 24, and 32 bits of length.

The Quad I/O mode is not supported when the Slave 3-Wire mode or the Byte Reorder function is enabled. The DUALIOEN (QSPIx_CTL[21]) and QUADIOEN (QSPIx_CTL[22]) shall not be set to 1 simultaneously.

For Quad I/O mode, if both the QUADIOEN (QSPIx_CTL[22]) and DATDIR (QSPIx_CTL[20]) are set as 1, the QSPIx_MOSI0 and QSPIx_MOSI1 are the even bit data output and the QSPIx_MISO0 and QSPIx_MISO1 will be set as the odd bit data output. If the QUADIOEN (QSPIx_CTL[22]) is set as 1 and DATDIR (QSPIx_CTL[20]) is set as 0, all the QSPIx_MISO0, QSPIx_MISO1, QSPIx_MOSI0 and QSPIx_MOSI1 pins will be set as data input ports.

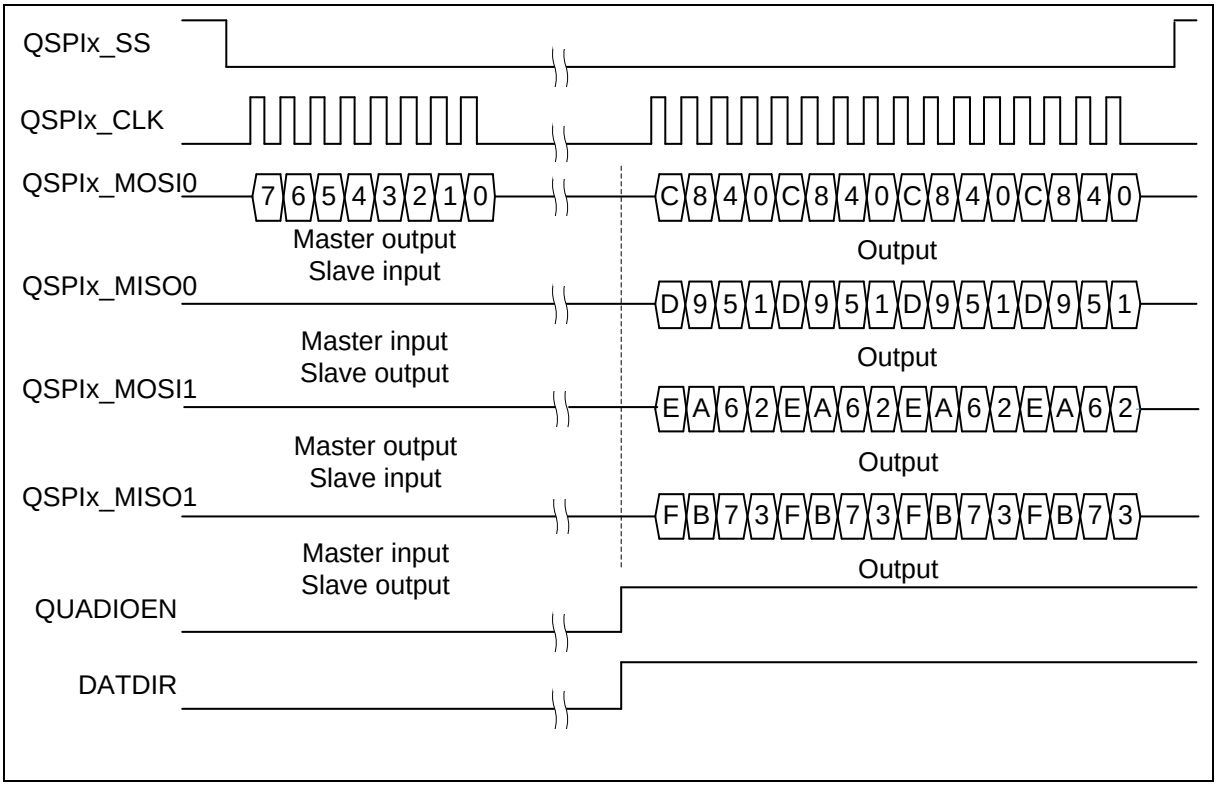


Figure 6.15-16 Bit Sequence of Quad Output Mode

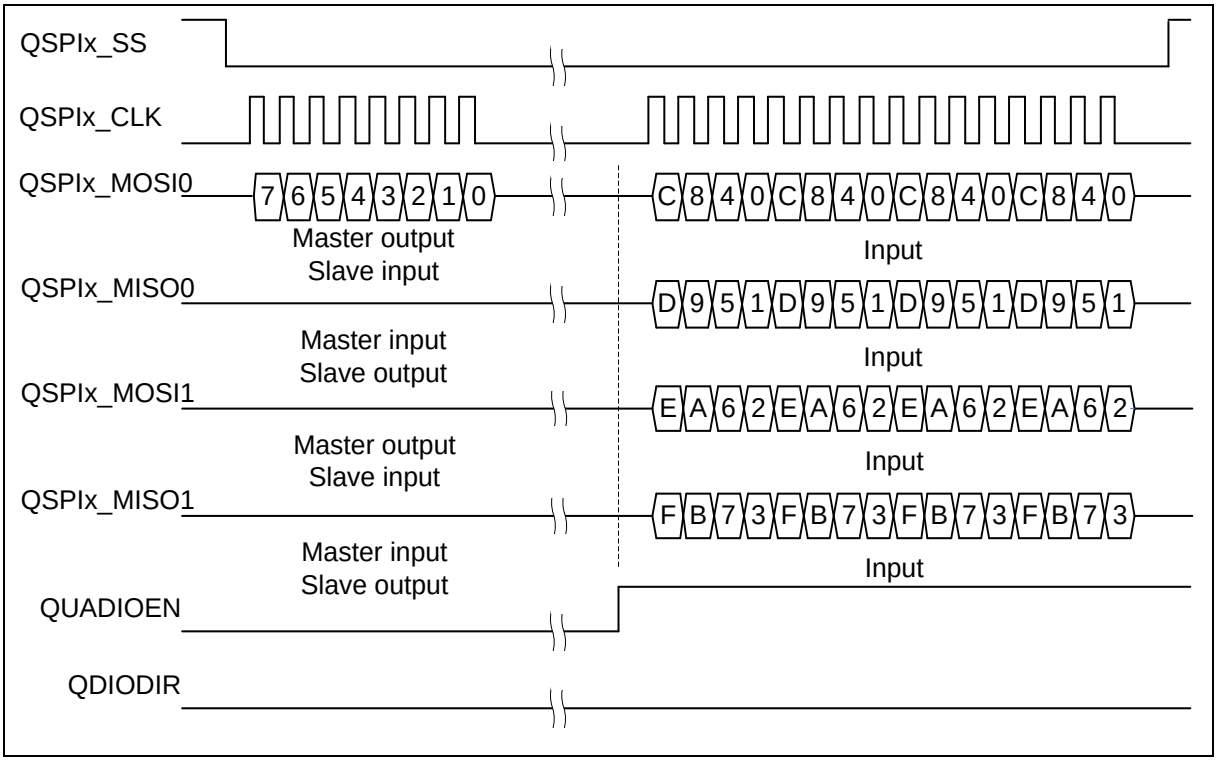


Figure 6.15-17 Bit Sequence of Quad Input Mode

6.15.5.11 FIFO Buffer Operation

The QSPI controller is equipped with eight 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 (QSPIx_STATUS[17]) will be set to 1. When the QSPI transmission logic unit draws out the last datum of the transmit FIFO buffer, so that the transmit FIFO buffer is empty, the TXEMPTY (QSPIx_STATUS[16]) will be set to 1. Note that the TXEMPTY (QSPIx_STATUS[16]) flag is set to 1 while the last transaction is still in progress. In Master mode, the BUSY (QSPIx_STATUS[0]) is set to 1 when the FIFO buffer is written any data or there is any transaction on the QSPI bus. (e.g. the slave selection signal is active and the QSPI 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 (QSPIx_STATUS[0]) should be checked by software to make sure whether the QSPI is in idle or not.

The receive control logic will store the QSPI input data into the receive FIFO buffer. There are FIFO related status bits, like RXEMPTY (QSPIx_STATUS[8]) and RXFULL (QSPIx_STATUS[9]), to indicate the current status of RX FIFO buffer.

The transmitting and receiving threshold can be configured by setting TXTH (QSPIx_FIFOCTL[30:28]) and RXTH (QSPIx_FIFOCTL[26:24]). When the count of valid data stored in transmit FIFO buffer is less than or equal to TXTH (QSPIx_FIFOCTL[30:28]) setting, TXTHIF (QSPIx_STATUS[18]) will be set to 1. When the count of valid data stored in receive FIFO buffer is larger than RXTH (QSPIx_FIFOCTL[26:24]) setting, RXTHIF (QSPIx_STATUS[10]) will be set to 1.

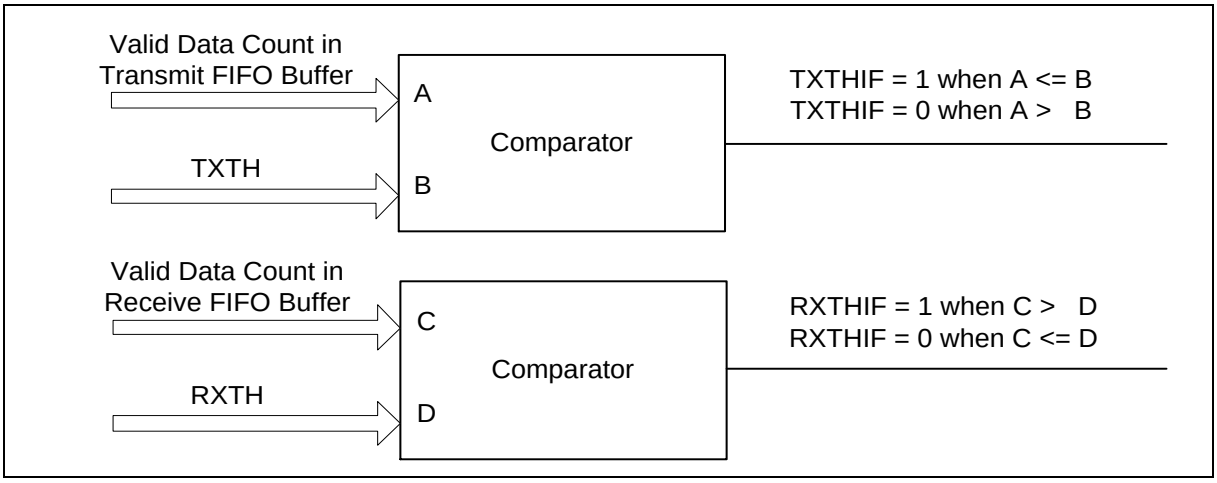


Figure 6.15-18 FIFO Threshold Comparator

In Master mode, when the first datum is written to the QSPIx_TX register, the TXEMPTY flag (QSPIx_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 QSPIx_TX register immediately. The QSPI controller will insert a suspend interval between two successive transactions. The period of suspend interval is decided by the setting of SUSPITV (QSPIx_CTL[7:4]). If the SUSPITV (QSPIx_CTL[7:4]) equals 0, QSPI controller can perform continuous transfer. User can write data into QSPIx_TX register as long as the TXFULL (QSPIx_STATUS[17]) is 0.

In the example 1 of Figure 6.14-13, it indicates the updated condition of TXEMPTY (QSPIx_STATUS[16]) and the relationship among the FIFO buffer, shift register and the skew buffer. The TXEMPTY (QSPIx_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 (QSPIx_STATUS[16]) will be to 1. The Data0 in shift register will be shift into skew buffer by bit for transmission until the transfer is done.

In the Example 2, it indicates the updated condition of TXFULL (QSPIx_STATUS[17]) when there are 8 data in the FIFO buffer and the next data of Data9 does not be written into the FIFO buffer when the TXFULL = 1.

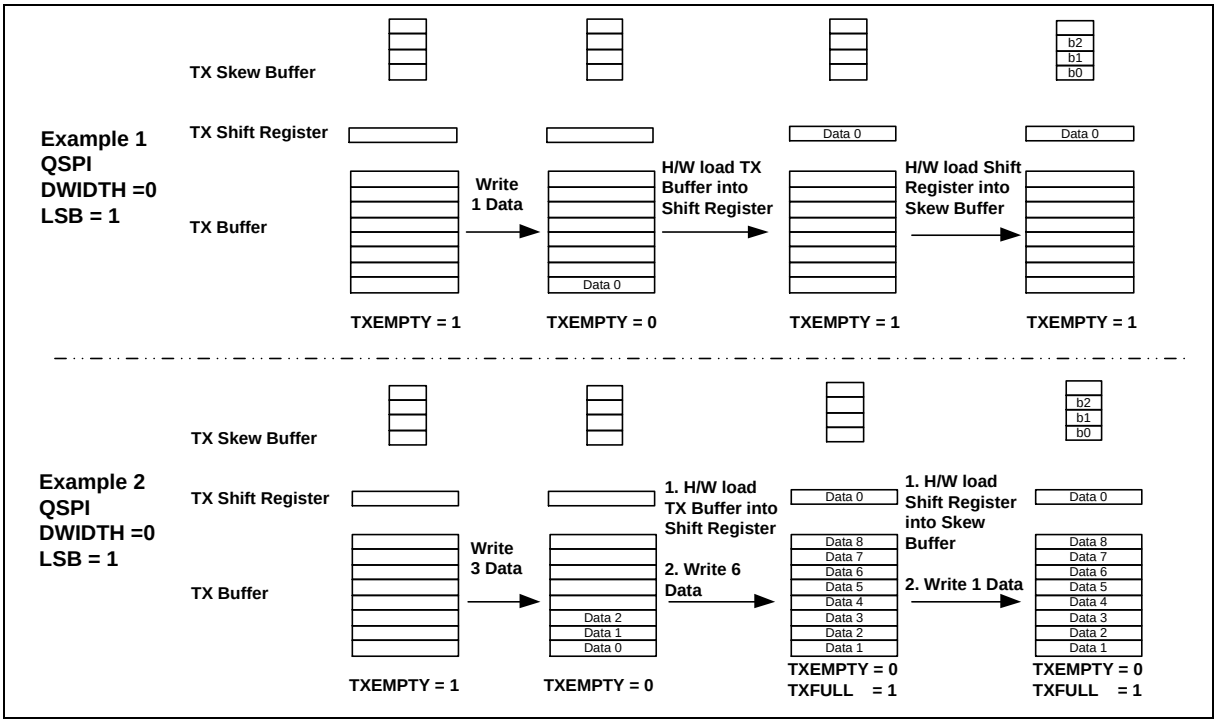


Figure 6.15-19 Transmit FIFO Buffer Example

The subsequent transactions will be triggered automatically if the transmitted data are updated in time. If the QSPIx_TX register does not be updated after all data transfer are done, the transfer will stop.

In Master mode, during receiving operation, the serial data are received from QSPIx_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 (QSPIx_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 (QSPIx_CTL[12:8]). The RXEMPTY (QSPIx_STATUS[8]) will be cleared to 0 while the receive FIFO buffer contains unread data (see the Example 1 of Receive FIFO Buffer Example). The received data can be read by software from QSPIx_RX register as long as the RXEMPTY (QSPIx_STATUS[8]) is 0. If the receive FIFO buffer contains 8 unread data, the RXFULL (QSPIx_STATUS[9]) will be set to 1 (see the Example 2 of Receive FIFO Buffer Example).

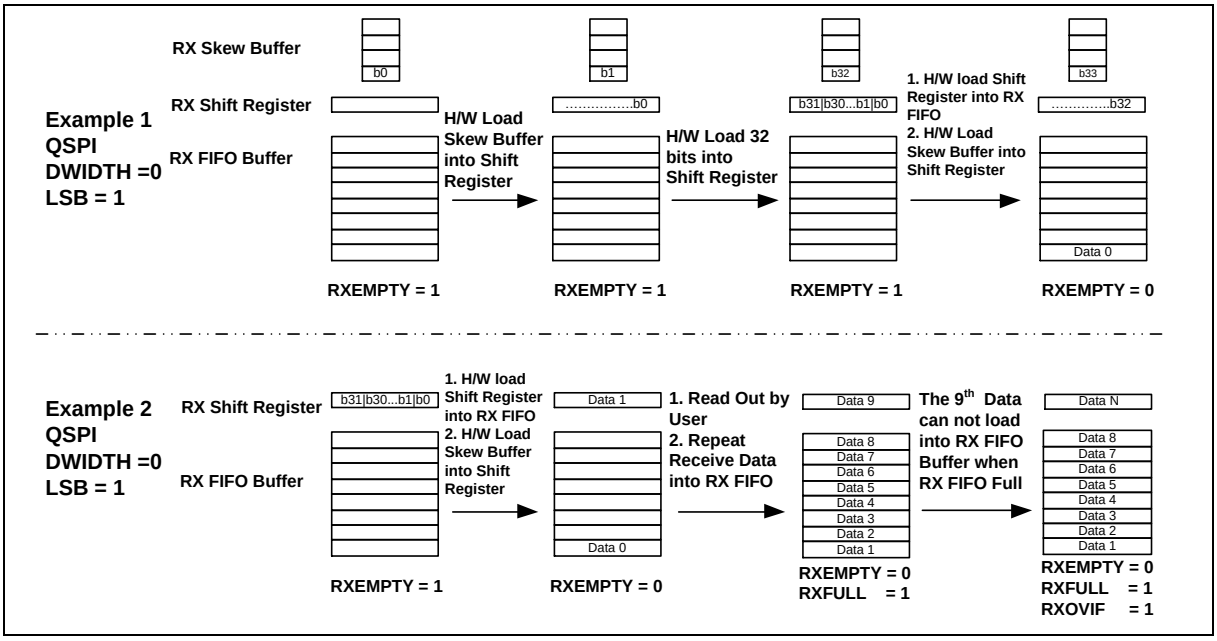


Figure 6.15-20 Receive FIFO Buffer Example

In Slave mode, during transmission operation, when data is written to the QSPIx_TX register by software, the data will be loaded into transmit FIFO buffer and the TXEMPTY (QSPIx_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 QSPIx_TX register as long as the TXFULL (QSPIx_STATUS[17]) is 0. After all data have been drawn out by the QSPI transmission logic unit and the QSPIx_TX register is not updated by software, the TXEMPTY (QSPIx_STATUS[16]) will be set to 1.

If there is no any data written to the QSPIx_TX register, the transmit underflow interrupt flag, TXUFIF (QSPIx_STATUS[19]) will be set to 1 when the slave selection signal is active. The output data will be held by TXUFPOL (QSPIx_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 (QSPIx_STATUS[7]), will be set to 1 as QSPIx_SS goes to inactive state.

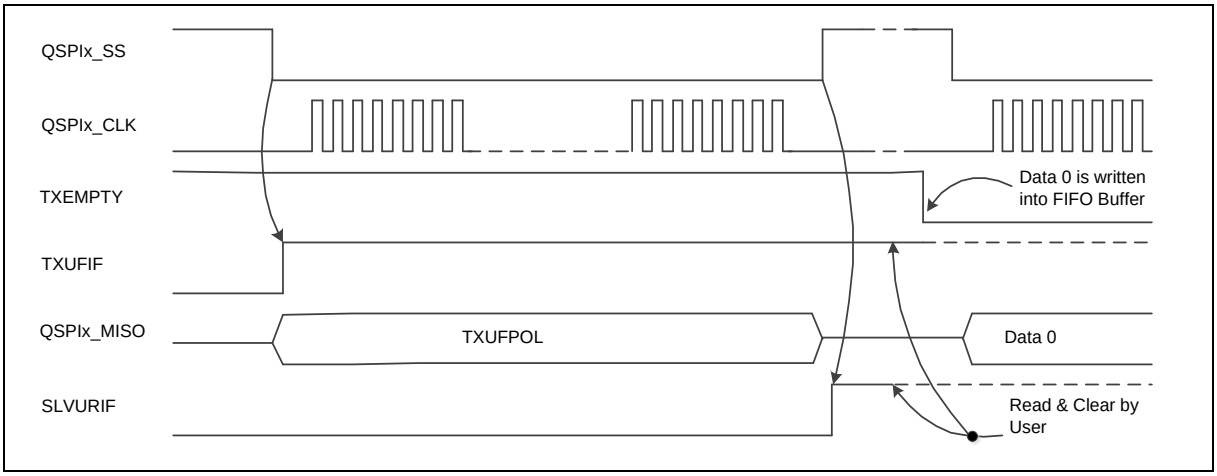


Figure 6.15-21 TX Underflow Event and Slave Under Run Event

In 2-bit Transfer mode, the transmit data is loaded into shift register after 2 datum have been written into the TX FIFO buffer. It uses two shift registers and two 4-level skew buffers concurrently. The detail timing of 2-bit Transfer mode, please refer to the section of Two-bit Transfer mode.

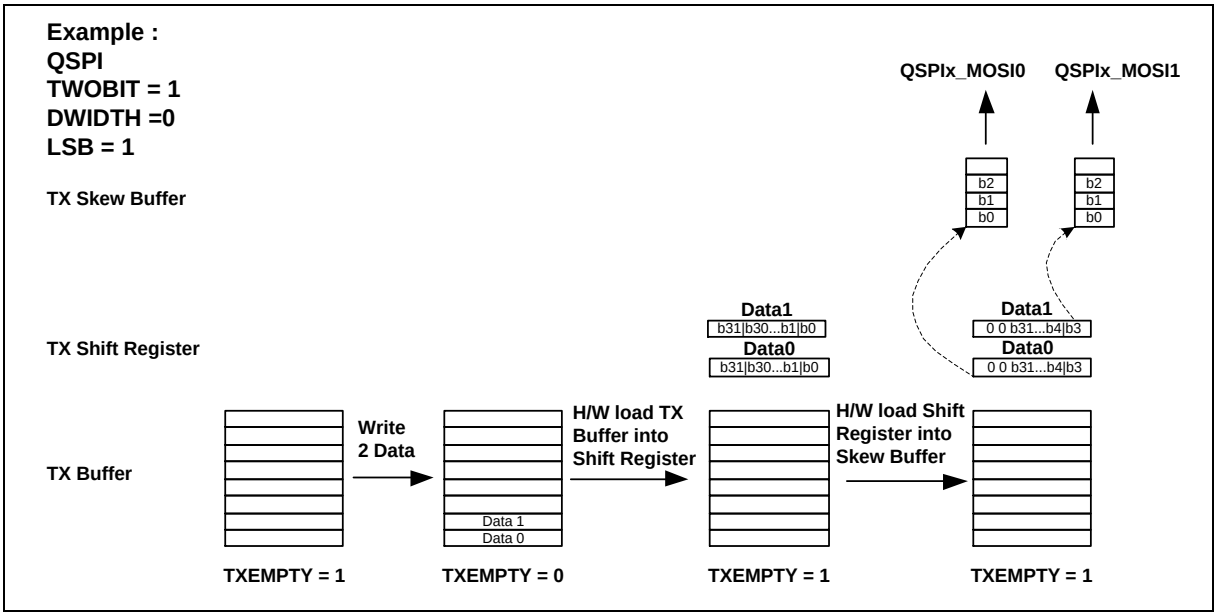


Figure 6.15-22 Two-bit Transfer Mode FIFO Buffer Example

In QSPI 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 QSPI bus clock is presented. The other bits are held by TXUFPOL (QSPIx_FIFOCTL[6]) because there is TX underflow event. The written data will be transmitted in the next transfer.

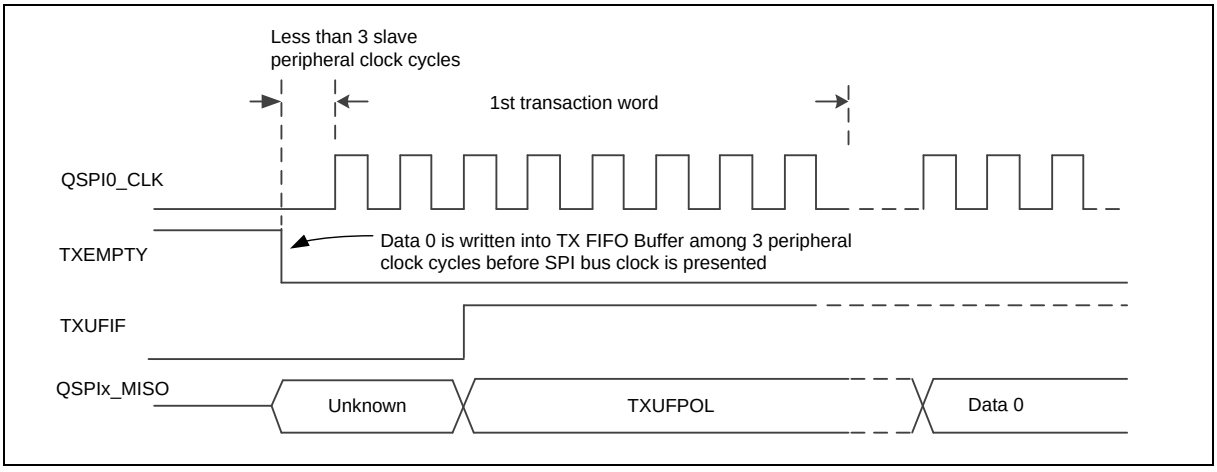


Figure 6.15-23 TX Underflow Event (QSPI0 Slave 3-Wire Mode Enabled)

In Slave mode, during receiving operation, the serial data is received from QSPIx_MOSI pin and stored to QSPIx_RX register. The reception mechanism is similar to Master mode reception operation. If the receive FIFO buffer contains 8 unread data, the RXFULL (QSPIx_STATUS[9]) will be set to 1 and the RXOVIF (QSPIx_STATUS[11]) will be set to 1 if there is more serial data received from QSPIx_MOSI and follow-up data will be dropped (refer to the Receive FIFO Buffer Example figure). If the receive bit count mismatch with the DWIDTH (QSPIx_CTL[12:8]) when the slave selection line goes to inactive state, the SLVBEIF (QSPIx_STATUS[6]) will be set to 1.

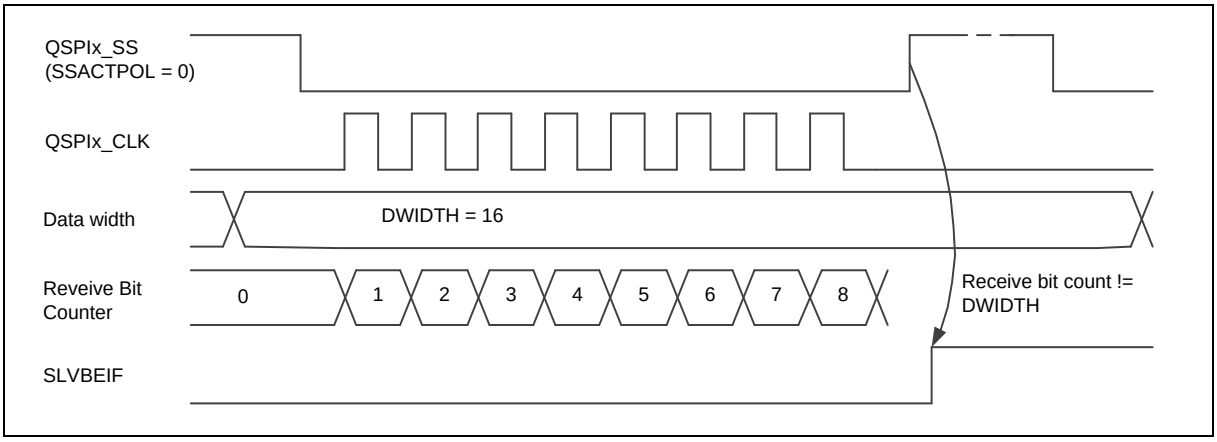


Figure 6.15-24 Slave Mode Bit Count Error

When the Slave select signal is active and the value of SLVTOCNT (QSPIx_SSCTL[31:16]) is not 0, the Slave time-out counter in the QSPI controller logic will start after the serial clock input. This counter will be cleared after one transaction done or the SLVTOCNT is set to 0. If the value of the time-out counter is equal to the value of SLVTOCNT before one transaction done, the slave time-out event occurs and the SLVTOIF (QSPIx_STATUS[5]) will be set to 1.

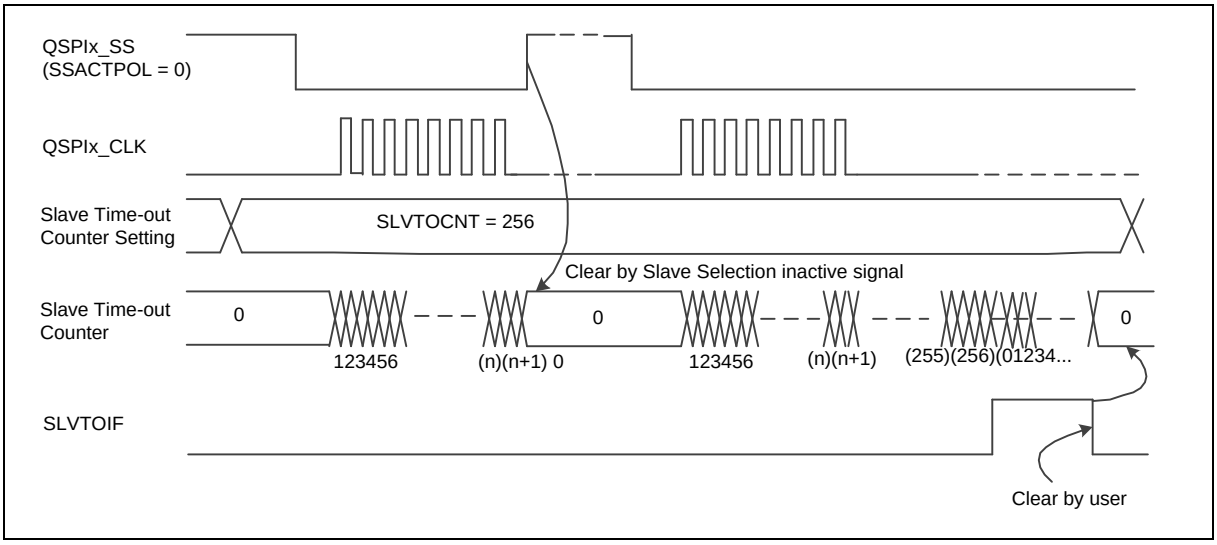


Figure 6.15-25 Slave Time-out Event

The QSPI controller has the receive time-out function. When the receive FIFO is not empty and no read operation in receive FIFO over 64 QSPI peripheral clock periods in Master mode or over 576 QSPI peripheral clock periods in Slave mode, the receive time-out occurs and the RXTOIF (QSPIx_STATUS[12]) will be set to 1. When the receive FIFO is read by user, the time-out status will be cleared automatically.

6.15.5.12 Interrupt

- QSPI unit transfer interrupt

  As the QSPI controller finishes a unit transfer, the unit transfer interrupt flag UNITIF (QSPIx_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 (QSPIx_CTL[17]) is set. The unit transfer interrupt flag can be cleared only by writing 1 to it.

- QSPI slave selection active/inactive interrupt

In Slave mode, the slave selection active/inactive interrupt flag, SSACTIF (QSPIx_STATUS[2]) and SSINAIF (QSPIx_STATUS[3]), will be set to 1 when the SPIEN (QSPIx_CTL[0]) and SLAVE (QSPIx_CTL[18]) are set to 1 and the slave selection signal goes to active/inactive state. The QSPI controller will issue an interrupt if the SSINAIEN (QSPIx_SSCTL[13]) or SSACTIEN (QSPIx_SSCTL[12]), are set to 1.

- Slave time-out interrupt

  In Slave mode, there is slave time-out function for user to know that there is serial clock input but one transaction is not finished over the period of SLVTOCNT (QSPIx_SSCTL[31:16]) basing on Slave peripheral clock.

  When the slave selection signal is active and the value of SLVTOCNT (QSPIx_SSCTL[31:16]) is not 0, the slave time-out counter in the QSPI controller logic will start after the serial clock input. This counter will be cleared after one transaction done or the SLVTOCNT (QSPIx_SSCTL[31:16]) is set to 0. If the value of the time-out counter is greater than or equal to the value of SLVTOCNT (QSPIx_SSCTL[31:16]) before one transaction done, the slave time-out event occurs and the SLVTOIF (QSPIx_STATUS[5]) will be set to 1. The QSPI controller will issue an interrupt if the SLVTOIEN (QSPIx_SSCTL[5]) is set to 1.

- Slave bit count error interrupt

  In Slave mode, if the transmit/receive bit count mismatch with the DWIDTH (QSPIx_CTL[12:8]) when the slave selection line goes to inactive state, the SLVBEIF (QSPIx_STATUS[6]) will be set to 1. The uncompleted transaction will be dropped from TX and RX shift registers. The QSPI controller will issue an interrupt if the SLVBEIEN (QSPIx_SSCTL[8]) is set to 1.

  Note: If the slave selection signal is active but there is no any serial clock input, the SLVBEIF (QSPIx_STATUS[6]) will be set to 1 when the slave selection signal goes to inactive state.

- TX underflow interrupt

  In QSPI Slave mode, if there is no any data is written to the QSPIx_TX register, the TXUFIF (QSPIx_STATUS[19]) will be set to 1 when the slave selection signal is active. The QSPI controller will issue a TX underflow interrupt if the TXUFIEN (QSPIx_FIFOCTL[7]) is set to 1.

  Note: If underflow event occurs in QSPI Slave mode, there are two conditions which make QSPI 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 while SLV3WIRE=0.

- Slave TX under run interrupt

  If the TX underflow event occurs, the SLVURIF (QSPIx_STATUS[7]) will be set to 1 when QSPIx_SS goes to inactive state. The QSPI controller will issue a TX under run interrupt if the SLVURIEN (QSPIx_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 (QSPIx_STATUS[19]) to know if there is TX underflow event or not.

- Receive Overrun interrupt

  In Slave mode, if the receive FIFO buffer contains 8 unread data, the RXFULL (QSPIx_STATUS[9]) will be set to 1 and the RXOVIF (QSPIx_STATUS[11]) will be set to 1 if there is more serial data received from QSPI bus and follow-up data will be dropped. The QSPI controller will issue an interrupt if the RXOVIEN (QSPIx_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 QSPI peripheral clock periods in Master mode or over 576 QSPI 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 (QSPIx_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 (QSPIx_FIFOCTL[30:28]), the transmit FIFO interrupt flag TXTHIF (QSPIx_STATUS[18]) will be set to 1. The QSPI controller will generate a transmit FIFO interrupt to the system if the transmit FIFO interrupt enable bit, TXTHIEN (QSPIx_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 (QSPIx_FIFOCTL[26:24]), the receive FIFO interrupt flag RXTHIF (QSPIx_STATUS[10]) will be set to 1. The QSPI controller will generate a receive FIFO interrupt to the system if the receive FIFO interrupt enable bit, RXTHIEN (QSPIx_FIFOCTL[2]), is set to 1.

6.15.6 Timing Diagram

The active state of slave selection signal can be defined by setting the SSACTPOL (QSPIx_SSCTL[2]). The QSPI clock which is in idle state can be configured as high or low state by setting the CLKPOL (QSPIx_CTL[3]). It also provides the bit length of a transaction word in DWIDTH (QSPIx_CTL[12:8]), and transmitting/receiving data from MSB or LSB first in LSB (QSPIx_CTL[13]). User can also select which edge of QSPI clock to transmit/receive data in TXNEG/RXNEG (QSPIx_CTL[2:1]). Four QSPI timing diagrams for master/slave operations and the related settings are shown below.

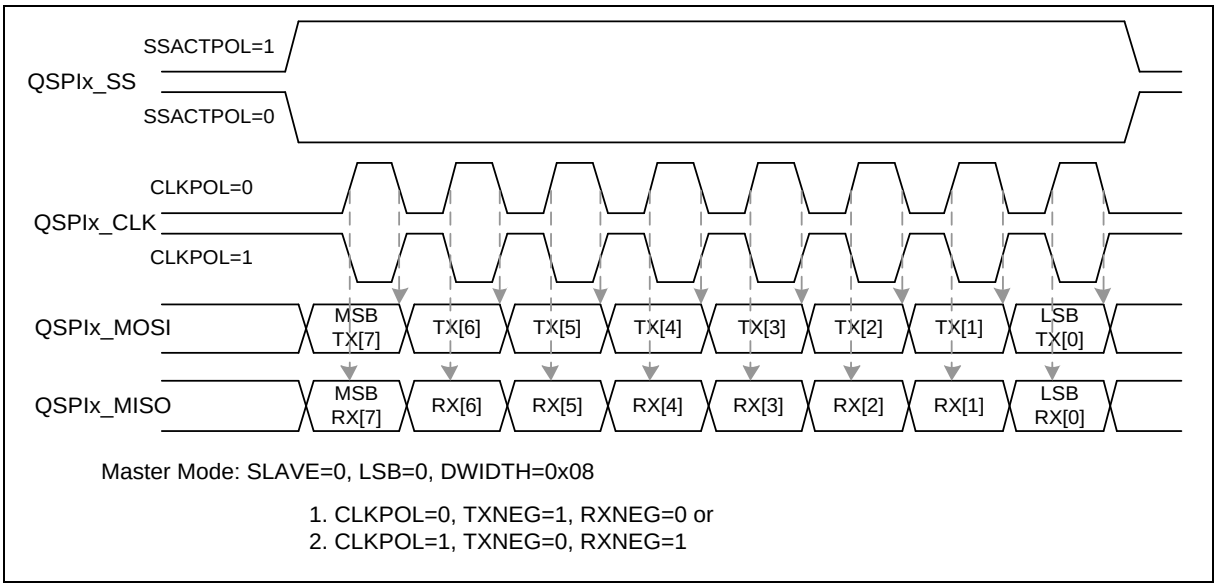


Figure 6.15-26 QSPI Timing in Master Mode

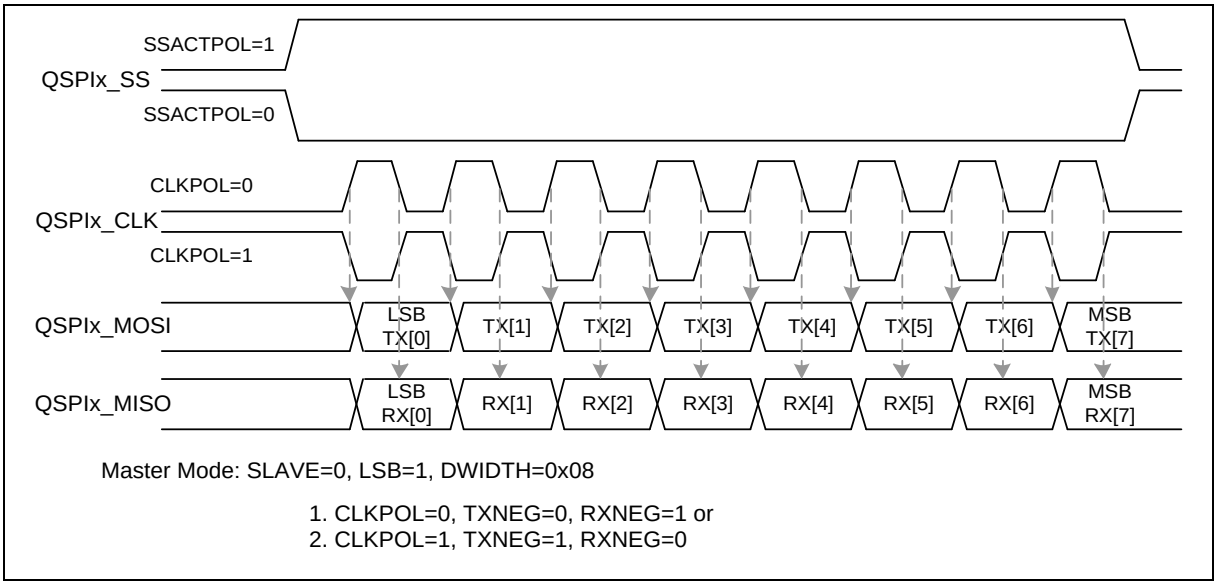


Figure 6.15-27 QSPI Timing in Master Mode (Alternate Phase of QSPIx_CLK)

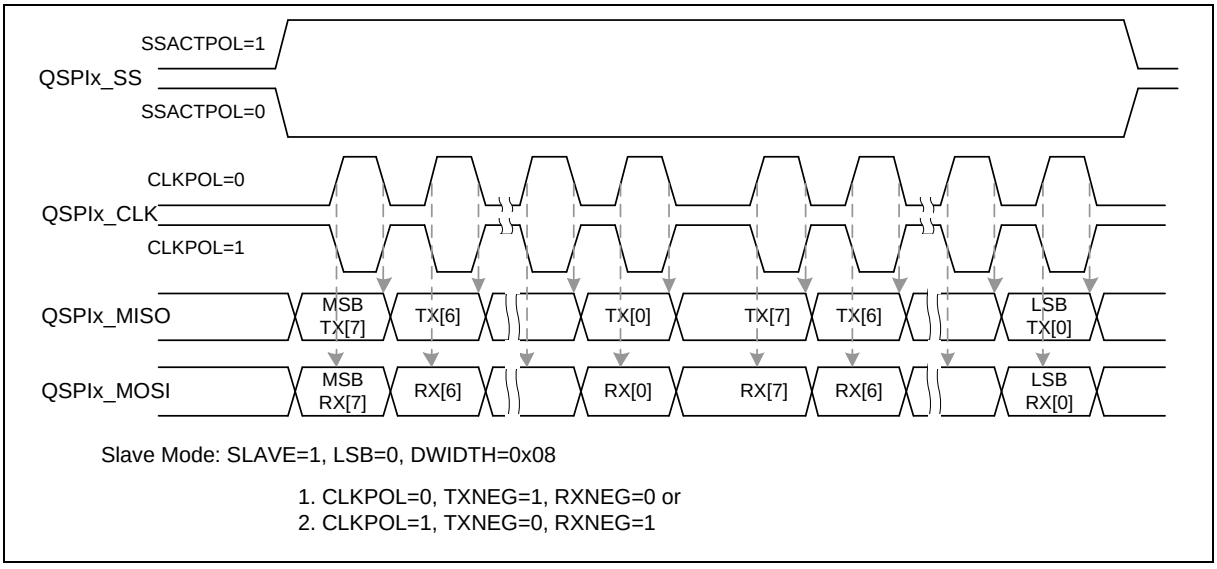


Figure 6.15-28 QSPI Timing in Slave Mode

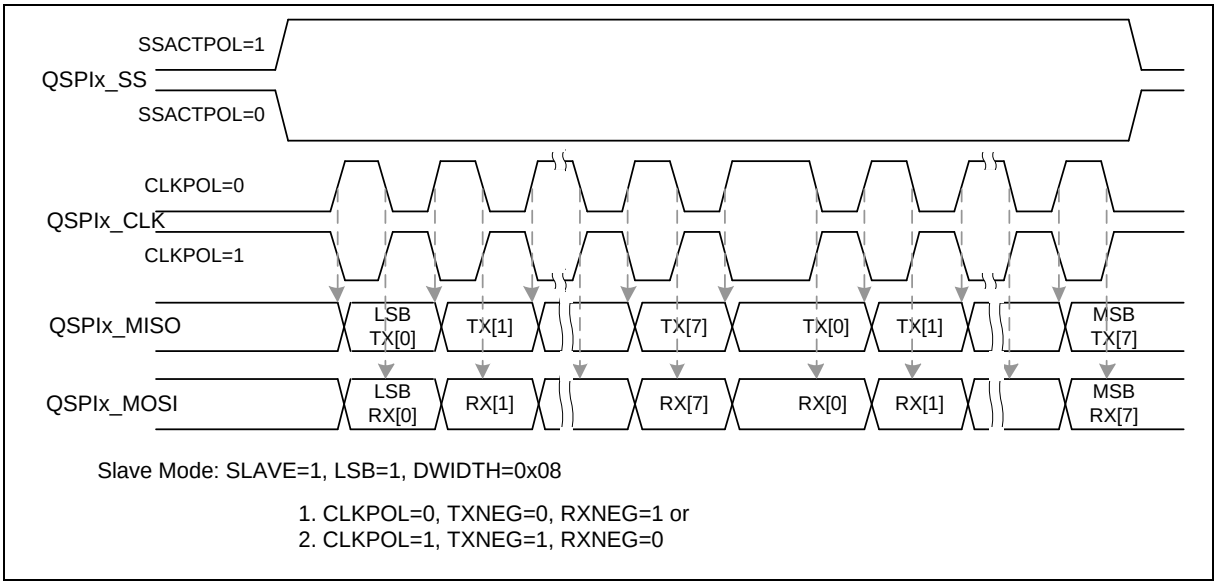


Figure 6.15-29 QSPI Timing in Slave Mode (Alternate Phase of QSPIx_CLK)

6.15.7 Programming Examples

Example 1:

The QSPI 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 QSPI bus clock.
- Data bit is driven on negative edge of QSPI bus clock.
- Data is transferred from MSB first.
- QSPI bus clock is idle at low state.
- Only one byte of data to be transmitted/received in a transaction.
- Uses the first QSPI 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 (QSPIx_CLKDIV [8:0]) to determine the output frequency of QSPI clock.
2. Write the QSPIx_SSCTL register a proper value for the related settings of Master mode:
   1) Clear AUTOSS (QSPIx_SSCTL[3]) to 0 to disable the Automatic Slave Selection function.
   2) Configure slave selection signal as active low by clearing SSACTPOL (QSPIx_SSCTL[2]) to 0.
   3) Enable slave selection signal by setting SS (QSPIx_SSCTL[0]) to 1 to activate the off-chip slave device.
3. Write the related settings into the QSPIx_CTL register to control the QSPI master actions.
   1) Configure this QSPI controller as master device by setting SLAVE (QSPIx_CTL[18]) to 0.
   2) Force the QSPI clock idle state at low by clearing CLKPOL (QSPIx_CTL[3]) to 0.
   3) Select data transmitted on negative edge of QSPI bus clock by setting TXNEG (QSPIx_CTL[2]) to 1.
   4) Select data latched on positive edge of QSPI bus clock by clearing RXNEG

(QSPIx_CTL[1]) to 0.

5) Set the bit length of a transaction as 8-bit in DWIDTH bit field (QSPIx_CTL[12:8] = 0x08).

6) Set MSB transfer first by clearing LSB (QSPIx_CTL[13]) to 0.

4. Set SPIEN (QSPIx_CTL[0]) to 1 to enable the data transfer with the QSPI interface.
5. If this QSPI master attempts to transmit (write) one byte data to the off-chip slave device, write the byte data that will be transmitted into the QSPIx_TX register.
6. Waiting for QSPI interrupt if the UNITIEN (QSPIx_CTL[17]) is set to 1, or just polling the unit transfer interrupt flag UNITIF (QSPIx_STATUS[1]).
7. Read out the received one byte data from QSPIx_RX register.
8. Go to 5) to continue another data transfer or set SS (QSPIx_SSCTL[0]) to 0 to inactivate the off-chip slave device.

Example 2:

The QSPI 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 QSPI slave controller through the QSPI interface with the following specifications:

- Data bit is latched on positive edge of QSPI bus clock.
- Data bit is driven on negative edge of QSPI bus clock.
- Data is transferred from LSB first.
- QSPI 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 QSPIx_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 (QSPIx_SSCTL[2]) to 1.
3. Write the related settings into the QSPIx_CTL register to control this QSPI slave actions

   1) Set the QSPI controller as slave device by setting SLAVE (QSPIx_CTL[18]) to 1.

   2) Select the QSPI clock idle state at high by setting CLKPOL (QSPIx_CTL[3]) to 1.

   3) Select data transmitted on negative edge of QSPI bus clock by setting TXNEG (QSPIx_CTL[2]) to 1.

   4) Select data latched on positive edge of QSPI bus clock by clearing RXNEG (QSPIx_CTL[1]) to 0.

   5) Set the bit length of a transaction as 8-bit in DWIDTH bit field (QSPIx_CTL[12:8] = 0x08).

4. Set LSB transfer first by setting LSB (QSPIx_CTL[13]) to 1.
5. Set the SPIEN (QSPIx_CTL[0]) to 1. Wait for the slave select trigger input and QSPI clock input from the off-chip master device to start the data transfer.
6. If this QSPI 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 QSPIx_TX register.
7. If this QSPI 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 QSPIx_TX register does not need to be updated by software.

8. Waiting for QSPI interrupt if the UNITIEN (QSPIx_CTL[17]) is set to 1, or just polling the unit transfer interrupt flag UNITIF (QSPIx_STATUS[1]).
9. Read out the received one byte data from QSPIx_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
QSPI Base Address: **QSPIx_BA = 0x4006_0000**				
QSPIx_CTL	QSPIx_BA+0x00	R/W	QSPI Control Register	0x0000_0034
QSPIx_CLKDIV	QSPIx_BA+0x04	R/W	QSPI Clock Divider Register	0x0000_0000
QSPIx_SSCTL	QSPIx_BA+0x08	R/W	QSPI Slave Select Control Register	0x0000_0000
QSPIx_PDMACTL	QSPIx_BA+0x0C	R/W	QSPI PDMA Control Register	0x0000_0000
QSPIx_FIFOCTL	QSPIx_BA+0x10	R/W	QSPI FIFO Control Register	0x4400_0000
QSPIx_STATUS	QSPIx_BA+0x14	R/W	QSPI Status Register	0x0005_0110
QSPIx_TX	QSPIx_BA+0x20	W	QSPI Data Transmit Register	0x0000_0000
QSPIx_RX	QSPIx_BA+0x30	R	QSPI Data Receive Register	0x0000_0000

6.15.9 Register Description

QSPI Control Register (QSPIx_CTL)

Register	Offset	R/W	Description	Reset Value
QSPIx_CTL	QSPIx_BA+0x00	R/W	QSPI Control Register	0x0000_0034

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved	QUADIOEN	DUALIOEN	DATDIR	REORDER	SLAVE	UNITIEN	TWOBIT
15	14	13	12	11	10	9	8
RXONLY	HALFDPX	LSB	DWIDTH				
7	6	5	4	3	2	1	0
SUSPITV				CLKPOL	TXNEG	RXNEG	SPIEN

Bits	Description	
[31:23]	**Reserved**	Reserved.
[22]	**QUADIOEN**	**Quad I/O Mode Enable Bit** 0 = Quad I/O mode Disabled. 1 = Quad I/O mode Enabled.
[21]	**DUALIOEN**	**Dual I/O Mode Enable Bit** 0 = Dual I/O mode Disabled. 1 = Dual I/O mode Enabled.
[20]	**DATDIR**	**Data Port Direction Control** This bit is used to select the data input/output direction in half-duplex transfer and Dual/Quad transfer 0 = QSPI data is input direction. 1 = QSPI 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 = QSPI unit transfer interrupt Disabled. 1 = QSPI unit transfer interrupt Enabled.
[16]	**TWOBIT**	**2-bit Transfer Mode Enable Bit** 0 = 2-bit Transfer mode Disabled.

		1 = 2-bit Transfer mode Enabled. **Note:** When 2-bit Transfer mode is enabled, the first serial transmitted bit data is from the first FIFO buffer data, and the 2nd serial transmitted bit data is from the second FIFO buffer data. As the same as transmitted function, the first received bit data is stored into the first FIFO buffer and the 2nd received bit data is stored into the second FIFO buffer at the same time.
[15]	**RXONLY**	**Receive-only Mode Enable Bit (Master Only)** This bit field is only available in Master mode. In receive-only mode, QSPI Master will generate QSPI bus clock continuously for receiving data bit from QSPI slave device and assert the BUSY status. 0 = Receive-only mode Disabled. 1 = Receive-only mode Enabled.
[14]	**HALFDPX**	**QSPI Half-duplex Transfer Enable Bit** This bit is used to select full-duplex or half-duplex for QSPI transfer. The bit field DATDIR (QSPIx_CTL[20]) can be used to set the data direction in half-duplex transfer. 0 = QSPI operates in full-duplex transfer. 1 = QSPI 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 QSPIx TX register, is sent first to the QSPI data output pin, and the first bit received from the QSPI data input pin will be put in the LSB position of the RX register (bit 0 of QSPIx_RX).
[12:8]	**DWIDTH**	**Data Width** This field specifies how many bits can be transmitted / received in one transaction. The minimum bit length is 8 bits and can up to 32 bits. DWIDTH = 0x08 …. 8 bits. DWIDTH = 0x09 …. 9 bits. …… DWIDTH = 0x1F …. 31 bits. DWIDTH = 0x00 …. 32 bits.
[7:4]	**SUSPITV**	**Suspend Interval (Master Only)** The four bits provide configurable suspend interval between two successive transmit/receive transaction in a transfer. The definition of the suspend interval is the interval between the last clock edge of the preceding transaction word and the first clock edge of the following transaction word. The default value is 0x3. The period of the suspend interval is obtained according to the following equation. (SUSPITV + 0.5) * period of QSPICLK clock cycle Example: SUSPITV = 0x0 …. 0.5 QSPICLK clock cycle. SUSPITV = 0x1 …. 1.5 QSPICLK clock cycle. …… SUSPITV = 0xE …. 14.5 QSPICLK clock cycle. SUSPITV = 0xF …. 15.5 QSPICLK clock cycle.
[3]	**CLKPOL**	**Clock Polarity** 0 = QSPI bus clock is idle low. 1 = QSPI bus clock is idle high.
[2]	**TXNEG**	**Transmit on Negative Edge** 0 = Transmitted data output signal is changed on the rising edge of QSPI bus clock. 1 = Transmitted data output signal is changed on the falling edge of QSPI bus clock.

[1]	**RXNEG**	**Receive on Negative Edge** 0 = Received data input signal is latched on the rising edge of QSPI bus clock. 1 = Received data input signal is latched on the falling edge of QSPI bus clock.
[0]	**SPIEN**	**QSPI 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 QSPIx_CTL, QSPIx_CLKDIV, QSPIx_SSCTL and QSPIx_FIFOCTL registers, user shall clear the SPIEN (QSPIx_CTL[0]) and confirm the SPIENSTS (QSPIx_STATUS[15]) is 0.

QSPI Clock Divider Register (QSPIx_CLKDIV)

Register	Offset	R/W	Description	Reset Value
QSPIx_CLKDIV	QSPIx_BA+0x04	R/W	QSPI 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_{qspi_eclk}, and the QSPI bus clock of QSPI Master. The frequency is obtained according to the following equation. $f_{qspi\_eclk} = \frac{f_{qspi\_clock\_src}}{(DIVIDER+1)}$ where $f_{qspi\_clock\_src}$ is the peripheral clock source, which is defined in the clock control register, CLK_CLKSEL2. **Note:** The time interval must be larger than or equal 8 peripheral clock cycles between releasing QSPI 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.

QSPI Slave Select Control Register (QSPIx_SSCTL)

Register	Offset	R/W	Description	Reset Value
QSPIx_SSCTL	QSPIx_BA+0x08	R/W	QSPI Slave Select Control Register	0x0000_0000

31	30	29	28	27	26	25	24
SLVTOCNT							
23	22	21	20	19	18	17	16
SLVTOCNT							
15	14	13	12	11	10	9	8
Reserved		SSINAIEN	SSACTIEN	Reserved		SLVURIEN	SLVBEIEN
7	6	5	4	3	2	1	0
Reserved	SLVTORST	SLVTOIEN	SLV3WIRE	AUTOSS	SSACTPOL	Reserved	SS

Bits	Description	
[31:16]	**SLVTOCNT**	**Slave Mode Time-out Period** In Slave mode, these bits indicate the time-out period when there is bus clock input during slave select active. The clock source of the time-out counter is Slave peripheral clock. If the value is 0, it indicates the slave mode time-out function is disabled.
[15:14]	**Reserved**	Reserved.
[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]	**Reserved**	Reserved.
[6]	**SLVTORST**	**Slave Mode Time-out Reset Control** 0 = When Slave mode time-out event occurs, the TX and RX control circuit will not be reset. 1 = When Slave mode time-out event occurs, the TX and RX control circuit will be reset by hardware.
[5]	**SLVTOIEN**	**Slave Mode Time-out Interrupt Enable Bit** 0 = Slave mode time-out interrupt Disabled. 1 = Slave mode time-out interrupt Enabled.

[4]	**SLV3WIRE**	**Slave 3-wire Mode Enable Bit** In Slave 3-wire mode, the QSPI controller can work with 3-wire interface including QSPIx_CLK, QSPIx_MISO and QSPIx_MOSI pins. 0 = 4-wire bi-direction interface. 1 = 3-wire bi-direction interface.
[3]	**AUTOSS**	**Automatic Slave Selection Function Enable Bit (Master Only)** 0 = Automatic slave selection function Disabled. Slave selection signal will be asserted/de-asserted according to SS (QSPIx_SSCTL[0]). 1 = Automatic slave selection function Enabled.
[2]	**SSACTPOL**	**Slave Selection Active Polarity** This bit defines the active polarity of slave selection signal (QSPIx_SS). 0 = The slave selection signal QSPIx_SS is active low. 1 = The slave selection signal QSPIx_SS is active high.
[1]	**Reserved**	Reserved.
[0]	**SS**	**Slave Selection Control (Master Only)** If AUTOSS bit is cleared to 0, 0 = set the QSPIx_SS line to inactive state. 1 = set the QSPIx_SS line to active state. If the AUTOSS bit is set to 1, 0 = Keep the QSPIx_SS line at inactive state. 1 = QSPIx_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 QSPIx_SS is specified in SSACTPOL (QSPIx_SSCTL[2]).

QSPI PDMA Control Register (QSPIx_PDMACTL)

Register	Offset	R/W	Description	Reset Value
QSPIx_PDMACTL	QSPIx_BA+0x0C	R/W	QSPI 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 QSPI controller. This bit will be automatically cleared to 0.
[1]	**RXPDMAEN**	**Receive PDMA Enable Bit** 0 = Receive PDMA function Disabled. 1 = Receive PDMA function Enabled.
[0]	**TXPDMAEN**	**Transmit PDMA Enable Bit** 0 = Transmit PDMA function Disabled. 1 = Transmit PDMA function Enabled. **Note:** In QSPI 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.

QSPI FIFO Control Register (QSPIx_FIFOCTL)

Register	Offset	R/W	Description	Reset Value
QSPIx_FIFOCTL	QSPIx_BA+0x10	R/W	QSPI FIFO Control Register	0x4400_0000

31	30	29	28	27	26	25	24
Reserved	TXTH			Reserved	RXTH		
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						TXFBCLR	RXFBCLR
7	6	5	4	3	2	1	0
TXUFIEN	TXUFPOL	RXOVIEN	RXTOIEN	TXTHIEN	RXTHIEN	TXRST	RXRST

Bits	Description	
[31]	**Reserved**	Reserved.
[30:28]	**TXTH**	**Transmit FIFO Threshold** If the valid data count of the transmit FIFO buffer is less than or equal to the TXTH setting, the TXTHIF bit will be set to 1, else the TXTHIF bit will be cleared to 0.
[27]	**Reserved**	Reserved.
[26:24]	**RXTH**	**Receive FIFO Threshold** If the valid data count of the receive FIFO buffer is larger than the RXTH setting, the RXTHIF bit will be set to 1, else the RXTHIF bit will be cleared to 0.
[23:10]	**Reserved**	Reserved.
[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 (QSPIx_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 QSPI data out is keep 0 if there is TX underflow event in Slave mode. 1 = The QSPI 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:** When TX underflow event occurs, QSPIx_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 QSPIx_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**	**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 (QSPIx_STATUS[23]) to check if reset is accomplished or not. **Note:** If TX underflow event occurs in QSPI Slave mode, this bit can be used to make QSPI 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 (QSPIx_STATUS[23]) to check if reset is accomplished or not.

QSPI Status Register (QSPIx_STATUS)

Register	Offset	R/W	Description	Reset Value
QSPIx_STATUS	QSPIx_BA+0x14	R/W	QSPI Status Register	0x0005_0110

31	30	29	28	27	26	25	24
TXCNT				RXCNT			
23	22	21	20	19	18	17	16
TXRXRST	Reserved			TXUFIF	TXTHIF	TXFULL	TXEMPTY
15	14	13	12	11	10	9	8
SPIENSTS	Reserved		RXTOIF	RXOVIF	RXTHIF	RXFULL	RXEMPTY
7	6	5	4	3	2	1	0
SLVURIF	SLVBEIF	SLVTOIF	SSLINE	SSINAIF	SSACTIF	UNITIF	BUSY

Bits	Description	
[31:28]	**TXCNT**	**Transmit FIFO Data Count (Read Only)** This bit field indicates the valid data count of transmit FIFO buffer.
[27:24]	**RXCNT**	**Receive FIFO Data Count (Read Only)** This bit field indicates the valid data count of receive FIFO buffer.
[23]	**TXRXRST**	**TX or RX Reset Status (Read Only)** 0 = The reset function of TXRST or RXRST is done. 1 = Doing the reset function of TXRST or RXRST. **Note:** Both the reset operations of TXRST and RXRST need 3 system clock cycles + 2 peripheral clock cycles. User can check the status of this bit to monitor the reset function is doing or done.
[22:20]	**Reserved**	Reserved.
[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**	**QSPI Enable Status (Read Only)** 0 = QSPI controller Disabled. 1 = QSPI controller Enabled. **Note:** The QSPI peripheral clock is asynchronous with the system clock. In order to make sure the QSPI control logic is disabled, this bit indicates the real status of QSPI 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 QSPI peripheral clock periods in Master mode or over 576 QSPI 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]	**SLVTOIF**	**Slave Time-out Interrupt Flag**

		When the slave select is active and the value of SLVTOCNT is not 0, if the bus clock is detected, the slave time-out counter in QSPI controller logic will be started. When the value of time-out counter is greater than or equal to the value of SLVTOCNT (QSPIx_SSCTL[31:16]) before one transaction is done, the slave time-out interrupt event will be asserted. 0 = Slave time-out is not active. 1 = Slave time-out is active. **Note:** This bit will be cleared by writing 1 to it.
[4]	**SSLINE**	**Slave Select Line Bus Status (Read Only)** 0 = The slave select line status is 0. 1 = The slave select line status is 1. **Note:** This bit is only available in Slave mode. If SSACTPOL (QSPIx_SSCTL[2]) is set 0, and the SSLINE is 1, the QSPI 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 = QSPI controller has finished one unit transfer. **Note:** This bit will be cleared by writing 1 to it.
[0]	**BUSY**	**Busy Status (Read Only)** 0 = QSPI controller is in idle state. 1 = QSPI controller is in busy state. The following lists the bus busy conditions: ● SPIEN (QSPIx_CTL[0]) = 1 and TXEMPTY = 0. ● For QSPI Master mode, SPIEN (QSPIx_CTL[0]) = 1 and TXEMPTY = 1 but the current transaction is not finished yet. ● For QSPI Master mode, SPIEN (QSPIx_CTL[0]) = 1 and RXONLY = 1. ● For QSPI Slave mode, SPIEN (QSPIx_CTL[0]) = 1 and there is serial clock input into the QSPI core logic when slave select is active. ● For QSPI Slave mode, SPIEN (QSPIx_CTL[0]) = 1 and the transmit buffer or transmit shift register is not empty even if the slave select is inactive.

QSPI Data Transmit Register (QSPIx_TX)

Register	Offset	R/W	Description	Reset Value
QSPIx_TX	QSPIx_BA+0x20	W	QSPI Data Transmit Register	0x0000_0000

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

Bits	Description	
[31:0]	TX	**Data Transmit Register** The data transmit registers pass through the transmitted data into the 8-level transmit FIFO buffers. The number of valid bits depends on the setting of DWIDTH (QSPIx_CTL[12:8]). If DWIDTH is set to 0x08, the bits TX[7:0] will be transmitted. If DWIDTH is set to 0x00, the QSPI controller will perform a 32-bit transfer. **Note:** In Master mode, QSPI controller will start to transfer the QSPI bus clock after 1 APB clock and 6 peripheral clock cycles after user writes to this register.

QSPI Data Receive Register (QSPIx_RX)

Register	Offset	R/W	Description	Reset Value
QSPIx_RX	QSPIx_BA+0x30	R	QSPI 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 8-level FIFO buffers in this controller. The data receive register holds the data received from QSPI data input pin. If the RXEMPTY (QSPIx_STATUS[8) is not set to 1, the receive FIFO buffers can be accessed through software by reading this register.

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
- 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
- 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

Section	Sub-Section	M031xB/C/D/E M032xB/C/D/E	M031xG/I M032xG/I
6.16.5 Functional Description	6.16.5.2 Operation Modes - Bus Management (SMBus/PMBus Compatiable) - Device Identification – Slave Address - Bus Protocols - Address Resolution Protocol (ARP) - Received Command and Data acknowledge control - Host Notify Protocol - Bus Management Alert - Packet Error Checking - Time-out - Bus Management Time-out: - Bus Clock Low Time-out:	-	●

Section	Sub-Section	M031xB/C/D/E M032xB/C/D/E	M031xG/I M032xG/I
	- Bus Idle Detection		
Register Description	I2C Bus Manage Control Register (I2C_BUSCTL)	-	●
	I2C Bus Management Timer Control Register (I2C_BUSTCTL)	-	●
	I2C Bus Management Status Register (I2C_BUSSTS)	-	●
	I2C Byte Number Register (I2C_PKTSIZE)	-	●
	I2C PEC Value Register (I2C_PKTCRC)	-	●
	I2C Bus Management Timer Register (I2C_BUSTOUT)	-	●
	I2C Clock Low Timer Register (I2C_CLKTOUT)	-	●

Table 6.16-1 I^{2}C Feature Comparison Table at Different Chip

6.16.3 Block Diagram

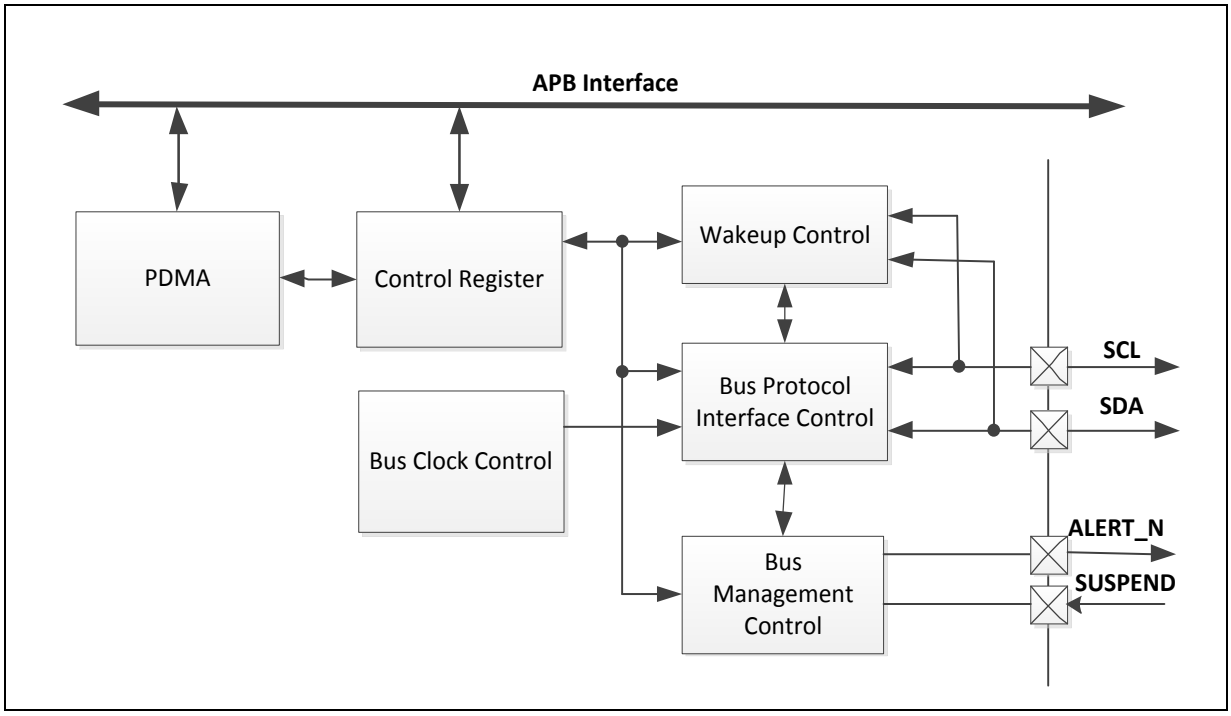


Figure 6.16-1 I^{2}C Controller Block Diagram

6.16.4 Basic Configuration

6.16.4.1 I2C0 basic configurations

- Clock source Configuration
  - Enable I2C0 peripheral clock in I2C0CKEN (CLK_APBCLK0[8]).
- Reset Configuration
  - Reset I2C0 controller in I2C0RST (SYS_IPRST1[8]).

6.16.4.2 I2C1 Basic Configurations

- Clock Source Configuration
  - Enable I2C1 peripheral clock in I2C1CKEN (CLK_APBCLK0[9]).
- Reset Configuration
  - Reset I2C1 controller in I2C1RST (SYS_IPRST1[9]).

6.16.5 Functional Description

On I^2C bus, data is transferred between a Master and a Slave. Data bits transfer on the SCL and SDA lines are synchronously on a byte-by-byte basis. Each data byte is 8-bit long. There is one SCL clock pulse for each data bit with the MSB being transmitted first, and an acknowledge bit follows each transferred byte. Each bit is sampled during the high period of SCL; therefore, the SDA line may be changed only during the low period of SCL and must be held stable during the high period of SCL. A transition on the SDA line while SCL is high is interpreted as a command (START or STOP). Please refer to Figure 6.16-2 for more detailed I^2C BUS Timing.

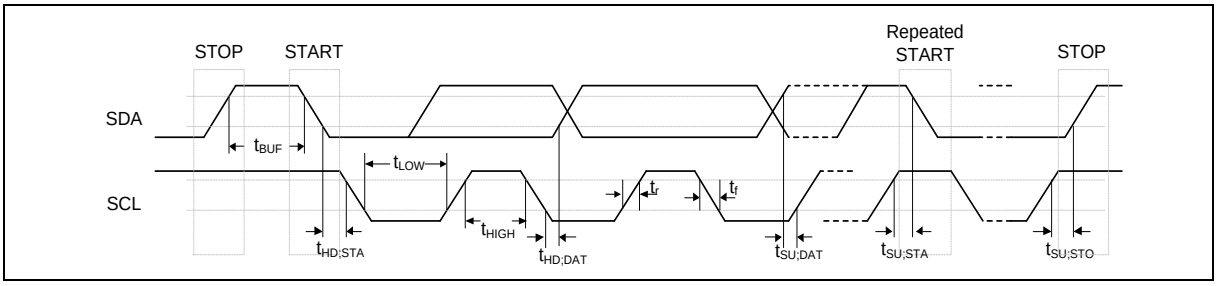


Figure 6.16-2 I^2C Bus Timing

The device's on-chip I^2C provides the serial interface that meets the I^2C bus standard mode specification. The I^2C port handles byte transfers autonomously. To enable this port, the bit I2CEN in I2C_CTL0 should be set to '1'. The I^2C hardware interfaces to the I^2C bus via two pins: SDA and SCL. When I/O pins are used as I^2C ports, user must set the pins function to I^2C in advance.

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

6.16.5.1 I^2C Protocol

Figure 6.16-3 shows the typical I^2C protocol. Normally, a standard communication consists of four parts:

- START or Repeated START signal generation
- Slave address and R/W bit transfer
- Data transfer
- STOP signal generation

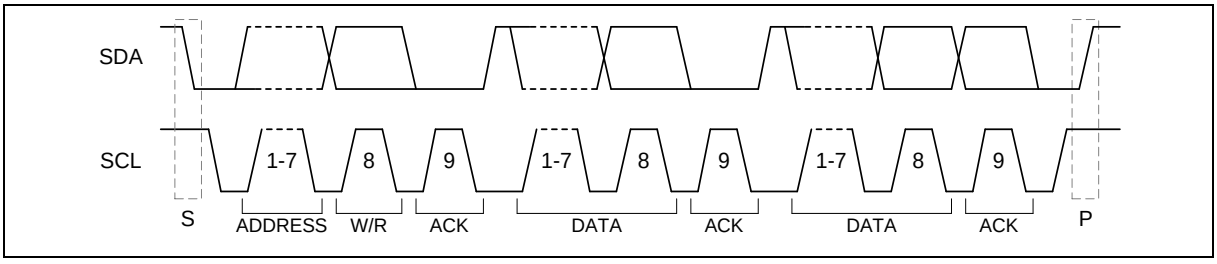


Figure 6.16-3 I^2C Protocol

- START or Repeated START signal

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

After having sent the address byte (address and read/write bit), the master may send any number of bytes followed by a stop condition. Instead of sending the stop condition it is also allowed to send another start condition again followed by an address (and of course including a read/write bit) and more data. The start condition is called as Repeat START (Sr). This is defined recursively allowing any number of start conditions to be sent. The purpose of this is to allow combined write/read operations to one or more devices without releasing the bus and thus with the guarantee that the operation is not interrupted. The controller uses this method to communicate with another slave or the same slave in a different transfer direction (e.g. from writing to a device to reading from a device) without releasing the bus.

- STOP signal

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

  Figure 6.16-4 shows the waveform of START, Repeat START and STOP.

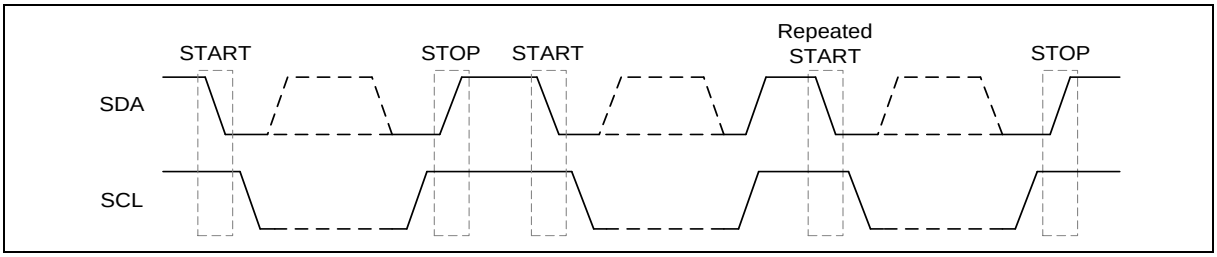


Figure 6.16-4 START and STOP Conditions

- 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 bytes. 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.

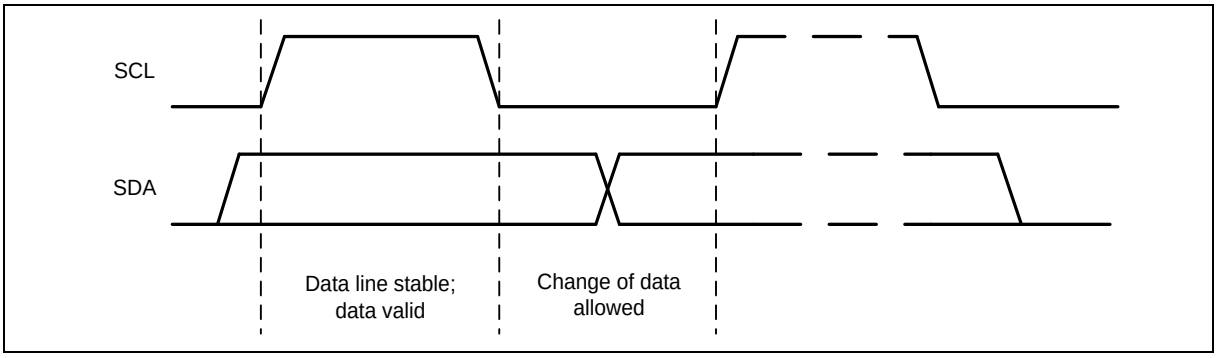


Figure 6.16-5 Bit Transfer on the I²C Bus

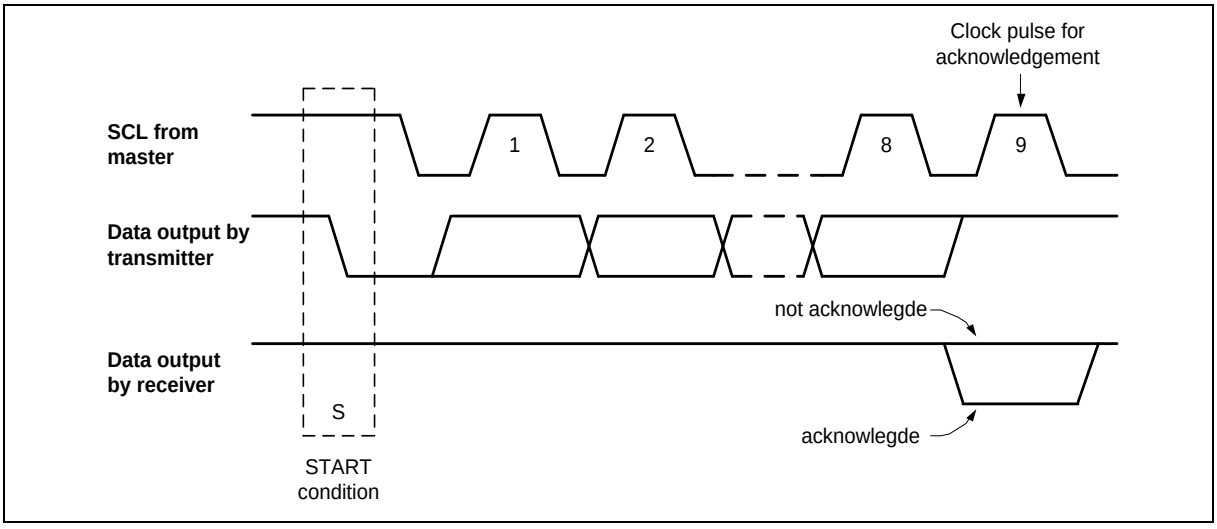


Figure 6.16-6 Acknowledge on the I²C Bus

- 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.

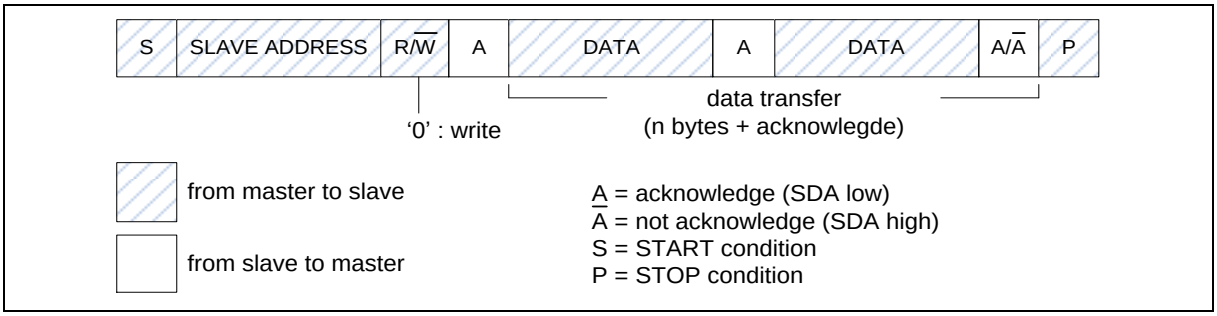


Figure 6.16-7 Master Transmits Data to Slave by 7-bit

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.

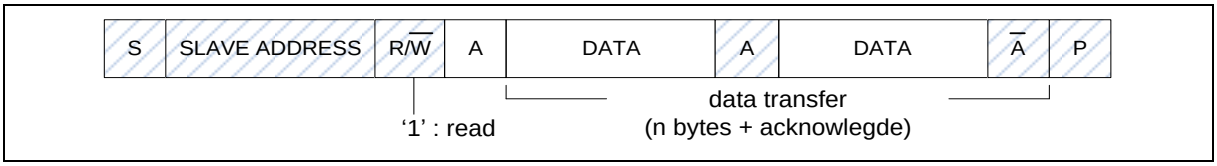


Figure 6.16-8 Master Reads Data from Slave by 7-bit

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 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 6.16-9 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.

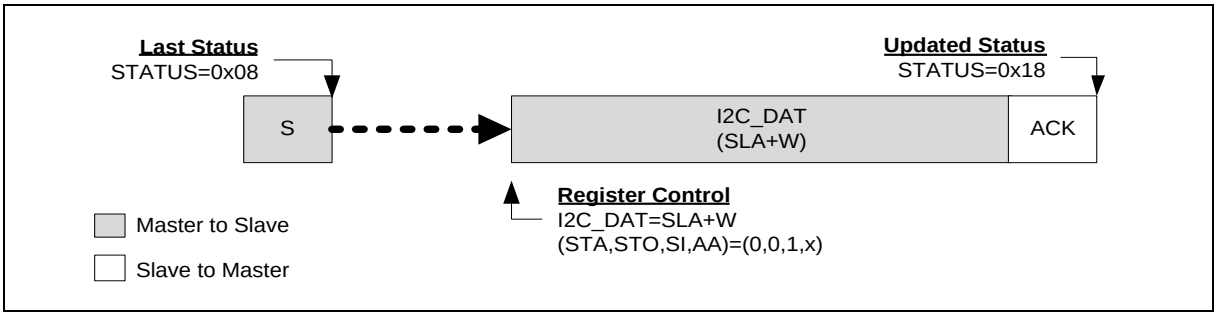


Figure 6.16-9 Control I²C Bus according to the Current I²C Status

Master Mode

In Figure 6.16-10 and Figure 6.16-11, 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 6.16-10) or Master receiver (MR) mode (Figure 6.16-11) after START signal has been sent successfully and new status code would be 0x08. Followed by START signal, user can send slave address, read/write bit, data and Repeat START, STOP to perform I^{2}C protocol.

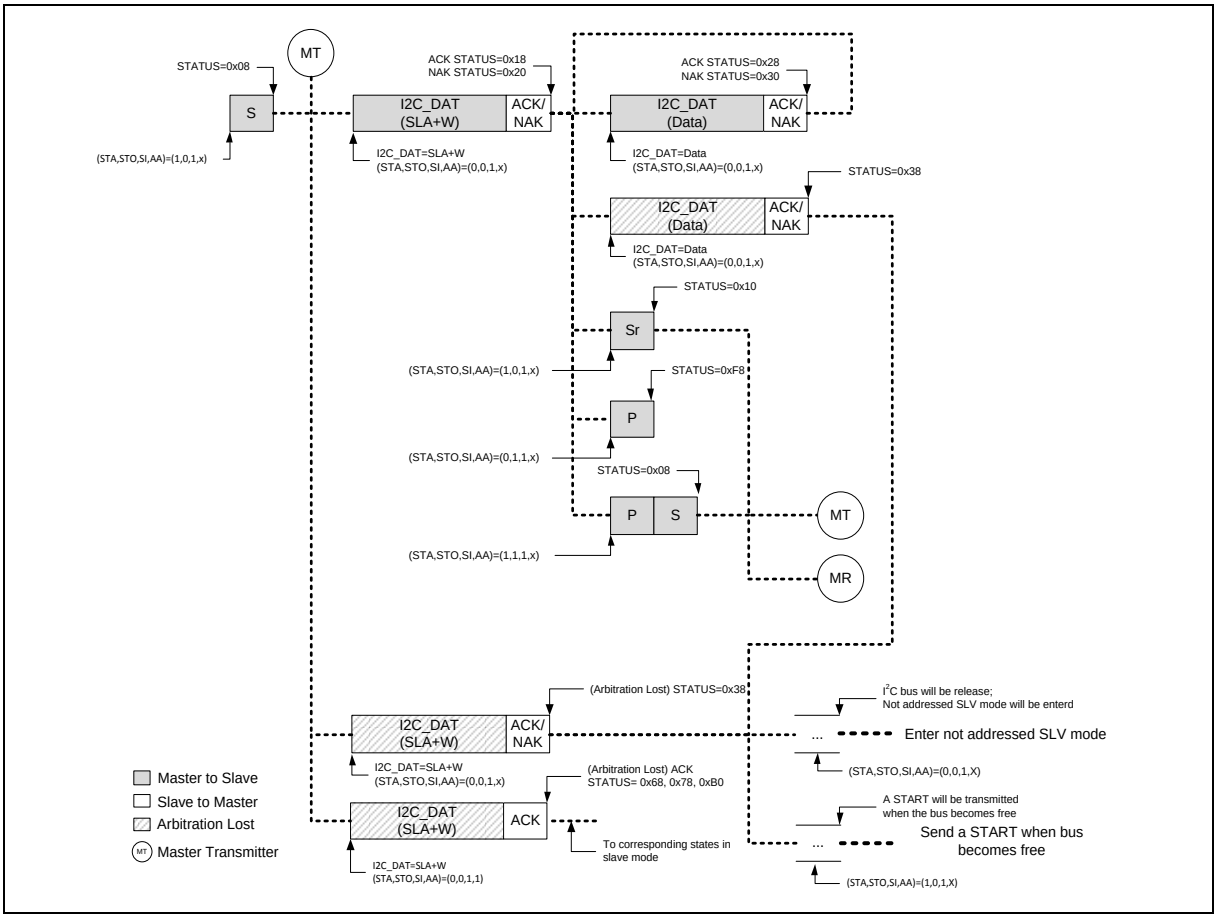


Figure 6.16-10 Master Transmitter Mode Control Flow

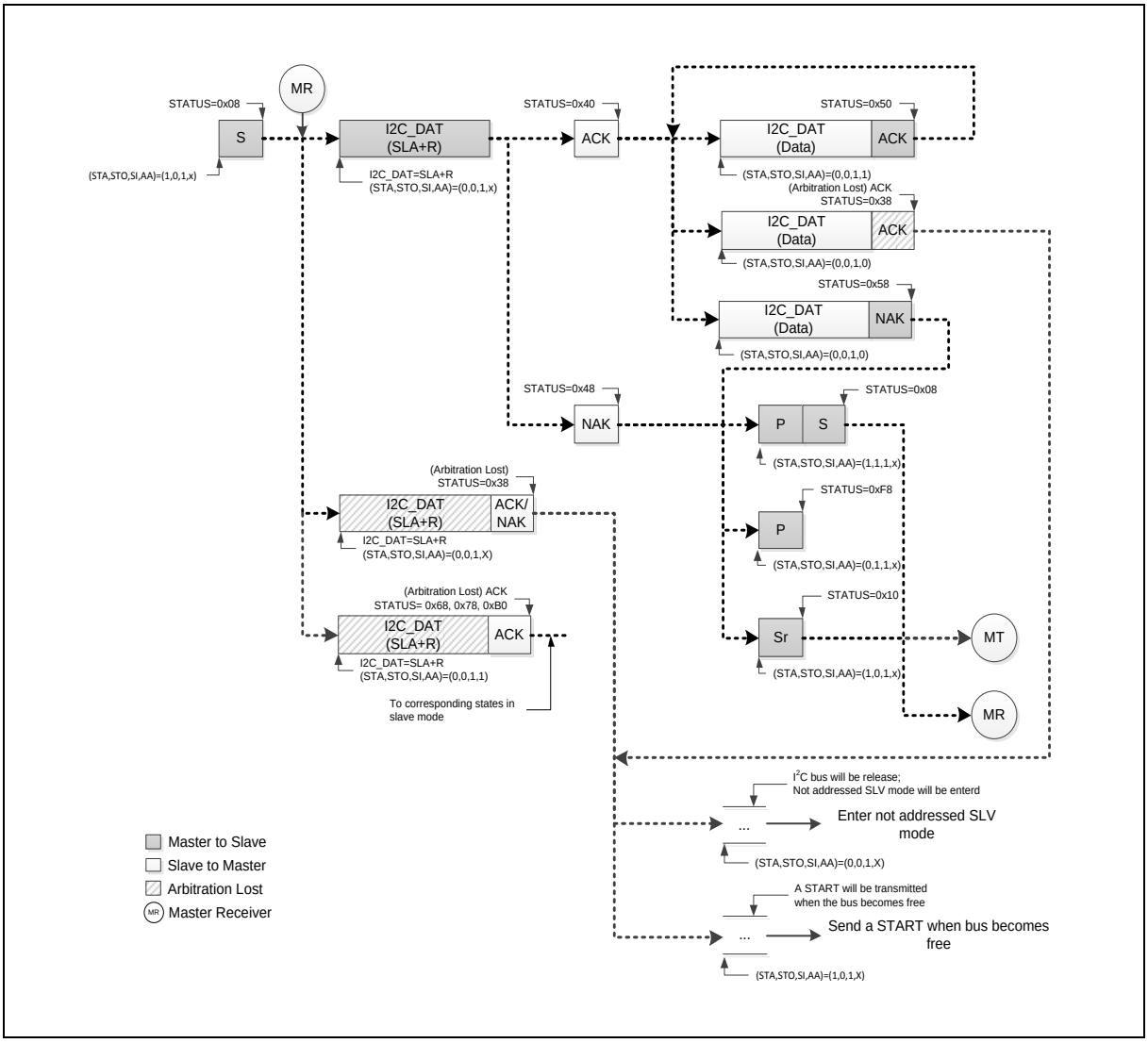


Figure 6.16-11 Master Receiver Mode Control Flow

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 6.16-12 shows all the possible flow for I²C in Slave mode. Users need to follow a proper flow (as shown in Figure 6.16-12 to implement their own I²C protocol.

If bus arbitration is lost in Master mode, I²C port switches to Slave mode immediately and can detect its own slave address in the same serial transfer. If the detected address is SLA+W (Master want to write data to Slave) after arbitration lost, the status code is 0x68. If the detected address is SLA+R (Master want to read data from Slave) after arbitration lost, the status code is 0xB0.

Note: During I²C communication, the SCL clock will be released when writing '1' to clear SI flag in Slave mode.

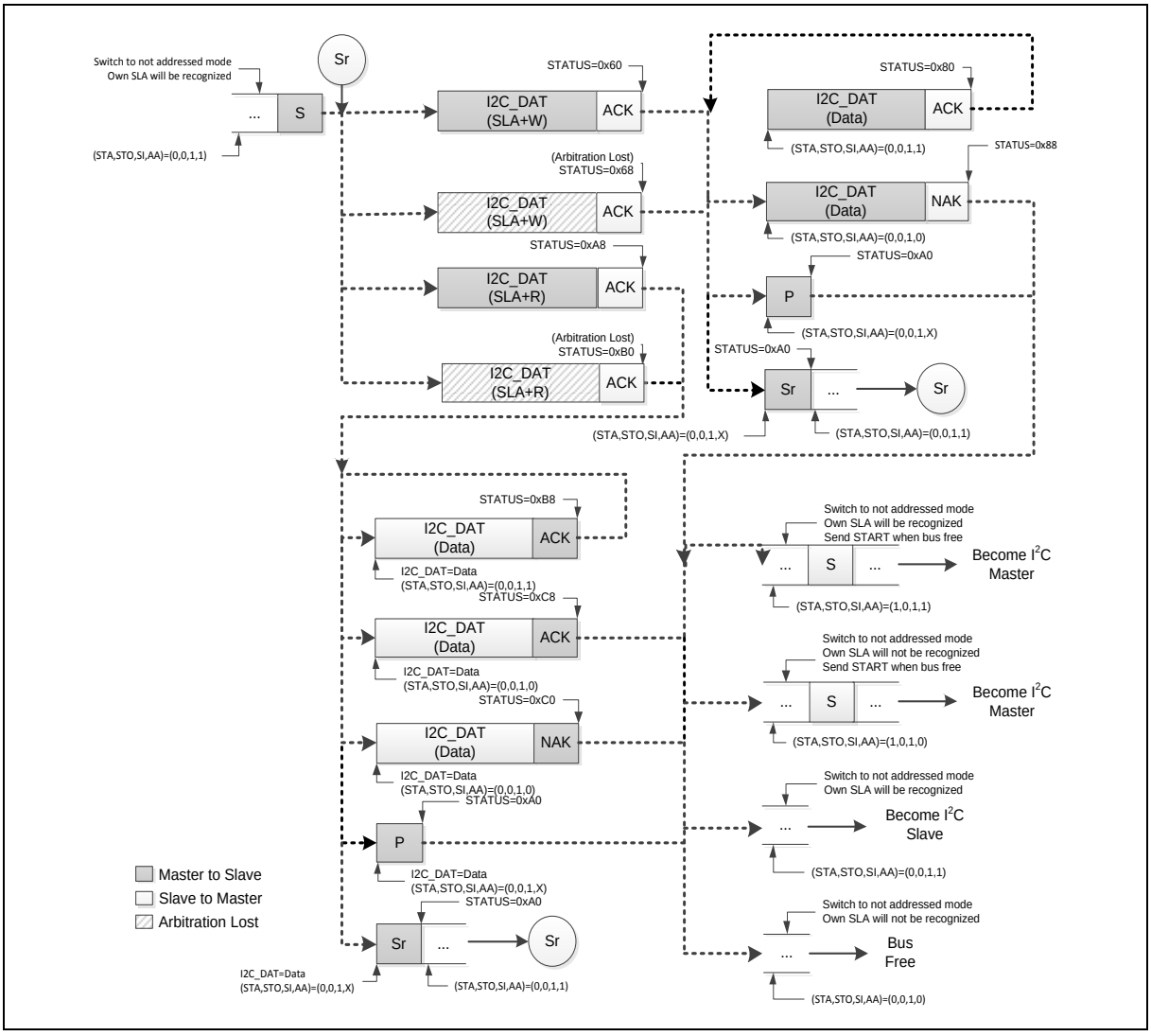


Figure 6.16-12 Slave Mode Control Flow

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_ADDR0 [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.

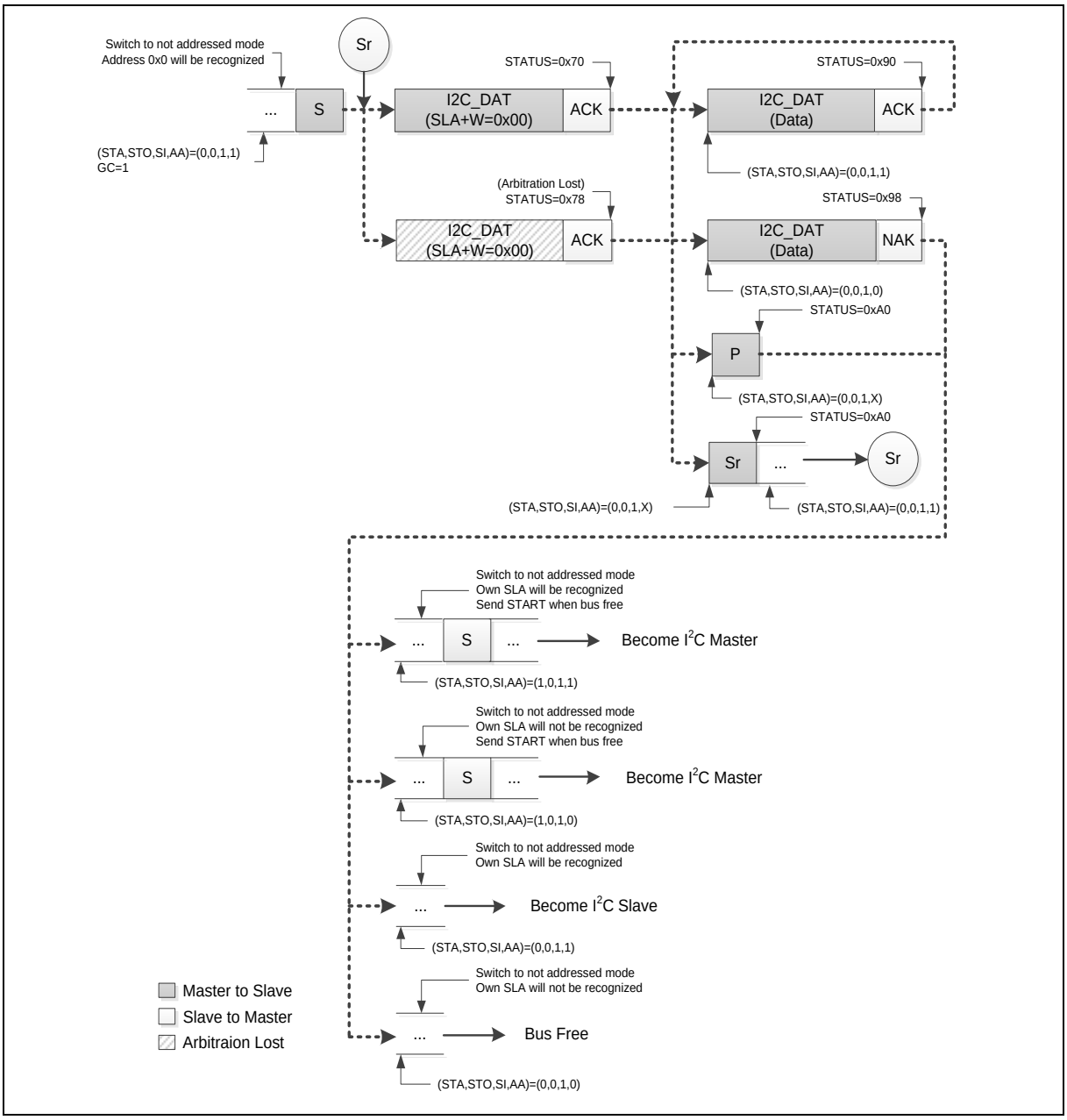


Figure 6.16-13 GC Mode

If I^2C is still receiving data in GC mode but got a STOP or Repeat START, the status code will be 0xA0. User could follow the action for status code 0x98 in above figure when getting 0xA0 status.

Note: After slave gets status of 0x98 and 0xA0, slave can switch to not address mode and own SLA will not be recognized. If entering this status, slave will not receive any I^2C signal or address from master. At this time, the I^2C controller should enter idle mode.

Multi-Master

In some applications, there are two or more masters on the same I^2C bus to access slaves, and the masters may transmit data simultaneously. The I^2C supports multi-master by including collision

detection and arbitration to prevent data corruption.

If for some reason two masters initiate command at the same time, the arbitration procedure determines which master wins and can continue with the command. Arbitration is performed on the SDA signal while the SCL signal is high. Each master checks if the SDA signal on the bus corresponds to the generated SDA signal. If the SDA signal on the bus is low but it should be high, then this master has lost arbitration. The device that has lost arbitration can generate SCL pulses until the byte ends and must then release the bus and go into slave mode. The arbitration procedure can continue until all the data is transferred. This means that in multi-master system each master must monitor the bus for collisions and act accordingly.

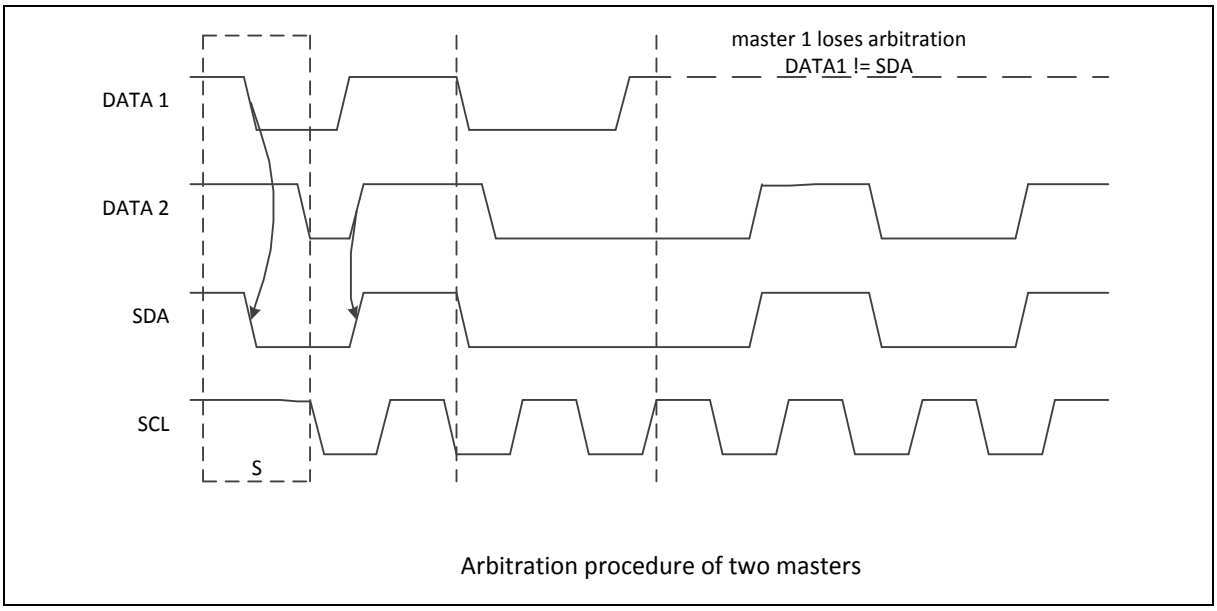


Figure 6.16-14 Arbitration Lost

- 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 6.16-2 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 or later versions)

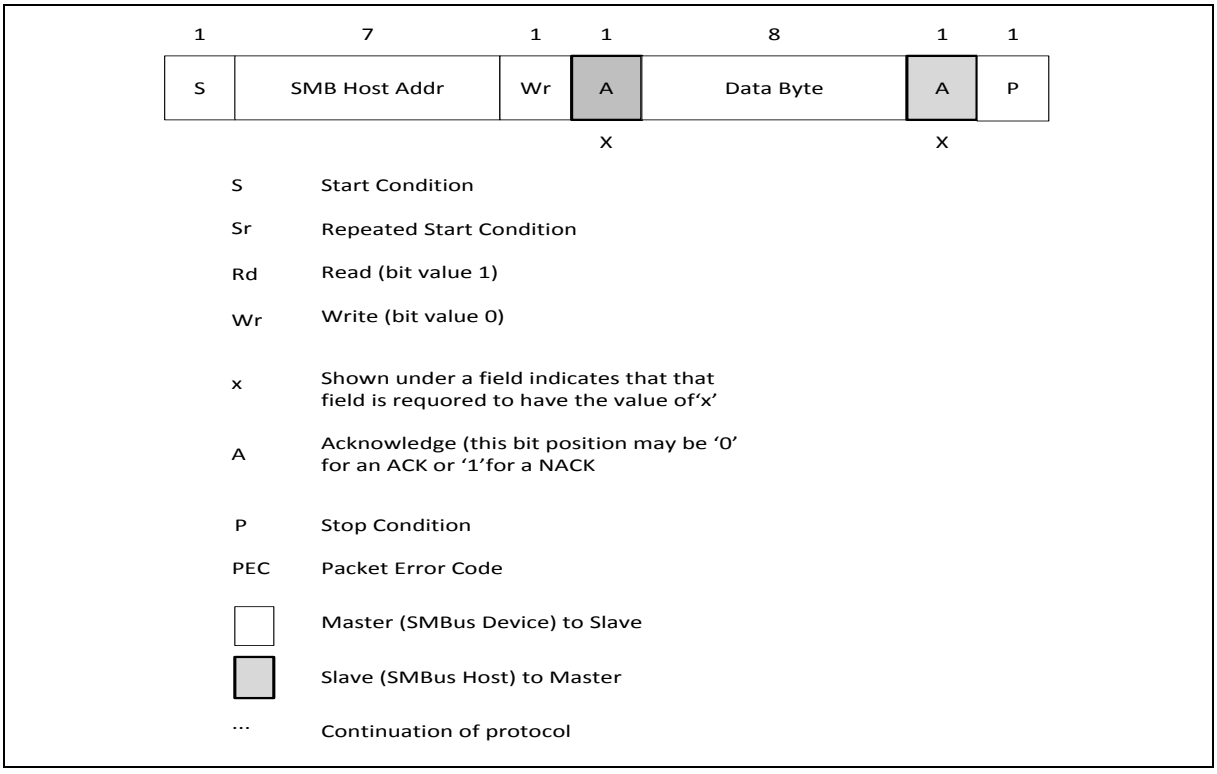


Figure 6.16-15 Bus Management Packet Protocol Diagram Element Key

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.

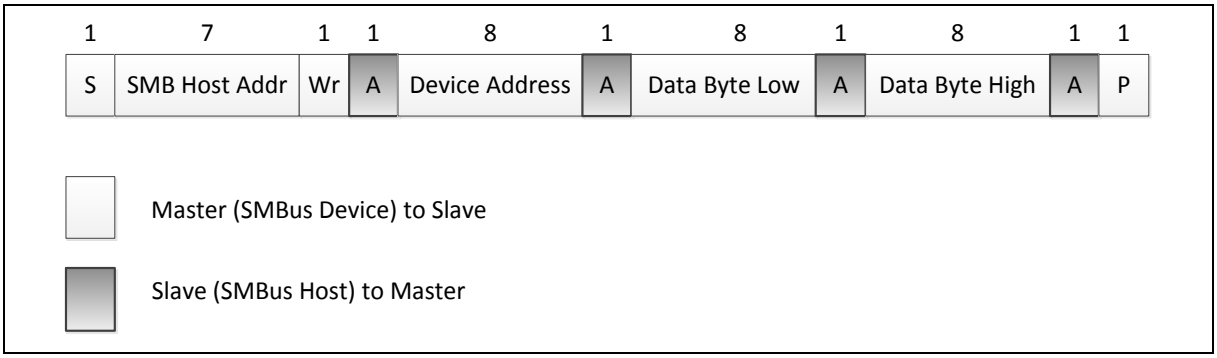


Figure 6.16-167-bit Addressable Device to Host Communication

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.

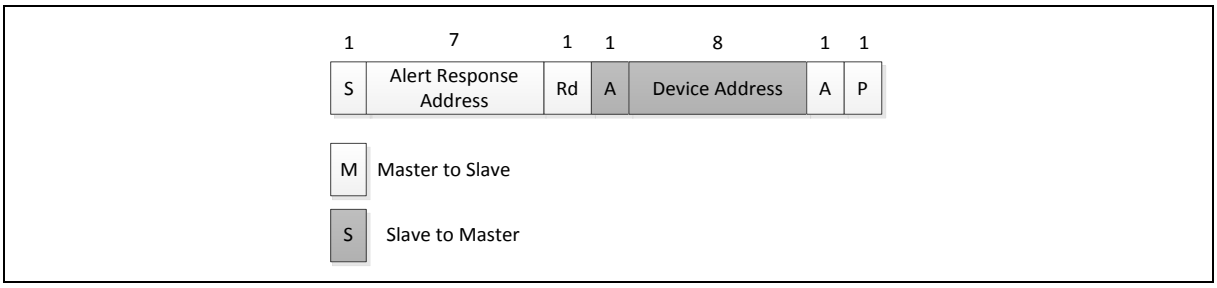


Figure 6.16-177-bit Addressable Device Responds to an ARA

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;

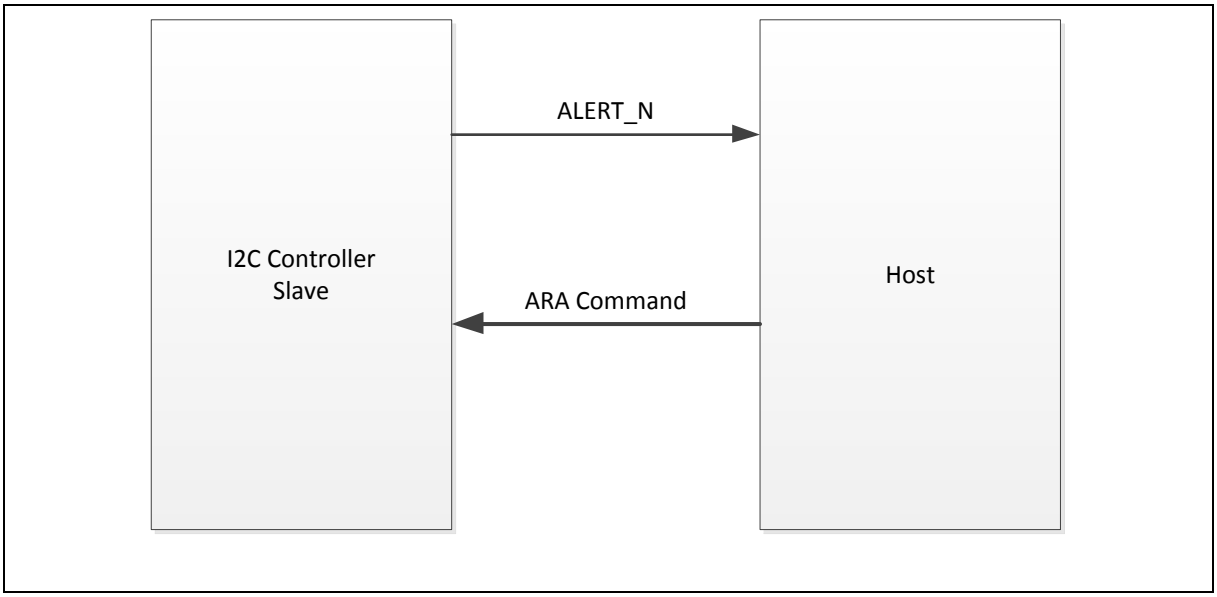


Figure 6.16-18 Bus Management ALERT function

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}.

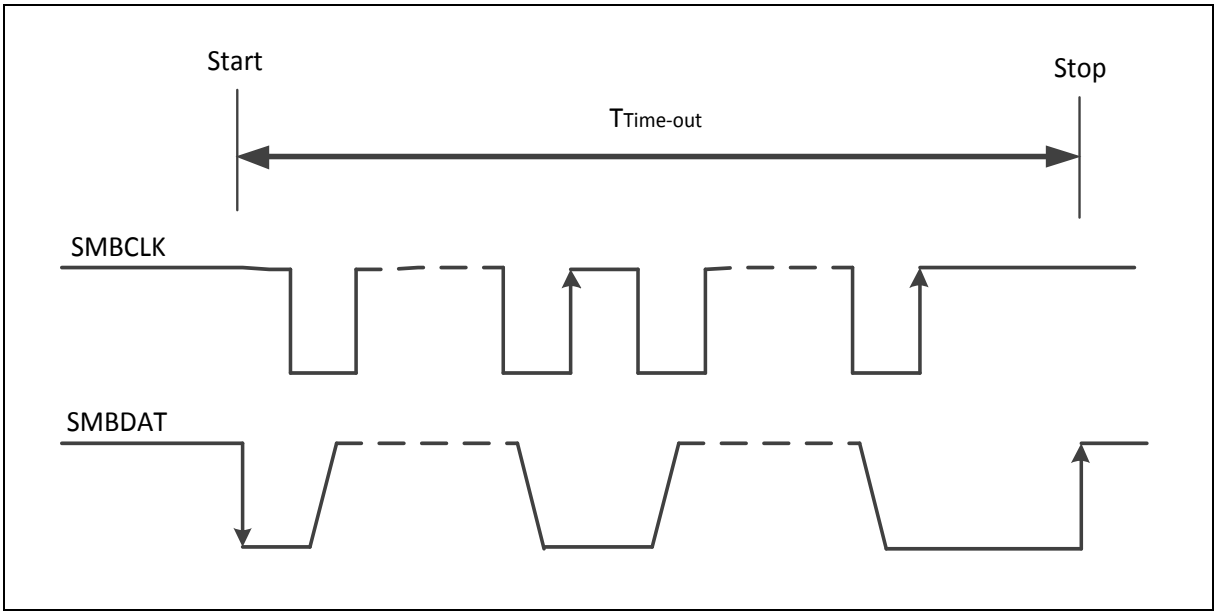


Figure 6.16-19 Bus Management Time Out Timing

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)

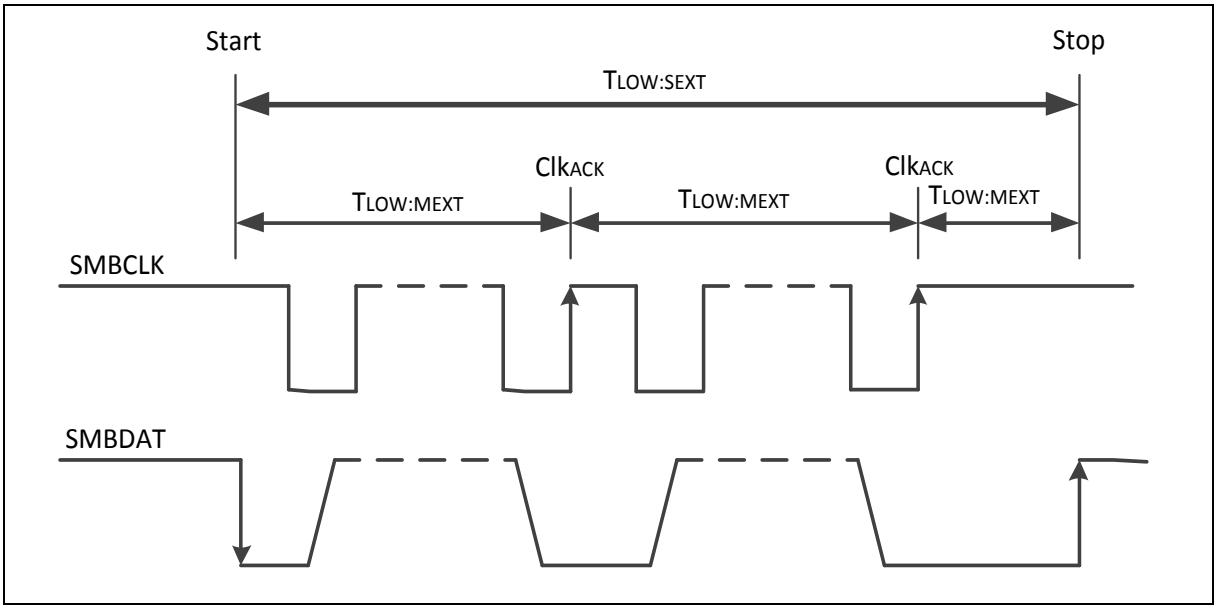


Figure 6.16-20 Bus Clock Low Time Out Timing

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 I^{2}C controller supports PDMA transfer function. When TXPDMAEN (I2C_CTL1 [0]) is set to 1, the I^{2}C controller will issue request to PDMA controller to start the DMA transmission process automatically.

When RXPDMAEN (I2C_CTL1 [1]) is set to 1, the I^{2}C controller will start the receive PDMA process. The I^{2}C controller will issue the request to PDMA controller automatically when there is data written into the received BUFFER.

When I^{2}C 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 I^{2}C controller in master TX mode. If PDMASTR is cleared to 0, I^{2}C will send STOP automatically after PDMA transfer done and buffer empty. If PDMASTR is set to 1, SI will be set to 1 and I^{2}C 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 I^{2}C protocol. Once setup time configuration greater than design limitation, that means if setup time setting make SCL output less than three PCLKs, the I^{2}C controller can't work normally due to SCL must sample three times. And once hold time configuration greater than I^{2}C clock limitation, I^{2}C will occur bus error. It is suggested that user calculate suitable timing with baud rate and protocol before setting timing. Table 6.16-3 shows the relationship between I^{2}C baud rate and PCLK, the number of table represent one clock duty contain how many PCLKs. Setup and hold time configuration even can program some extreme values in the design, but user should follow I^{2}C protocol standard.

I^{2}C 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 6.16-3 Relationship between I^{2}C 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.

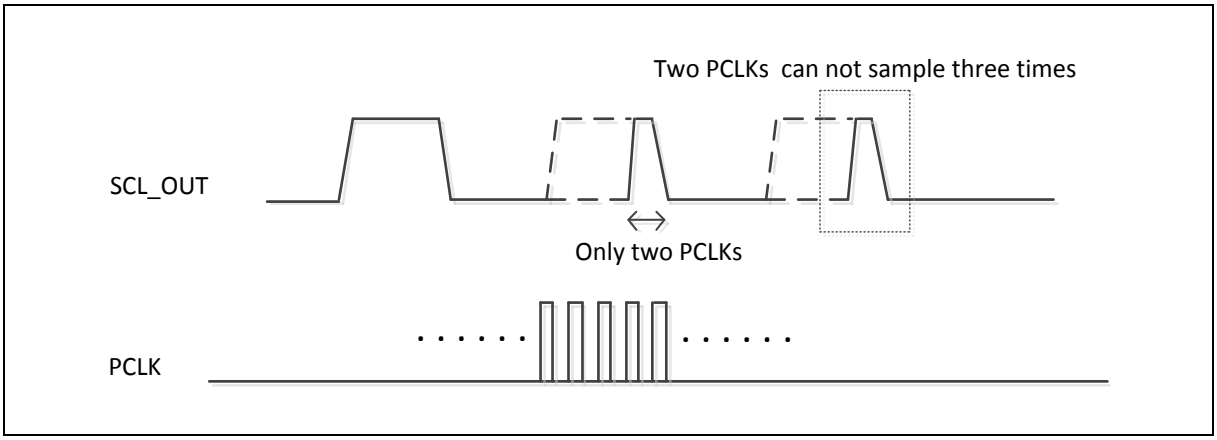


Figure 6.16-21 Setup Time Wrong Adjustment

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.

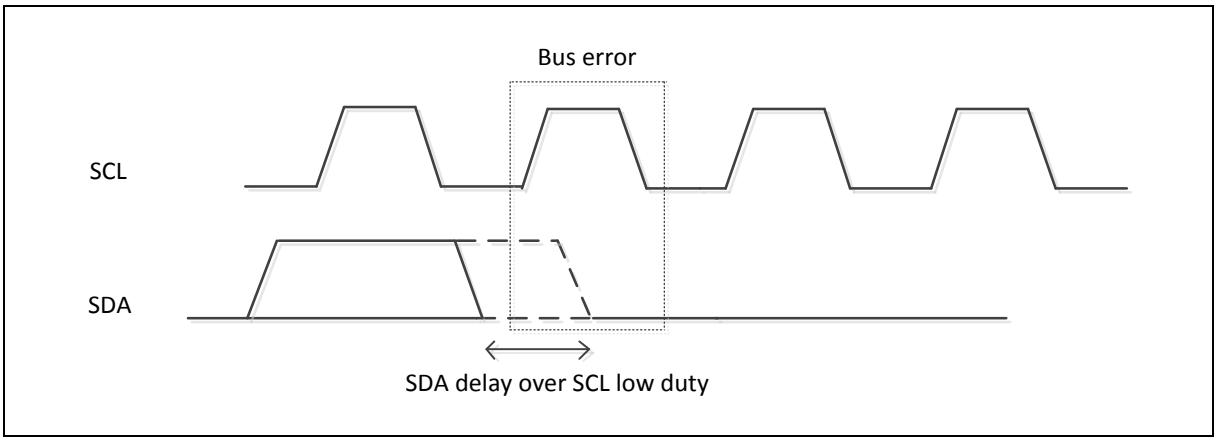


Figure 6.16-22 Hold Time Wrong Adjustment

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_ADDR0 [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 6.16-23 shows I²C Data Shifting Direction.

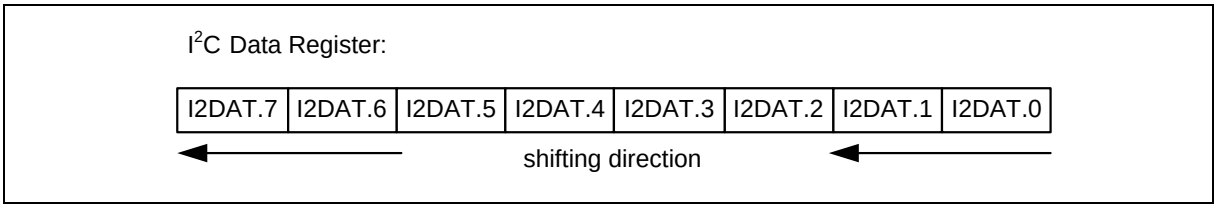


Figure 6.16-23 I²C Data Shifting Direction

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 6.16-4. 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 6.16-4 I^{2}C Status Code Description

Clock Baud Rate Bits (I2C_CLKDIV)

The data baud rate of I^{2}C is determines by DIVIDER(I2C_CLKDIV [7:0]) register when I^{2}C is in Master mode, and it is not necessary in a Slave mode. In the Slave mode, I^{2}C will automatically synchronize it with any clock frequency from master I^{2}C device. In the slave mode, system clock frequency should greater than I^{2}C bus maximum clock 20 times.

The data baud rate of I^{2}C setting is Data Baud Rate of I^{2}C = (system clock) / (4x (I2C_CLKDIV [7:0] +1)). If system clock = 16 MHz, the I2C_CLKDIV [7:0] = 40 (28H), the data baud rate of I^{2}C = 16 MHz/ (4x (40 +1)) = 97.5 Kbits/sec.

Time-out Control Register (I2C_TOCTL)

There is a 14-bit time-out counter which can be used to deal with the I^{2}C bus hang-up. If the time-out counter is enabled, the counter starts up counting until it overflows (TOIF=1) and generates I^{2}C interrupt to CPU or stops counting by clearing TOCEN to 0. When time-out counter is enabled, writing 1 to the SI flag will reset counter and re-start up counting after SI is cleared. If I^{2}C bus hangs up, it causes the I2C_STATUS0 and flag SI are not updated for a period, the 14-bit time-out counter may overflow and acknowledge CPU the I^{2}C interrupt. Refer to Figure 6.16-24 for the 14-bit time-out counter. User may write 1 to clear TOIF to 0.

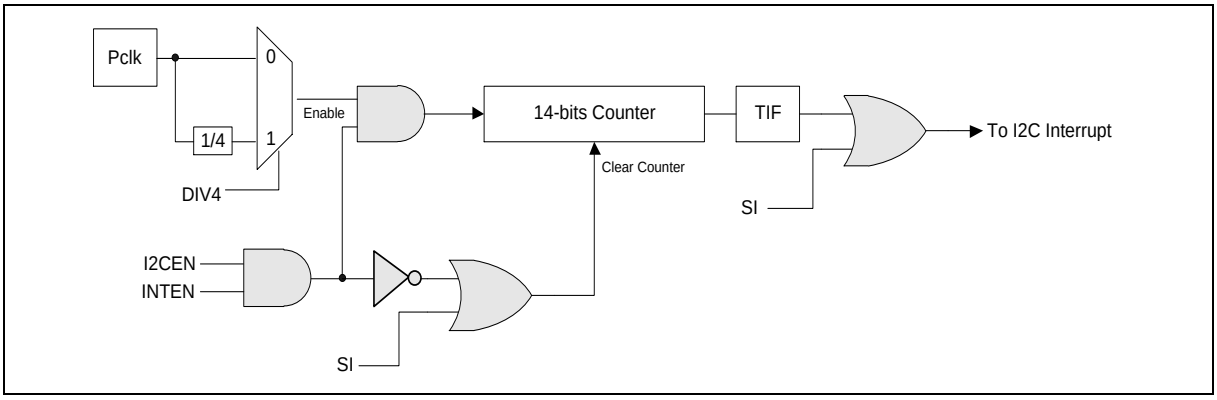


Figure 6.16-24 I²C Time-out Count Block Diagram

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.

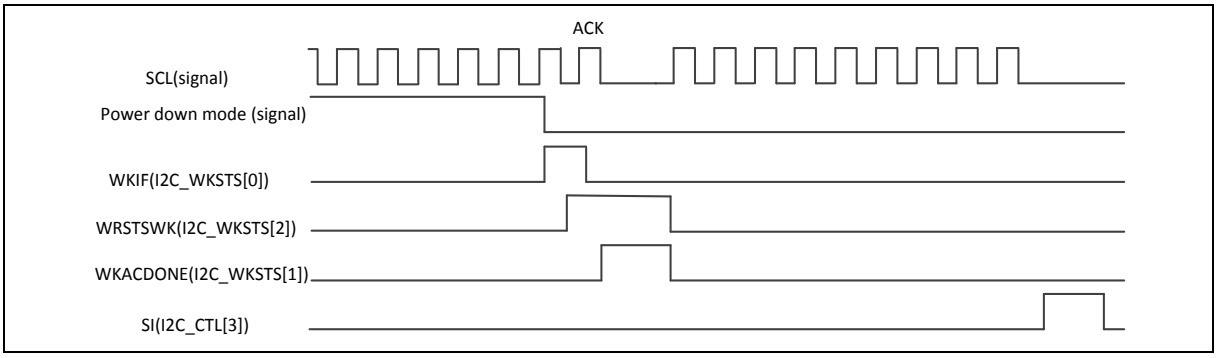


Figure 6.16-25 I²C Wake-Up Related Signals Waveform

I²C Control Register 1 (I2C_CTL1)

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 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 enable control by BUSEN(I2CBUSCTL[7]) bit. The BUSTOUT(I2CBUSCTL[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 the PEC section of SM Bus.

I²C Bus Management Timer and I²C Clock Low Timer Register (I2C_BUSTOUT/ I2C_CLKTOUT)

Both of the definitions of these registers are described in the time-out section of SM Bus.

6.16.5.6 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. The muti-function configuration reference Basic Configuration.
2. Enable I2C0 APB clock. The clock configuration reference Basic Configuration.
3. Set I2C0RST=1 to reset I2C0 controller then set I2C0 controller to normal operation. The reset controller configuration reference Basic 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 6.16-26 shows the EEPROM random read operation.

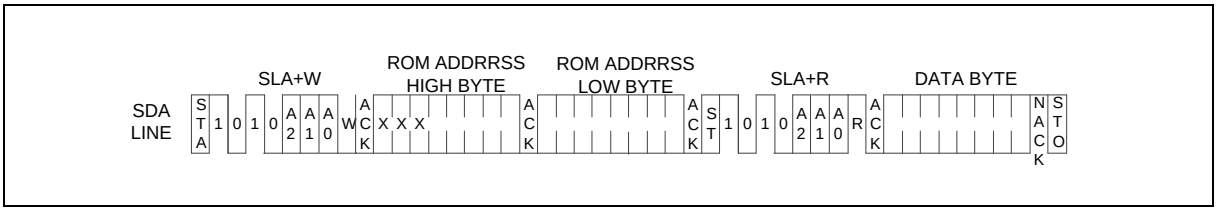


Figure 6.16-26 EEPROM Random Read

Figure 6.16-27 shows how to use the I²C controller to implement the protocol of EEPROM random read.

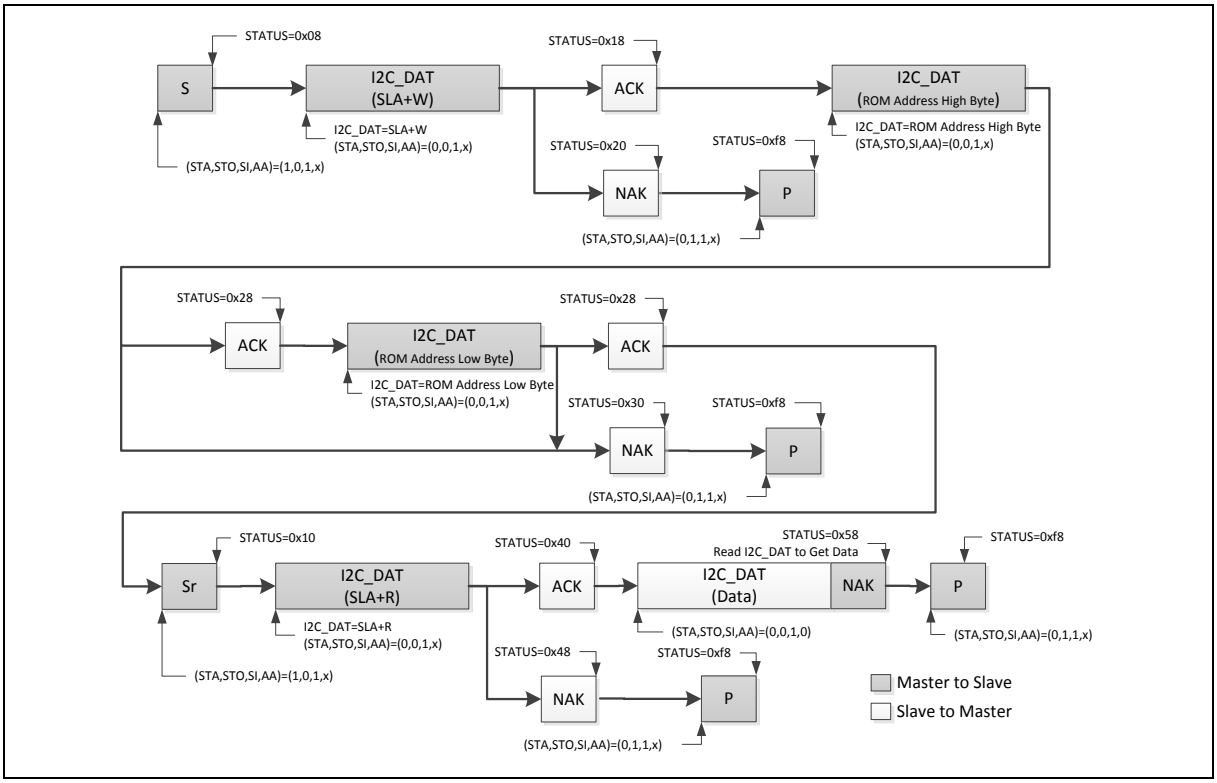


Figure 6.16-27 Protocol of EEPROM Random Read

The I^2C controller, which is a master, sends START to bus. Then, it sends a SLA+W (Slave address + Write bit) to EERPOM followed by two bytes data address to set the EEPROM address to read. Finally, a Repeat START followed by SLA+R is sent to read the data from EEPROM.

6.16.6 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	0x0000_0000
I2C_BUSCTL	I2Cn_BA+0x50	R/W	I²C Bus Management Control Register	0x0000_0000
I2C_BUSTCTL	I2Cn_BA+0x54	R/W	I²C Bus Management Timer Control Register	0x0000_0000
I2C_BUSSTS	I2Cn_BA+0x58	R/W	I²C Bus Management Status Register	0x0000_0000
I2C_PKTSIZE	I2Cn_BA+0x5C	R/W	I²C Packet Error Checking Byte Number Register	0x0000_0000
I2C_PKTCRC	I2Cn_BA+0x60	R	I²C Packet Error Checking Byte Value Register	0x0000_0000
I2C_BUSTOUT	I2Cn_BA+0x64	R/W	I²C Bus Management Timer Register	0x0000_0005
I2C_CLKTOUT	I2Cn_BA+0x68	R/W	I²C Bus Management Clock Low Timer Register	0x0000_0005

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**
Reserved						**DIVIDER**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DIVIDER							

Bits	Description	
[31:10]	**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 (ADDR0)

Register	Offset	R/W	Description	Reset Value
I2C_ADDR0	I2Cn_BA+0x04	R/W	I²C Slave Address Register0	0x0000_0000

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

Bits	Description	
[31:11]	**Reserved**	Reserved.
[7:1]	**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 7'h00, the address can not be used.
[0]	**GC**	**General Call Function** 0 = General Call Function Disabled. 1 = General Call Function Enabled.

I²C Slave Address Register (ADDRx)

Register	Offset	R/W	Description	Reset Value
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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ADDR							**Reserved**

Bits	Description	
[31:11]	**Reserved**	Reserved.
[7:1]	**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 7'h00, the address can not be used.
[0]	**Reserved**	Reserved.

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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ADDRMSK							**Reserved**

Bits	Description	
[31:11]	**Reserved**	Reserved.
[7:1]	**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.
[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 (Read Only)** 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 (I2C_WKSTS[1]) 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							PDMASTR
7	6	5	4	3	2	1	0
Reserved					PDMARST	RXPDMAEN	TXPDMAEN

Bits	Description	
[31:7]	**Reserved**	Reserved.
[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							

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.
[7:0]	**Reserved**	Reserved.

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	0x0000_0000

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

Bits	Description	
[31:25]	**Reserved**	Reserved.
[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 (Read Only)** 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 SMBALERT 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 (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 = 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 clear to 0 first in 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 clear to 0 first in the BUSEN is set.

6.17 USCI - Universal Serial Control Interface Controller (USCI)

6.17.1 Overview

The Universal Serial Control Interface (USCI) is a flexible interface module covering several serial communication protocols. The user can configure this controller as UART, SPI, or I^2C functional protocol.

6.17.2 Features

The controller can be individually configured to match the application needs. The following protocols are supported:

- UART
- SPI
- I^2C

6.17.3 Block Diagram

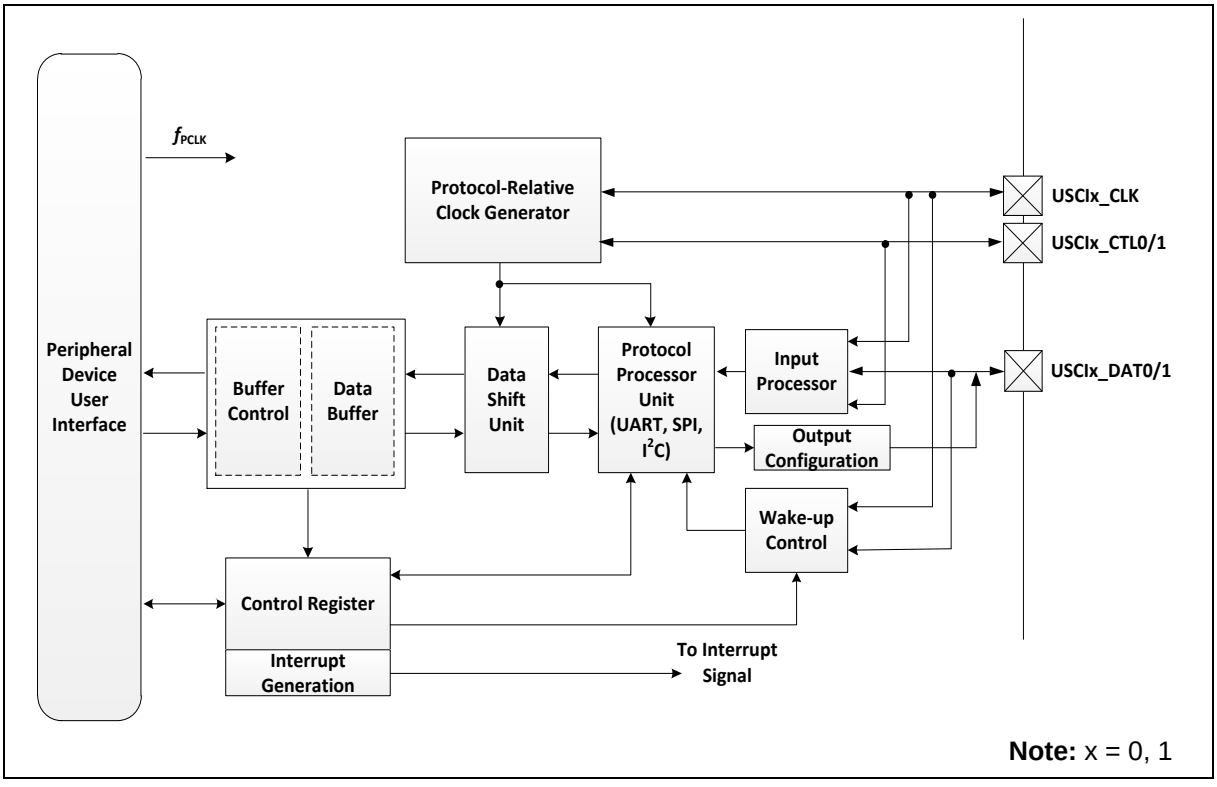


Figure 6.17-1 USCI Block Diagram

6.17.4 Functional Description

The structure of the Universal Serial Control Interface (USCI) controller is shown in Figure 6.17-1 USCI Block Diagram. The input signal is implemented in input processor. The data buffers and the data shift unit support the data transfers. Each protocol-specific function is handled by the protocol processor unit. The timing and time event control signals of the specific protocol are handled by the protocol-relative clock generator. All the protocol-specific events are processed in the interrupt generation unit. The wake-up function of the specific protocol is implemented in the wake-up control unit.

The USCI is equipped with three protocols including UART, SPI, and I^2C. They can be selected by

FUNMODE (USCI_CTL [2:0]). Note that the FUNMODE must be set to 0 before changing protocol.

6.17.4.1 I/O Processer

Input Signal

All input stages offer the similar feature set. They are used for all protocols.

Table 6.17-1lists the relative input signals for each selected protocol. Each input signal is handled by an input processor for signal conditioning, such as signal inverse selection control, or a digital input filter.

Selected Protocol		UART	SPI	I^2C
Serial Bus Clock Input	USCIx_CLK	-	SPI_CLK	SCL
Control Input	USCIx_CTL0	nCTS	SPI_SS	-
	USCIx_CTL1	-	-	-
Data Input	USCIx_DAT0	RX	SPI_MOSI_0	SDA
	USCIx_DAT1	-	SPI_MISO_0	-

Table 6.17-1 Input Signals for Different Protocols

The description of protocol-specific items are given in the related protocol chapters.

General Input Structure

The input structures of data and control signals include inverter, digital filter and edge detection (data signal only).

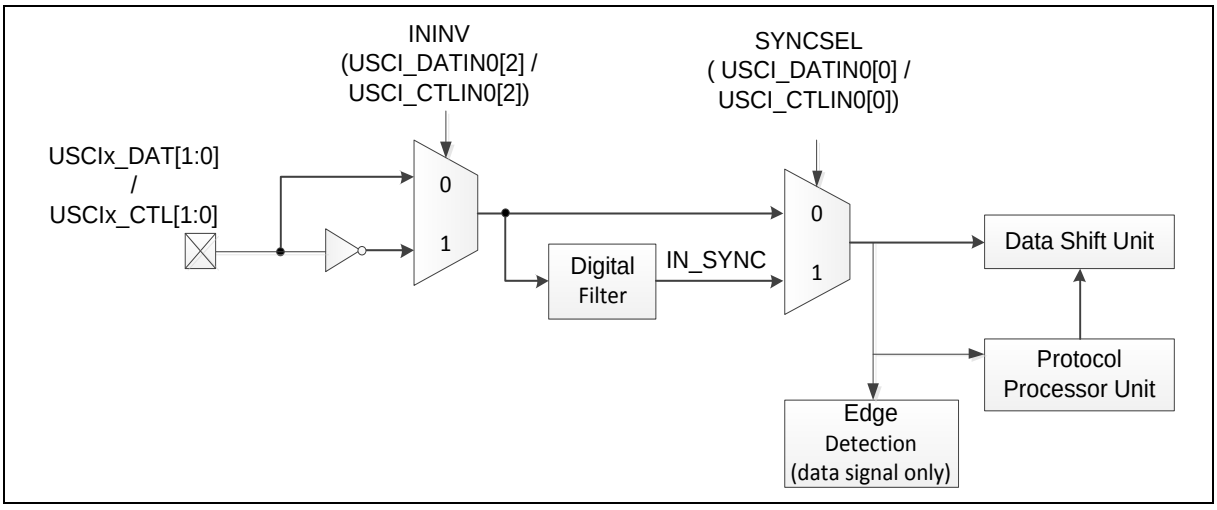


Figure 6.17-2 Input Conditioning for USCIx_DAT[1:0] and USCIx_CTL[1:0]

The input structure of USCIx_CLK is similar to USCIx_CTL[1:0] input structure, except it does not support inverse function.

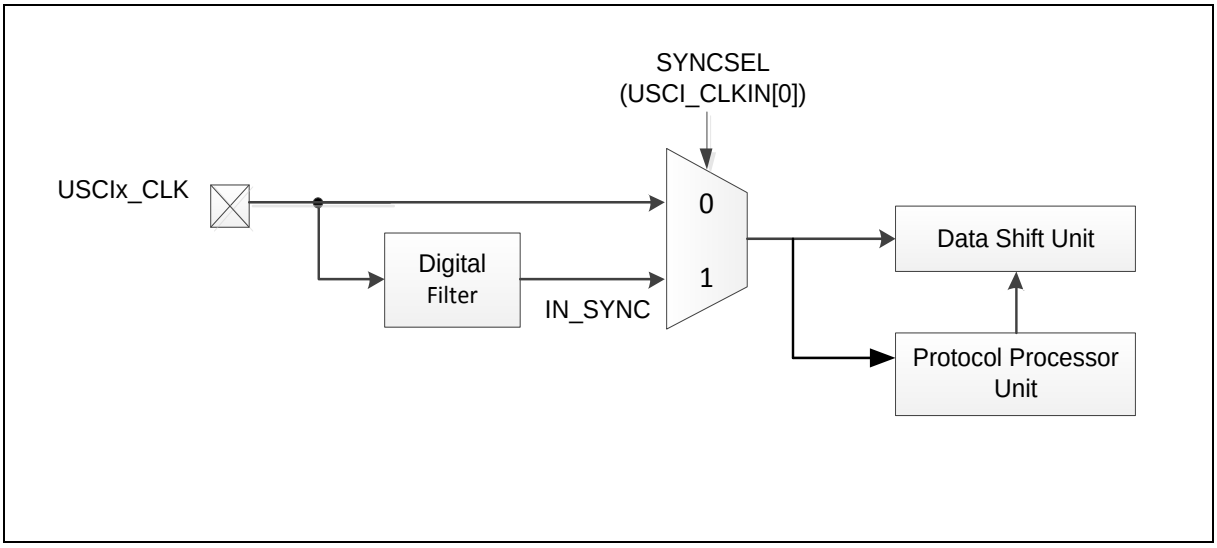


Figure 6.17-3 Input Conditioning for USCIx_CLK

All configurations of control, clock and data input structures are in USCI_CTLIN0, USCI_CLKIN and USCI_DATIN0 registers respectively. EDGEDET (USCI_DATIN0[4:3]) is used to select the edge detection condition. Note that the EDGEDET for USCI_DATIN0 must be set 2'b10 in UART mode. The programmable edge detection indicates that the desired event has occurred by activating the trigger signal.

ININV (USCI_DATIN0[2] / USCI_CTLIN0[2]) allows a polarity inversion of the selected input signal to adapt the input signal polarity to the internal polarity of the data shift unit and the protocol state machine.

If the SYNCSEL (USCI_DATIN0[0] / USCI_CTLIN0[0] / USCI_CLKIN[0]) is set to 0, the paths of input signals do not contain any delay due to synchronization or filtering. If there is noise on the input signals, there is the possibility to synchronize the input signal (signal IN_SYNC is synchronized to f_{PCLK}). The synchronized input signal is taken into account by SYNCSEL = 1. The synchronization leads to a delay in the signal path of 2-3 times the period of f_{PCLK}.

Output Signals

Table 6.17-2 shows the relative output signals for each protocol. The number of actually used outputs depends on the selected protocol and they can be classified according to their meaning for the protocols.

Selected Protocol		UART	SPI	I^2C
Serial Bus Clock Output	USCIx_CLK	-	SPI_CLK	SCL
Control Output	USCIx_CTL0	-	SPI_SS	-
	USCIx_CTL1	nRTS	-	-
Data Output	USCIx_DAT0	-	SPI_MOSI_0	SDA
	USCIx_DAT1	TX	SPI_MISO_0	-

Table 6.17-2 Output Signals for Different Protocols

The description of protocol-specific items are given in the related protocol chapters.

6.17.4.2 Data Buffering

The data handling of the USCI controller is based on a Data Shift Unit (DSU) and a buffer structure. Both of the data shift and buffer registers are 16-bit wide. The inputs of Data Shift Unit include the shift data, the serial bus clock, and the shift control. The output pin of transmission can be USCIx_DAT0 pin or USCIx_DAT1 pin depends on what protocol is selected.

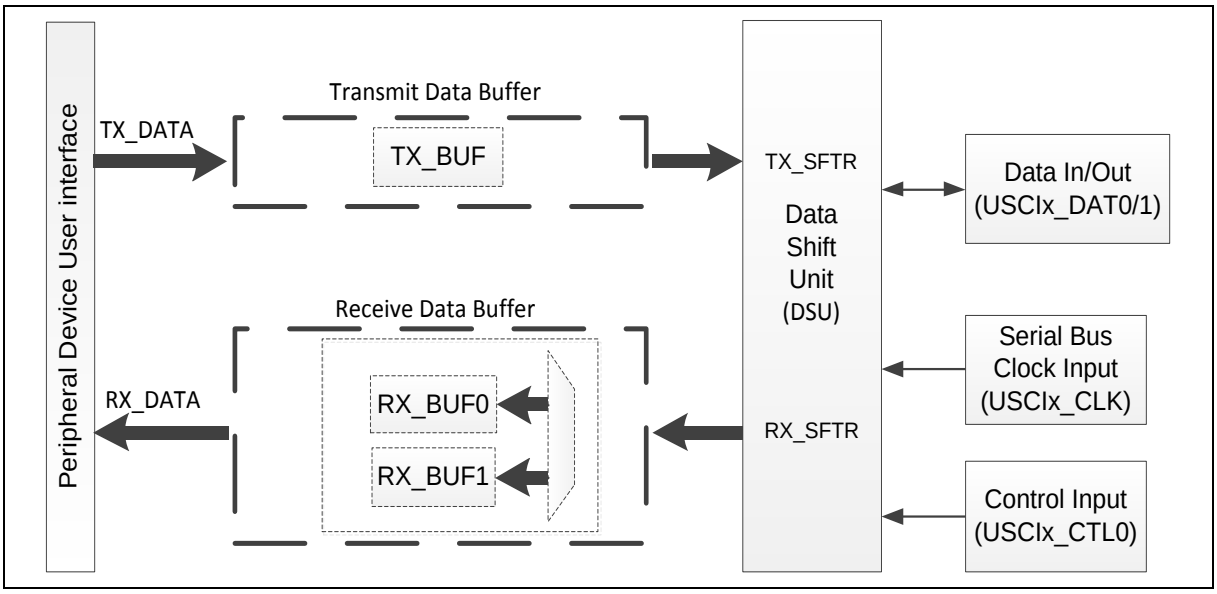


Figure 6.17-4 Block Diagram of Data Buffering

The operation of data handling includes:

- The peripheral device user interface (APB) is used to handle data, interrupts, status and control information.
- A transmitter includes transmit shift register (TX_SFTR) and a transmit data buffer (TX_BUF). The TXFULL / TXEMPTY (USCI_BUFSTS[9:8]) and TXENDIF (USCI_PROTSTS[2]) can indicate the status of transmitter.
- A receiver includes receive shift register (RX_SFTR) and a double receive buffer structure (RX_BUF0, RX_BUF1). In double buffer structure, user need not care about the reception sequence and two received data can be hold if user does not read the data of USCI_RXDAT register in time.

Data Access Structure

The Data Access Structure includes read access to received data and write access of data to be transmitted. The received data is stored in the receiver buffers including RX_BUF0 and RX_BUF1. User need not care about the reception sequence. The receive buffer can be accessed by reading USCI_RXDAT register. The first received data is read out first and the next received data becomes visible in USCI_RXDAT and can be read out next.

Transmit data can be loaded to TX_BUF by writing to the transmit register USCI_TXDAT.

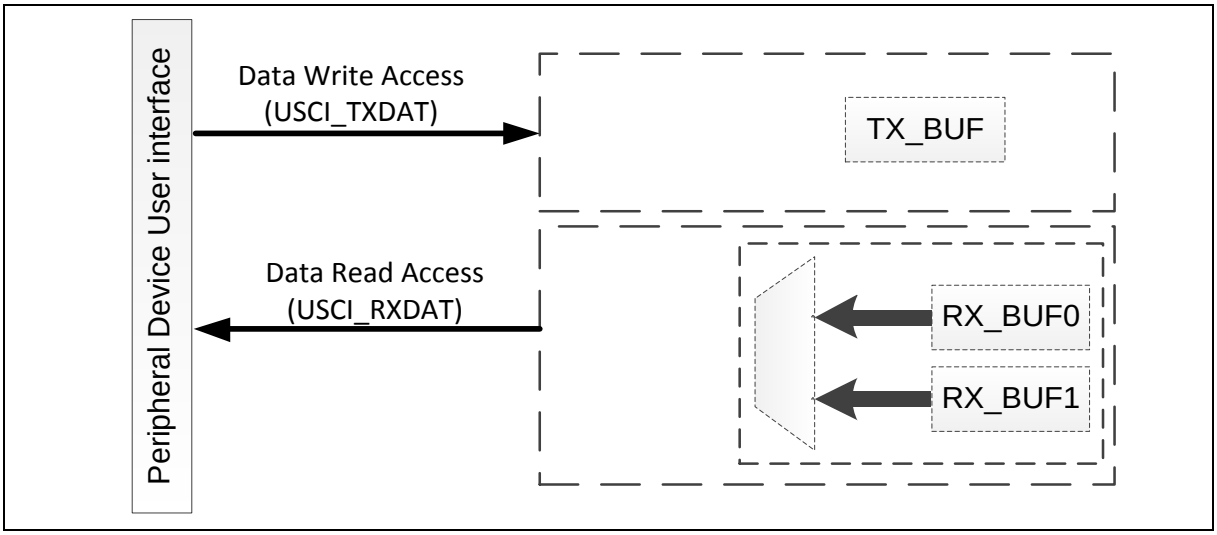


Figure 6.17-5 Data Access Structure

Transmit Data Path

The transmit data path is based on 16-bit wide transmit shift register (TX_SFTR) and transmit buffer TX_BUF. The data transfer parameters like data word length is controlled commonly for transmission and reception by the line control register USCI_LINECTL.

Transmit Buffering

The transmit shift register cannot be directly accessed by user. It is updated automatically with the value stored in the transmit buffer (TX_BUF) if a currently transmitted data is finished and new data is valid for transmission.

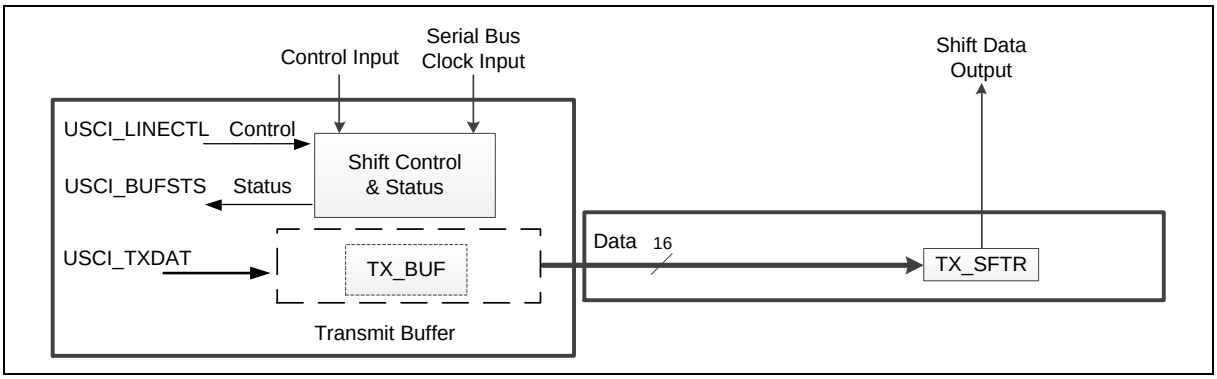


Figure 6.17-6 Transmit Data Path

Transmit Data Validation

The status of TXEMPTY (USCI_BUFSTS[8]) indicates the transmission data is valid or not in the transmit buffer (TX_BUF) and the TXSTIF (USCI_PROTSTS[1]) labels the start conditions for each data.

- If the USCI controller is a Master, the data transfer can only be started with valid data in the transmit buffer (TX_BUF). In this case, the transmit shift register is loaded with the content of transmit buffer.

  Note: Master defines the start of data transfer.

- If the USCI controller is a Slave, a data transfer requested by Master and it has to be started independently of the status in transmit buffer (TX_BUF). If a data transfer is requested and started by the Master, the transmit shift register is loaded from specific protocol control signal if it is valid for transmission.

Note: Slave can not define the start itself, but has to react.

- The timing of loading data from transmit buffer to data shift unit depends on protocol configurations.
- **UART:** A transmission of the data word in transmit buffer can be started if TXEMPTY = 0 in normal operation. In auto flow control, A transmission of the data word in transmit buffer can be started while TXEMPTY = 0 and USCIx_CTL0 in active stage.
- **SPI:** In Master mode, data transmission will be started when TXEMPTY (USCI_BUFSTS[8]) is 0. In Slave mode, the data transmission can be started only when slave selection signal is at active state and clock is presented on USCIx_CLK pin.
- **I^2C:** A transmission of the data byte in transmit buffer can be started if TXEMPTY = 0.
- A transmission data which is located in transmit buffer can be started if the TXEMPTY (USCI_BUFSTS [8]) = 0. The content of the transmit buffer (in TX_BUF condition) should not be overwritten with new data while it is valid for transmission and a new transmission can start. If the content of TX_BUF has to be changed, user can set TXRST (USCI_BUFCTL [16]) to 1 to clear the content of TX_BUF before updating the data. Moreover, TXEMPTY (USCI_BUFSTS [8]) will be cleared automatically when transmit buffer (TX_BUF) is updated with new data. While a transmission is in progress, TX_BUF can be loaded with new data. User has to update the TX_BUF before a new transmission.

Receive Data Path

The receive data path is based on 16-bit wide receive shift register RX_SFTR and receive buffers RX_BUF0 and RX_BUF1. The data transfer parameters like data word length, or the shift direction are controlled commonly for transmission and reception by the line control register USCI_LINECTL. Register USCI_BUFSTS monitors the data validation of USCI_RXDAT.

Receive Buffering

The receive shift register cannot be directly accessed by user, but its content is automatically loaded into the receive buffer if a complete data word has been received or the frame is finished. The received data words in Receive Buffer can be read out automatically from register USCI_RXDAT.

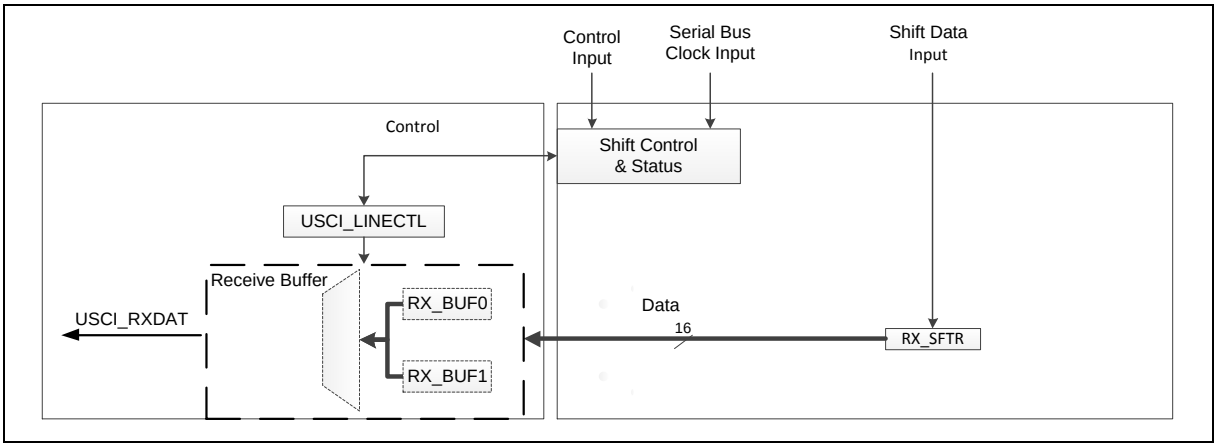


Figure 6.17-7 Receive Data Path

6.17.4.3 Port Direction Control

In SPI protocol with half-duplex configurations, the data port is bidirectional. Port direction control is intended to control the pin direction through a dedicated hardware interface.

The direction of selected pin is controlled by PORTDIR (USCI_TXDAT[16]). When user writes USCI_TXDAT register, the transmit data and its port direction are settled simultaneously.

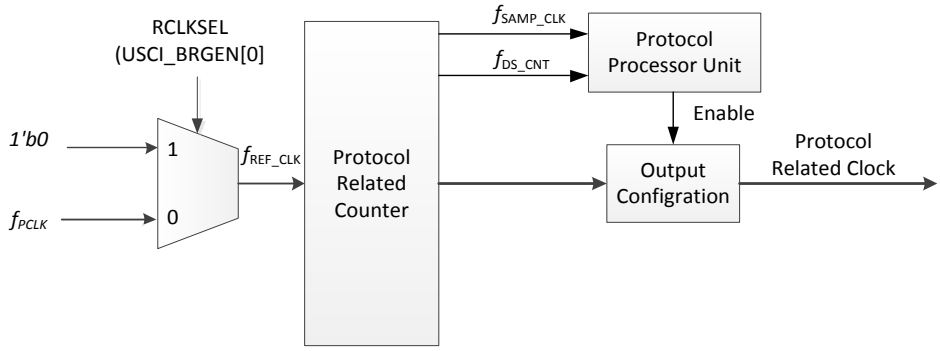
6.17.4.4 Protocol Control and Status

The protocol-related control and status information are located in the protocol control register

USCI_PROTCTL and in the protocol status register USCI_PROTSTS. These registers are shared between the available protocols. As a consequence, the meaning of the bit positions in these registers is different within the protocols. Refer to each protocol's relative register for detail information.

6.17.4.5 Protocol-Relative Clock Generator

The USCI controller contains a protocol-relative clock generator and it is controlled by register USCI_BRGEN. It is reset when the USCI_BRGEN register is written. The structured of protocol-relative clock generator is shown below.



Note: Refer the **Basic Clock Divider Counter** section to get the f_{SAMP_CLK}

Figure 6.17-8 Protocol-Relative Clock Generator

The protocol related counter contains basic clock divider counter and timing measurement counter. It is based on a divider stages, providing the frequencies needed for the different protocols. It contains:

- The basic clock divider counter provides the protocol relative clock signal and other protocol-related signals (fSAMP_CLK and fDS_CLK).
- The timing measurement counter for time interval measurement, e.g. baud rate detection on UART protocol.
- The output signals of protocol relative clock generator can be made available on pins (e.g USCIx_CLK for SPI).

Basic Clock Divider Counter

The basic clock divider counter is used for an integer division delivering f_{REF_CLK2}, f_{REF_CLK}, f_{DIV_CLK}, f_{SCLK} ,and f_{SAMP_CLK} . The frequencies of this divider are controlled by PTCLKSEL (USCI_BRGEN [1]), CLKDIV (USCI_BRGEN [25:16]), SPCLKSEL (USCI_BRGEN [3:2]).

The basic clock divider counter is used to generate the relative protocol timing signals.

$$f_{\mathrm{DIV\_CLK}} = f_{REF\_CLK} \times \frac{1}{\mathrm{CLKDIV} + 1} \text{ if } \mathrm{PTCLKSEL} = 0$$

$$f_{\mathrm{DIV\_CLK}} = f_{REF\_CLK} \times \frac{1}{(\mathrm{CLKDIV} + 1) \times 2} \text{ if } \mathrm{PTCLKSEL} = 1$$

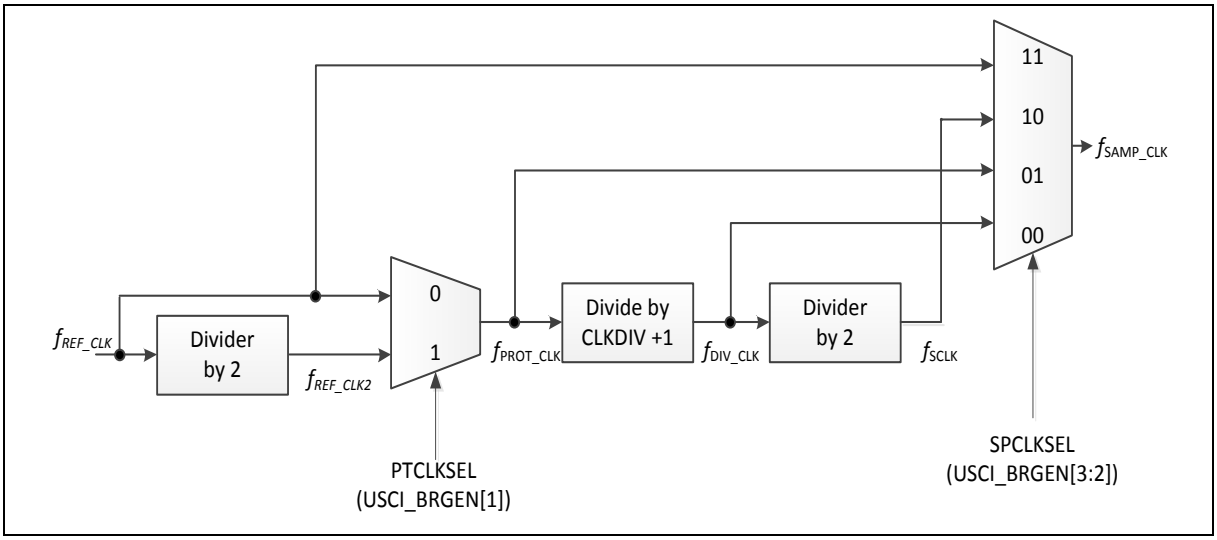


Figure 6.17-9 Basic Clock Divider Counter

Timing Measurement Counter

The timing measurement counter is used for time interval measurement and is enabled by TMCNTEN (USCI_BRGEN [4]) = 1. When TMCNTSRC (USCI_BRGEN [5]) is set to 1, the timer works on f_{DIV_CLK}, otherwise, the timer works independently from f_{PROT_CLK}. Therefore, any serial data reception or transmission can continue while the timer is performing timing measurements. The timer counts the length of protocol-related signals with f_{PROT_CLK} or f_{DIV_CLK}. It stops counting when it reaches the user-specified value.

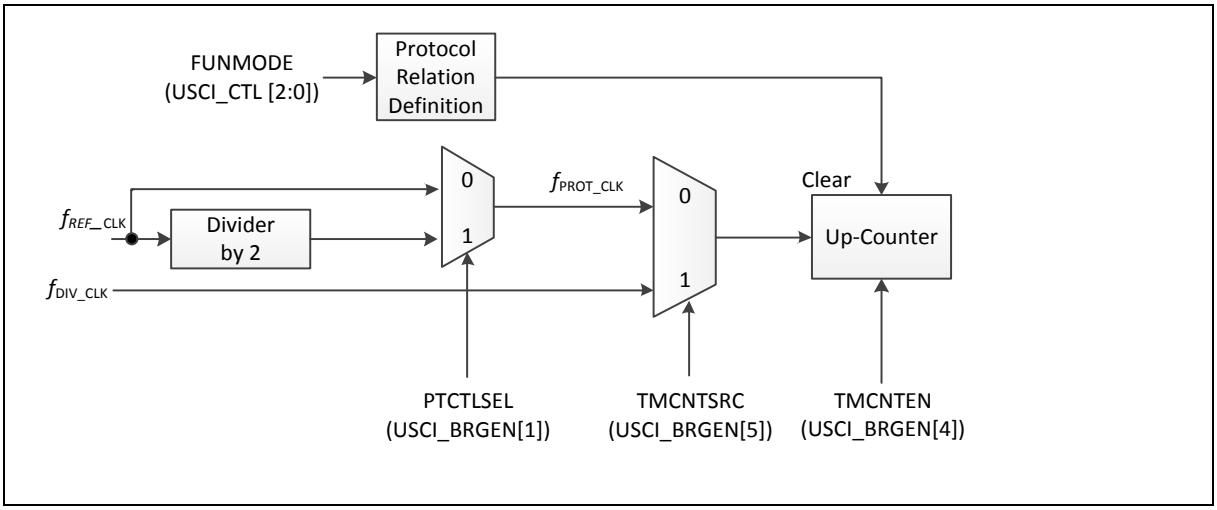


Figure 6.17-10 Block of Timing Measurement Counter

The timing measurement counter is used to perform time-out function or auto-baud rate mechanism. Its functionality depends on the selected protocol as shown below.

- UART: The timing measurement counter is used in auto baud rate detection.
- SPI: The timing measurement counter is used for counting the slave time-out period.
- I^2C: The timing measurement counter indicates time-out clock cycle.

Sample Time Counter

A sample time counter associated to the protocol related counter defining protocol specific timings, such shift control signals or bit timings, based on the input frequency f_{SAMP_CLK}. The sample time

counter allows generating time intervals for protocol-specific purposes. The period of a sample frequency f_{PDS_CNT} is given by the selected input frequency f_{SAMP_CLK} and the programmed pre-divider value (PDSCNT (USCI_BRGEN [9:8])). The meaning of the sample time depends on the selected protocol. Please refer to the corresponding chapters for more protocol-specific information.

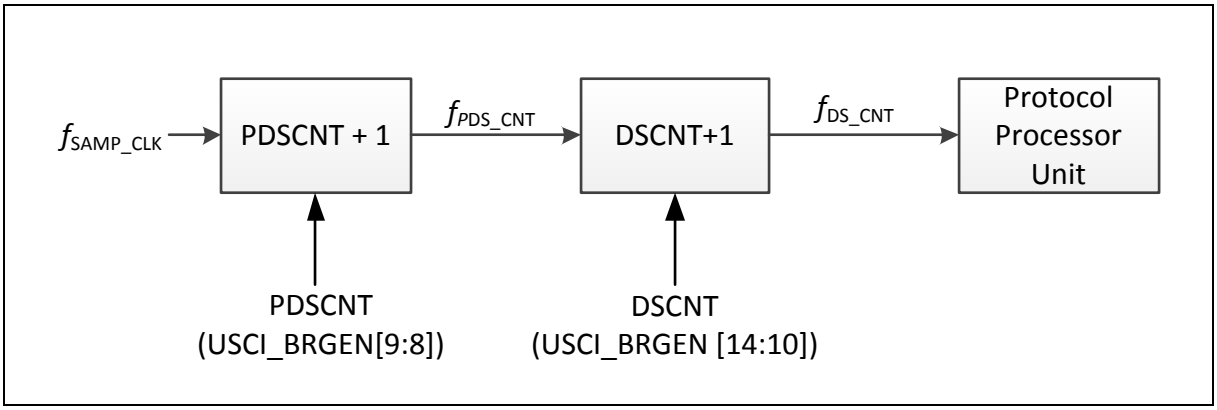


Figure 6.17-11 Sample Time Counter

6.17.4.6 Data Transfer Events and Interrupts

The data transfer events are based on the transmission or reception of a data word. The related indication flags are located in register USCI_PROTSTS. All events can be individually enabled for interrupt generation. If the FUNMODE (USCI_CTL [2:0]) is set to 0, the USCI is disabled. When FUNMODE (USCI_CTL [2:0]) is setting for a protocol port, the internal states will be controlled by logic hardware of the selected protocol.

- Transmit start interrupt event to indicate that a data word has been started:

  A transmit start interrupt event occurs when the data is loaded into transmitted shift register. It is indicated by flag TXSTIF (USCI_PROTSTS [1]) and, if enabled, leads to transmit start interrupt.

- Transmit end interrupt event to indicate that a data word transmission has been done:

  A transmit end interrupt event occurs when the current transmit data in shift register had been finished. It is indicated by flag TXENDIF (USCI_PROTSTS [2]) and, if enabled, leads to transmit end interrupt. This event also indicates when the shift control settings (word length, shift direction, etc.) are internally “frozen” for the current data word transmission. In UART and I^2C mode, the transmit data valid is according to TXEMPTY (USCI_BUFSTS [8]) and protocol relative internal signal with the transmit end interrupt event.

- Receiver start event to indicate that a data word reception has started:

  When the receive clock edge that shifts in the first bit of a new data word is detected and reception is enabled, a receiver start event occurs. It is indicated by flag RXSTIF (USCI_PROTSTS [3]) and, if enabled, leads to receiver start interrupt.

- Receive event to indicate that a data word has been received:

  If a new received word becomes available in the receive buffer, a receive event occurs. It is indicated by flag RXENDIF (USCI_PROTSTS [4]) and, if enabled, leads to receive interrupt.

- Data lost event to indicate a loss of the newest received data word:

  If the data word available in register USCI_RXDAT (oldest data word from RX_BUF0 or RX_BUF1) has not been read out and the receive buffer is FULL, the new incoming data will lose and this event occurs. It is indicated by flag RXOVIF (USCI_BUFSTS[3]) and, if enabled, leads to a protocol interrupt.

The general event and interrupt structure is shown in Figure 6.17-12.

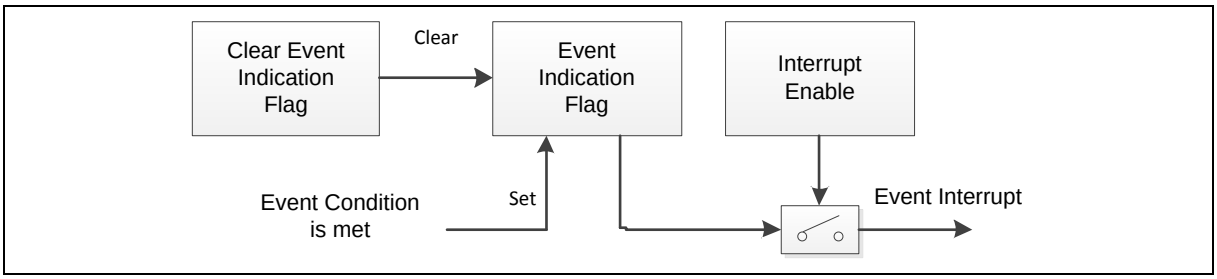


Figure 6.17-12 Event and Interrupt Structure

Each general interrupt enable can set by RXENDIEN, RXSTIEN, TXENDIEN, and TXSTIEN of USCI_INTEN [4:1]. The events are including receive end interrupt event, receive start interrupt event, transmit end interrupt event, and transmit start interrupt event. For protocol-specific interrupt, it is specified in each protocol interrupt enable register.

If a defined condition is met, an event is detected and an event indication flag becomes automatically set. The flag stays set until it is cleared by software. If enabled, an interrupt can be generated if an event is detected.

The registers, bits and bit fields indicate the data transfer events and control the general interrupts of a USCI are shown in Table 6.17-3.

Event	Indication Flag	Indication Cleared By	Interrupt Enabled By
Transmit start interrupt event	TXSTIF (USCI_PROTSTS [1])	It is cleared by software writes 1 to corresponding interrupt bit of USCI_PROTSTS.	TXSTIEN (USCI_INTEN [1])
Transmit end interrupt event	TXENDIF (USCI_PROTSTS [2])		TXENDIEN (USCI_INTEN [2])
Receive start interrupt event	RXSTIF (USCI_PROTSTS [3])		RXSTIEN (USCI_INTEN [3])
Receive end interrupt event	RXENDIF (USCI_PROTSTS [4])		RXENDIEN (USCI_INTEN [4])

Table 6.17-3 Data Transfer Events and Interrupt Handling

6.17.4.7 Protocol-specific Events and Interrupts

These events are related to protocol-specific actions that are described in the corresponding protocol chapters. The related indication flags are located in register USCI_PROTSTS. All events can be individually enabled for the generation of the common protocol interrupt.

Event	Indication Flag	Indication Cleared By	Interrupt Enabled By
Protocol-specific events in UART mode	USCI_PROTSTS [17:16] and USCI_PROTSTS [11:5]	It is cleared by software writes 1 to corresponding interrupt bit of USCI_PROTSTS.	USCI_PROTIEN[2:1]
Protocol-specific events in SPI mode	USCI_PROTSTS [9:8], USCI_PROTSTS [6:5]		USCI_PROTIEN [3:0]
Protocol-specific events in I^2C mode	USCI_PROTSTS [13:8], USCI_PROTSTS [5]		USCI_PROTIEN [6:0]

Table 6.17-4 Protocol-specific Events and Interrupt Handling

6.17.4.8 Wake-up

The protocol-related wake-up functional information is located in the Wake-up Control Register (USCI_WKCTL) and in the Wake-up Status Register (USCI_WKSTS). These registers are shared between the available protocols. As a consequence, the meaning of the bit positions in these registers is different within the protocols.

6.17.4.9 PDMA

The USCI supports PDMA transfer function. When PDMAEN (USCI_PDMACTL [3]) is set to 1, the PDMA function is enabled.

When TXPDMAEN (USCI_PDMACTL [1]) is set to 1, the controller will issue request to PDMA controller to start the PDMA transmission process automatically.

When RXPDMAEN (USCI_PDMACTL [2]) is set to 1, the controller will start the PDMA reception process. USCI will issue request to PDMA controller automatically when there is data in the receive FIFO buffer.

In UART function, the requirement of RXPDMAEN will be cleared and hold if there is any error condition events including frame error, parity error or break detection. The user shall read out the current data and then the requirement of RXPDMAEN will send to the PDMA module in the next data.

6.18 USCI – UART Mode

6.18.1 Overview

The asynchronous serial channel UART covers the reception and the transmission of asynchronous data frames. It performs a serial-to-parallel conversion on data received from the peripheral, and a parallel-to-serial conversion on data transmitted from the controller. The receiver and transmitter being independent, frames can start at different points in time for transmission and reception.

The UART controller also provides auto flow control. There are two conditions to wake-up the system.

6.18.2 Features

- Supports one transmit buffer and two receive buffer for data payload
- Supports hardware auto flow control function
- Supports programmable baud-rate generator
- Support 9-bit Data Transfer (Support 9-bit RS-485)
- Baud rate detection possible by built-in capture event of baud rate generator
- Supports PDMA capability
- Supports Wake-up function (Data and nCTS Wakeup Only)

6.18.3 Block Diagram

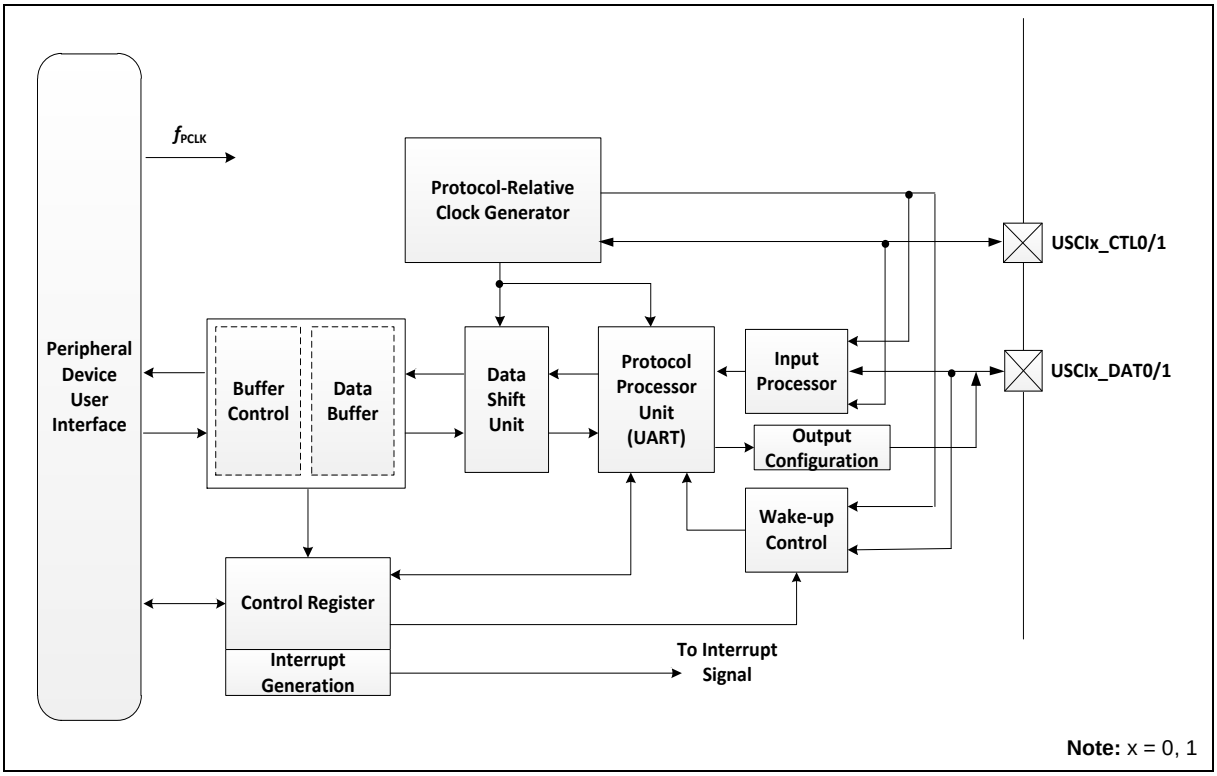


Figure 6.18-1 USCI-UART Mode Block Diagram

6.18.4 Basic Configuration

6.18.4.1 USCI0 UART Basic Configurations

The basic configurations of USCI0_UART are as follows:

- Clock Source Configuration
  - Enable USCI0 clock (USCI0CKEN) on CLK_APBCLK1[8] register.
  -
  - Enable USCI0_UART function UUART_CTL[2:0] register, UUART_CTL[2:0]=3'b010.
- Reset Configuration
  - Reset USCI0 controller in USCI0RST (SYS_IPRST2[8]).

6.18.4.2 USCI1 UART Basic Configurations

The basic configurations of USCI1_UART are as follows:

- Clock Source Configuration
  - Enable USCI1 clock (USCI1CKEN) on CLK_APBCLK1[9] register.
  - Enable USCI1_UART function UUART_CTL[2:0] register, UUART_CTL[2:0]=3'b010.
- Reset Configuration
  - Reset USCI1 controller in USCI1RST (SYS_IPRST2[9]).

6.18.5 Functional Description

6.18.5.1 USCI Common Function Description

Please refer to section **6.17.4** for detailed information.

6.18.5.2 Signal Description

An UART connection is characterized by the use of a single connection line between a transmitter and a receiver. The receiver input signal (RXD) is handled by the input stage USCIx_DAT0 and the transmit output (TXD) signal is handled by the output stage of USCIx_DAT1.

For full-duplex communication, an independent communication line is needed for each transfer direction. Figure 6.18-2 shows an example with a point-to-point full-duplex connection between two communication partners UART module A and UART module B.

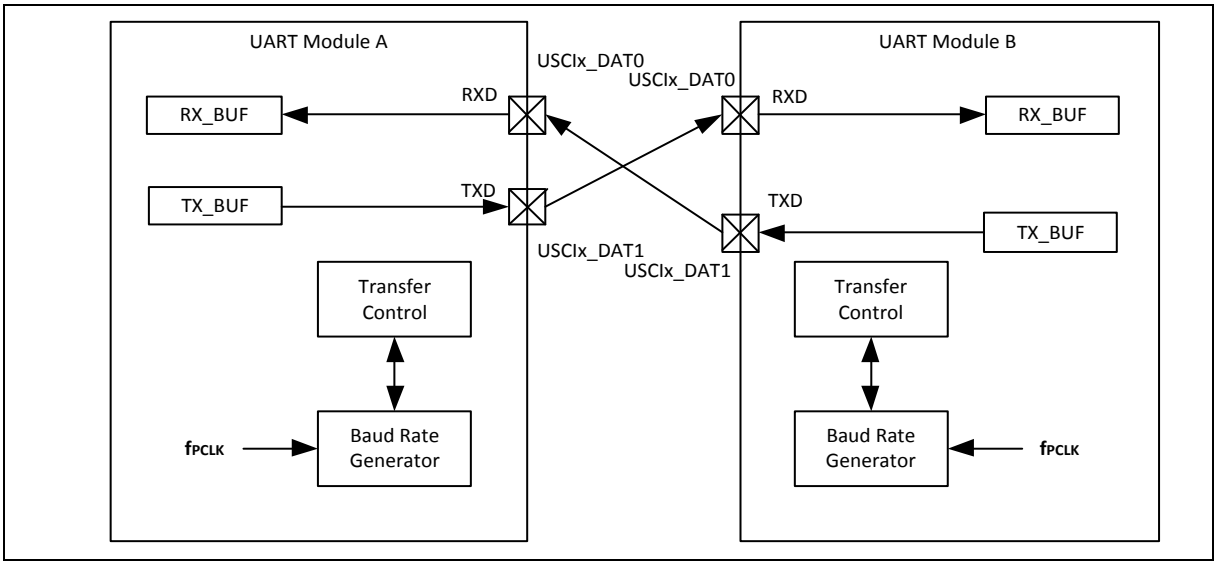


Figure 6.18-2 UART Signal Connection for Full-Duplex Communication

Input Signal

For UART protocol, the number of input signals is shown in Table 6.18-1. Each input signal is handled by an input processor for signal conditioning, such as signal inverse selection control, or a digital input filter. They can be classified according to their meaning for the protocols (see Table 6.18-1).

Selected Protocol		UART
Control Input	USCIx_CTL0	nCTS
	USCIx_CTL1	X
Data Input	USCIx_DAT0	RX
	USCIx_DAT1	X

Table 6.18-1 Input Signals for UART Protocol

Output Signals

For UART protocol, up to each protocol-related output signals are available. The number of actually used outputs depends on the selected protocol. They can be classified according to their meaning for the protocols.

Selected Protocol		UART
Control Output	USCIx_CTL0	X
	USCIx_CTL1	nRTS
Data Output	USCIx_DAT0	X
	USCIx_DAT1	TX

Table 6.18-2 Output Signals for Different Protocol

6.18.5.3 Frame Format

A standard UART frame is shown in Figure 6.18-3. It consists of:

- An idle time with the signal level 1.

- One start of frame bit (SOF) with the signal level 0.
- 6~13 bit data
- A parity bit (P), programmable for either even or odd parity. It is optionally possible to handle frames without parity bit.
- One or two stop bits with the signal level 1.

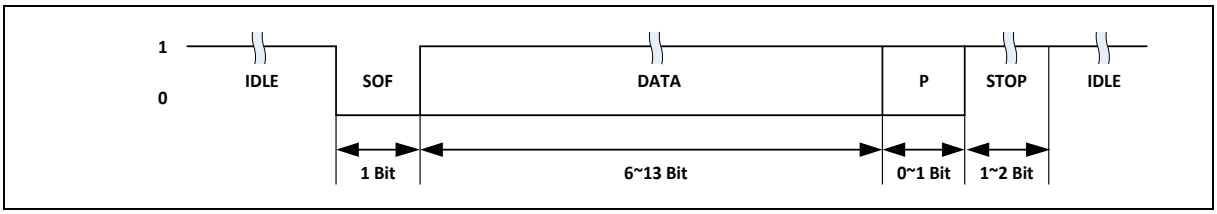


Figure 6.18-3 UART Standard Frame Format

The protocol specific bits (SOF, P, STOP) are automatically handled by the UART protocol state machine and do not appear in the data flow via the receive and transmit buffers.

Start Bit

The receiver input signal USCIx_DAT0 is checked for a falling edge. An SOF bit is detected when a falling edge occurs while the receiver is idle or after the sampling point of the last stop bit. To increase noise immunity, the SOF bit timing starts with the first falling edge that is detected. If the sampled bit value of the SOF is 1, the previous falling edge is considered to be due to noise and the receiver is considered to be idle again.

Data Field

The length of the data field (number of data bits) can be programmed by the bit field of DWIDTH (UUART_LINECTL[11:8]). It can vary between 6 to 13 data bits.

Note: In UART protocol, the data transmission order is LSB first by setting LSB (UUART_LINECTL[0]) to 1.

Parity Bit

The UART allows parity generation for transmission and parity check for reception on frame base. The type of parity can be selected by bit field PARITYEN (UUART_PROTCTL[1]) and EVENPARITY (UUART_PROTCTL[2]), common for transmission and reception (no parity, even or odd parity). If the parity handling is disabled, the UART frame does not contain any parity bit. For consistency reasons, all communication partners have to be programmed to the same parity mode.

After the last data bit of the data field, the transmitter automatically sends out its calculated parity bit if parity generation has been enabled. The receiver interprets this bit as received parity and compares it to its internally calculated one. The result of the parity check and frame check (STOP bit) are monitored in the protocol status registers (UUART_PROTSTS). The register contains bits to monitor a protocol-related status and protocol-related error indication (FRMERR, PARITYERR).

Stop Bit

Each UART frame is completed by 1 or 2 of stop bits with the signal level 1 (same level as the idle level). The number of stop bits is programmable by bit STOPB (UUART_PROTCTL[0]). A new start bit can be transferred directly after the last stop bit.

Transfer Status Indication

RXBUSY (UUART_PROTSTS[10]) indicates the receiver status.

The receiver status can be monitored by RXBUSY bit. In this case, bit RXBUSY is set during a complete frame reception from the beginning of the start of frame bit to the end of the last stop bit.

6.18.5.4 Operating Mode

To operate the UART protocol, the following issues have to be considered:

Select UART Mode

The UART protocol can be selected by setting FUNMODOE (UUART_CTL[2:0]) to 010B and the UART protocol can be enabled by setting PROTEN (UUART_PROTCTL [31]) to 1. Note that the FUNMODE must be set 0 before protocol changing and it is recommended to configure all parameters of the UART before UART protocol is enabled.

Pin Connections

The USCIx_DAT0 pin is used for UART receive data input signal (RX) in UART protocol. The property of input data signal can be configured in UUART_DATIN0. It is suggested to set EDGEDET (UUART_DATIN0[4:3]) as 10B for start bit detection.

The USCIx_DAT1 pin is used for UART transmit data output signal (TX) in UART protocol. The property of output data signal can be configured in UUART_LINECTL.

The USCIx_CTL0 pin is used for UART clear to send signal (nCTS) in UART protocol. The property of input control signal can be configured in UUART_CTLIN0.

The USCIx_CTL1 pin is used for UART request to send signal (nRTS) in UART protocol. The property of output control signal can be configured in UUART_LINECTL.

Bit Timing Configuration

The desired baud rate setting has to be selected, comprising the baud rate generator and the bit timing.

Frame Format Configuration

The word length, the stop bit number, and the parity mode has to be set up according to the application requirements by programming UUART_LINECTL and the UUART_PROTCTL register. If required by the application, the data input and output signals can be inverted. The data transmission order is LSB first by setting LSB (UUART_LINECTL[0]) to 1.

6.18.5.5 Bit Timing

In UART mode, each frame bit is divided into data sample time in order to provide granularity in the sub-bit range to adjust the sample point to the application requirements. The number of data sample time per bit is defined by bit fields DSCNT (UUART_BRGEN[14:10]) and the length of a data sample time is given by PDSCNT (UUART_BRGEN[9:8]).

In the example given in Figure 6.18-4, one bit time is composed of 16 data sample time DSCNT(UUART_BRGEN[14:10]) = 15. It is not recommended to program less and equal than 4 data sample time per bit time.

The position of the sampling point for the bit value is fixed in 1/2 samples time. It is possible to sample the bit value to take the average of samples.

The bit timing setup (number of data sample time) is common for the transmitter and the receiver because they use the same hardware circuit.

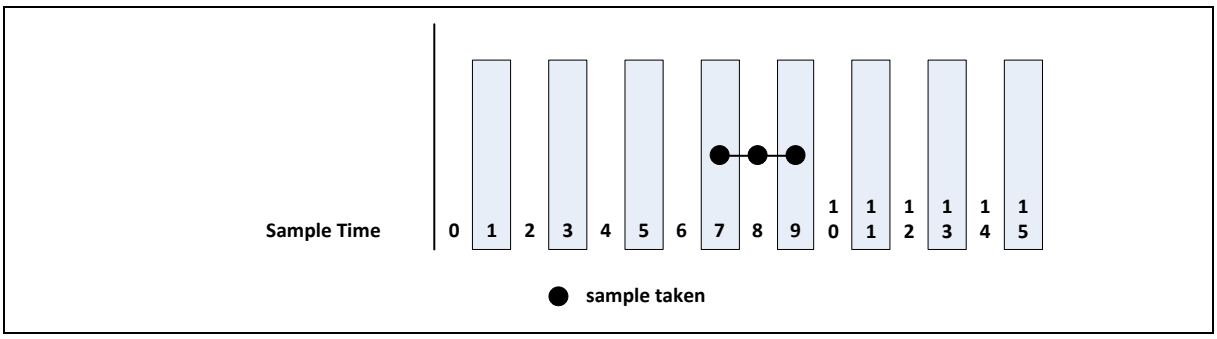


Figure 6.18-4 UART Bit Timing (data sample time)

6.18.5.6 Baud Rate Generation

The baud rate fUART in UART mode depends on the number of data sample time per bit time and their timing. The baud rate setting should only be changed while the transmitter and the receiver are idle. The bits RCLKSEL, SPCLKSEL, PDSCNT, and DSCNT define the baud rate setting:

RCLKSEL (UUART_BRGEN [0])

to define the input frequency f_{REF_CLK}

SPCLKSEL (UUART_BRGEN[3:2])

to define the multiple source of the sample clock f_{SAM_CLK}

PDSCNT (UUART_BRGEN [9:8])

to define the length of a data sample time (division of f_{REF_CLK} by 1, 2, 3, or 4)

DSCNT (UUART_BRGEN [14:10])

to define the number of data sample time per bit time

The standard setting is given by RCLKSEL = 0 ($f_{REF\_CLK} = f_{PCLK}$), PTCLKSEL = 0 ($f_{PROT\_CLK} = f_{REF\_CLK}$) and SPCLKSEL = 2'b00 (fSAMP_CLK = f_{DIV_CLK}). Under these conditions, the baud rate is given by:

$$f_{\mathrm{UART}} = f_{REF\_CLK} \times \frac{1}{\mathrm{CLKDIV}+1} \times \frac{1}{\mathrm{PDSCNT}+1} \times \frac{1}{\mathrm{DSCNT}+1}$$

In order to generate slower frequencies, additional divide-by-2 stages can be selected by PTCLKSEL = 1 ($f_{PROT\_CLK} = f_{REF\_CLK2}$), leading to:

$$f_{\mathrm{UART}} = \frac{f_{REF\_CLK}}{2} \times \frac{1}{\mathrm{CLKDIV}+1} \times \frac{1}{\mathrm{PDSCNT}+1} \times \frac{1}{\mathrm{DSCNT}+1}$$

If SPCLKSEL = 2'b10 (fSAMP_CLK = f_{SCLK}), and RCLKSEL = 0 ($f_{REF\_CLK} = f_{PCLK}$), PTCLKSEL = 0 ($f_{PROT\_CLK} = f_{REF\_CLK}$). The baud rate is given by:

$$f_{\mathrm{UART}} = f_{REF\_CLK} \times \frac{1}{\mathrm{CLKDIV}+1} \times \frac{1}{2} \times \frac{1}{\mathrm{PDSCNT}+1} \times \frac{1}{\mathrm{DSCNT}+1}$$

There is error tolerance for the UART baud rate after setting the baud rate parameter.Table 6.18-3 lists the relative error percentage examples for user to calculate his relative baud rate setting.

HCLK Source	PCLK Source	Expect Baud Rate	CLKDIV (UUART_BRGEN[25:16])	DSCNT (UUART_BRGEN[14:10])	PDSCNT	Active Baud Rate	Error Percentage
12 MHz	HCLK	115200	0xC	0x7	0x0	115384	0.16%
12 MHz	HCLK	9600	0x7C	0x9	0x0	9600	0%
12 MHz	HCLK	2400	0x1F3	0x9	0x0	2400	0%

Table 6.18-3 Baud Rate Relationship

Note: {SPCLKSEL, PTCLKSEL, RCLKSEL = 2'b0,1'b0,1'b0}

6.18.5.7 Auto Baud Rate Detection

The UART controller supports auto baud rate detection function. It is used to identify the input baud rate from the receiver signal (USCIx_DAT0) and then revised the baud rate clock divider CLKDIV (UUART_BRGEN[25:16]) after the baud rate function done to meet the detected baud rate information. According the section of Timing Measurement Counter, the timing measurement counter is used for time interval measurement of the input signal (USCIx_DAT0) and the actual timer value is captured into bit field BRDETITV (UUART_PROTCTL [24:16]) in each falling edge of the detected signal.

When the ABREN (UUART_PROTCTL[6]) bit is enabled, the 0x55 data patterns is necessary for auto baud rate detection. The falling edge of input signal starts the baud rate counter and it loads the timing

measurement counter value into the BRDETITV (UUART_PROTCTL [24:16]) in the next falling edge. It is suggested to use the fDIV_CLK (TMCNTSRC (UUART_BRGEN[5]) =1) as the counter source.

The CLKDIV (UUART_BRGEN[25:16]) will be revised by BRDETITV (UUART_PROTCTL [25:16]) after the auto baud rate function done (the time of 4th falling edge of input signal). If the user want to receive the next successive frame correctly, it is better to set the value of CLKDIV (UUART_BRGEN[25:16]) and DSCNT (UUART_BRGEN[14:10]) as the same value (the value shall be among the rang of 0xF and 0x5 because the DSCNT is used to define the sample counter of each bit and the PDSCNT (UUART_BRGEN[9:8]) is 0x0.

During the auto baud rate detection, the ABRDETIF (UUART_PROTSTS[9]) and the BRDETITV (UUART_PROTCTL [24:16]) will be updated after each falling edge of input signal and the auto baud rate pattern, 0x55, won't be received into the receiver buffer after the frame done. The bit of ABREN will be cleared by hardware after the 4th falling edge of input signal is detected thus the user can read the status of ABREN to know the auto baud rate function is done or not.

If the CLKDIV and DSCNT are not set as the same value in calculation the auto baud rate function, the user shall calculate the proper average baud rate by the value of BRDETITV and CLKDIV after the auto baud rate function done.

If the baud rate of input signal is very slower and the bit time of timing measurement counter can't calculate the correct period of the input bit time, there is a ABERRSTS bit (UUART_PROTSTS[11]) to indicate the error information of the auto baud rate detection. At this time, the user shall revise the value of CLKDIV and require the Host device to send the 0x55 pattern again.

According the limitation of timing measurement counter, the maximum auto baud rate detection is 0x1FE for BRDETITV. The UART Auto Baud Rate Control is shown in Figure 6.18-5.

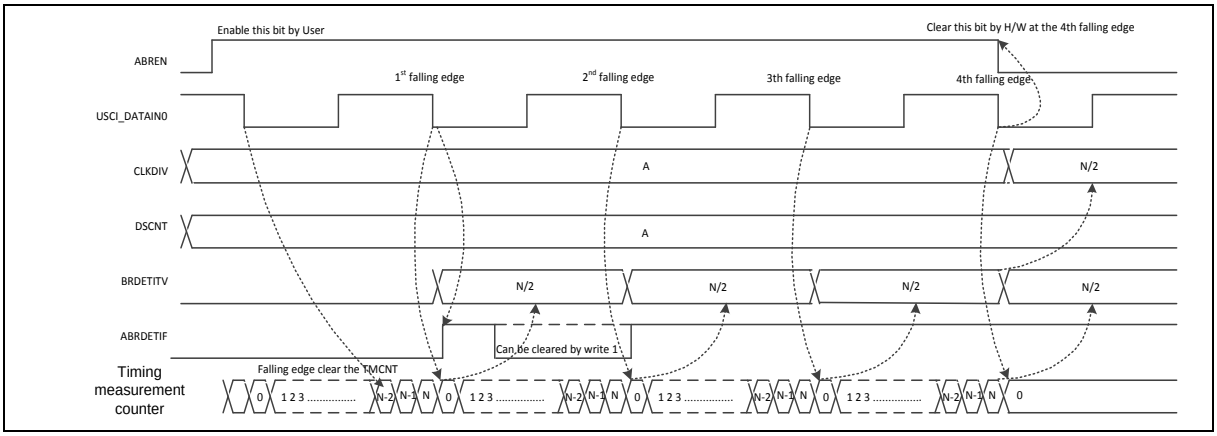


Figure 6.18-5 UART Auto Baud Rate Control

6.18.5.8 Auto Flow Control

The UART supports hardware auto-flow control that provides nRTS flow control by indicator RXFULL (UUART_BUFSTS[1]) on receiver buffer. When the buffer is full (RXFULL = 1), the nRTS is de-asserted.

The UART also provides nCTS flow control on transmitter. The nCTS is used to control the transmitted data is sent out when the nCTS is asserted.

6.18.5.9 RS-485 Support

The UART controller can play the role of the RS-485 master transmitter will identify an address character by setting the parity (9-th bit) to 1. For data characters, the parity is set to 0. Software can use the bit15 of each data to control the parity bit (PARITYEN (UUART_PROTCTL[1]) be set) when the STICKEN (UUART_PROTCTL[26]) is set. For example, if the STICKEN is set to 1 and data sequence are 0x8015, 0x8033, 0x0055, 0x0033 and 0x80AA the transmitted parity of data 0x15, 0x33, 0x55 0x33 and 0xAA will be 1, 1, 0, 0 and 1.

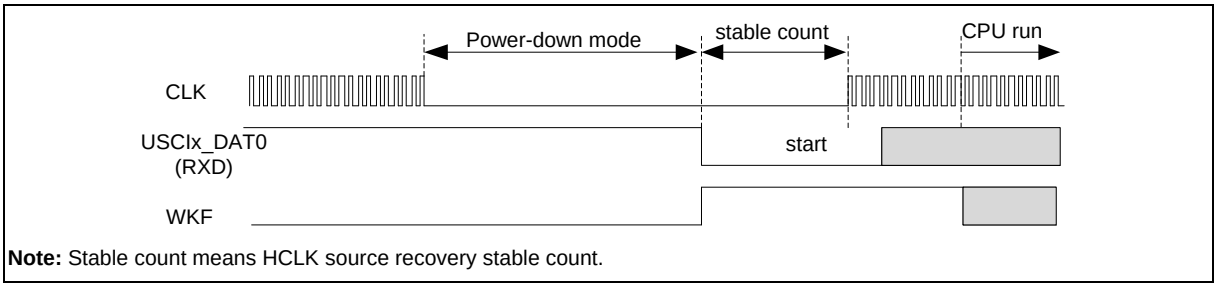
The UART controller can also play as an RS-485 addressable slave, the protocol-related error of PARITYERR (UUART_PROTSTS[5]) can be acted as the address bit detection when the PARITYEN (UUART_PROTCTL[1]), EVENPARITY (UUART_PROTCTL[2]) and STICKEN (UUART_PROTCTL[26]) were set. If the PARITYERR was set, it means that the address bit in the received bus is detected otherwise, the data is received into Buffer.

6.18.5.10 Wake-up Function

The USCI Controller in UART mode supports wake-up system function. The wake-up source includes incoming data and nCTS pin. Each wake-up source description is as follows:

(a) Incoming data wake-up

When system is in power-down and both of the WKEN (UUART_WKCTL [0]) and DATWKEN (UUART_PROTCTL[9]) are set, the toggle of incoming data pin can wake-up the system. In order to receive the incoming data after the system wake-up, the WAKECNT (UUART_PROTCTL[14:11]) shall be set. These bits field of WAKECNT (UUART_PROCTL[14:11]) indicate how many clock cycle selected by fPDS_CNT do the controller can get the 1st bit (start bit) when the device is wakeup from Power-down mode. The incoming data wake-wp is shown in Figure 6.18-6.



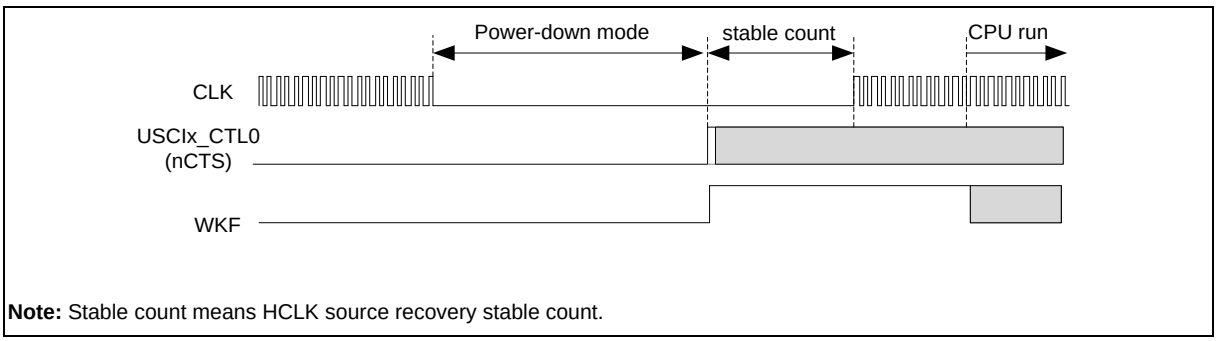
Note: Stable count means HCLK source recovery stable count.

Figure 6.18-6 Incoming Data Wake-Up

(b) nCTS pin wake-up

When system is in power-down and both of the WKEN (UUART_WKCTL [0]) and CTSWKEN (UUART_PROTCTL[10]) are set, the toggle of nCTS pin can wake-up the system. The nCTS wake-wp is shown in Figure 6.18-7 and Figure 6.18-8.

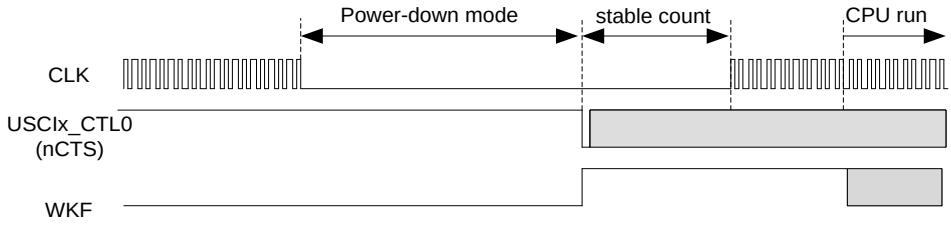
Case 1(nCTS transition from low to high):



Note: Stable count means HCLK source recovery stable count.

Figure 6.18-7 nCTS Wake-Up Case 1

Case 2 (nCTS transition from high to low):



Note 1: Stable count means HCLK source recovery stable count.

Figure 6.18-8 nCTS Wake-Up Case 2

6.18.5.11 Interrupt Events

The UART provided interrupt for protocol event and data transfer event. The description show below:

Protocol Interrupt Events

The following protocol-related events are generated in UART mode and can lead to a protocol interrupt.

Please note that the bits in register UUART_PROTSTS are not automatically cleared by hardware and have to be cleared by software in order to monitor new incoming events.

Receiver Line Status

The protocol-related error FRMERR (UUART_PROTSTS[6]) or PARITYERR (UUART_PROTSTS[5]) are two flags that are assigned to each received data word in the corresponding receiver buffer status registers.

In UART mode, the result of the parity check by the protocol-related error indication (0 = received parity bit equal to calculated parity value), and the result of frame check by the protocol-related error indication (0 = received stop bit equal to the format value ‘1’). This information is elaborated for each data frame.

The break error flag BREAK (UUART_PROTSTS[7]) is assigned when the receive data is 0, the received parity and the stop bit are also 0.

The interrupt indicates that there are parity error, frame error or the break data detection in the BREAK, FRMERR, PARITYERR (UUART_PROTSTS[7:5]) bits.

Auto Baud Rate Detection

The auto baud rate interrupt, ABRDETIF (UUART_PROTSTS [9]), indicates that the timing measurement counter has getting 2-bit duration for auto baud rate capture function.

The auto baud rate detection function will be enabled in the first falling edge of receiver signal. The auto baud rate detection function is measurement after the next following falling is detected and it is finished when the frame transfer done. After the transfer done, the timing measurement counter value divided by twice is equal to the number of sample time per bit. The user can read the value of BRDETITV (UUART_PROTCTL[24:16]) and write into the baud rate generator register CLKDIV (UUART_BRGEN[25:16]).

Data Transfer Interrupt Handling

The data transfer interrupts indicate events related to UART frame handling.

Transmit Start Interrupt

Bit TXSTIF (UUART_PROTSTS [1]) is set after the start bit of a data word. In buffer mode, this is the earliest point in time when a new data word can be written to UUART_TXDAT.

Transmitter Finished

This interrupt indicates that the transmitter has completely finished all data in the buffer. Bit TXENDIF (UUART_PROTSTS [2]) becomes set at the end of the last stop bit.

Receiver Starts Interrupt

Bit RXSTIF (UUART_PROTSTS [3]) is set after the sample point of the start bit.

Receiver Frame Finished

This interrupt indicates that the receiver has completely finished a frame. Bit RXENDIF (UUART_PROTSTS [4]) becomes set at the end of the last receive bit.

6.18.5.12 Programming Example

The following steps are used to configure the UART protocol setting and the data transmission.

1. Set FUNMODE (UUART_CTL[2:0]) to 0x2 to select UART protocol.
2. Write baud rate generator register UUART_BRGEN to select desired baud rate.
   - Set SPCLKSEL (UUART_BRGEN[3:2]), PTCLKSEL (UUART_BRGEN[1]) and RCLKSEL (UUART_BRGEN[0]) to select the clock source.
   - Configure CLKDIV (UUART_BRGEN[25:16]), DSCNT (UUART_BRGEN[14:10]) and PDSCNT (UUART_BRGEN[9:8]) to determine the baud rate divider.
3. Write line control register UUART_LINECTL and protocol control register UUART_PROTCTL to configure the transmission data format and UART protocol setting.
   - Program data field length in DWIDTH (UUART_LINECTL[11:8]).
   - Enable parity bit and determine the parity bit type by setting EVENPARITY (UUART_PROTCTL[2]) and PARITYEN (UUART_PROTCTL[1]).
   - Configure stop bit length by setting STOPB (UUART_PROTCTL[0]).
   - Enable LSB (UUART_LINECTL[0]) to select LSB first transmission for UART protocol.
   - Set EDGEDET (UUART_DATIN0[4:3]) to “10” to select the detected edge as falling edge for receiver start bit detection.
4. Set PROTEN (UUART_PROTCTL[31]) to 1 to enable UART protocol.
5. Transmit and receive data.
   - Write transmit data register UUART_TXDAT to transmit data.
   - Wait until TXSTIF(UUART_PROTSTS[1]) is set and then user can write the next data in UUART_TXDAT.
   - When TXENDIF(UUART_PROTSTS[2]) is set, the transmit buffer is empty and the stop bit of the last data has been transmitted.
   - If RXENDIF(UUART_PROTSTS[4]) is set, the receiver has finished a data frame completely. User can get the data by reading receive data register UUART_RXDAT.

6.18.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
USCI_UART Base Address: **UUARTn_BA = 0x400D_0000 + (0x1000*n)** **n= 0, 1**				
UUART_CTL	UUARTn_BA+0x00	R/W	USCI Control Register	0x0000_0000
UUART_INTEN	UUARTn_BA+0x04	R/W	USCI Interrupt Enable Register	0x0000_0000
UUART_BRGEN	UUARTn_BA+0x08	R/W	USCI Baud Rate Generator Register	0x0000_3C00
UUART_DATIN0	UUARTn_BA+0x10	R/W	USCI Input Data Signal Configuration Register 0	0x0000_0000
UUART_CTLIN0	UUARTn_BA+0x20	R/W	USCI Input Control Signal Configuration Register 0	0x0000_0000
UUART_CLKIN	UUARTn_BA+0x28	R/W	USCI Input Clock Signal Configuration Register	0x0000_0000
UUART_LINECTL	UUARTn_BA+0x2C	R/W	USCI Line Control Register	0x0000_0000
UUART_TXDAT	UUARTn_BA+0x30	W	USCI Transmit Data Register	0x0000_0000
UUART_RXDAT	UUARTn_BA+0x34	R	USCI Receive Data Register	0x0000_0000
UUART_BUFCTL	UUARTn_BA+0x38	R/W	USCI Transmit/Receive Buffer Control Register	0x0000_0000
UUART_BUFSTS	UUARTn_BA+0x3C	R	USCI Transmit/Receive Buffer Status Register	0x0000_0101
UUART_PDMACTL	UUARTn_BA+0x40	R/W	USCI PDMA Control Register	0x0000_0000
UUART_WKCTL	UUARTn_BA+0x54	R/W	USCI Wake-up Control Register	0x0000_0000
UUART_WKSTS	UUARTn_BA+0x58	R/W	USCI Wake-up Status Register	0x0000_0000
UUART_PROTCTL	UUARTn_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0000
UUART_PROTIEN	UUARTn_BA+0x60	R/W	USCI Protocol Interrupt Enable Register	0x0000_0000
UUART_PROTSTS	UUARTn_BA+0x64	R/W	USCI Protocol Status Register	0x0000_0000

6.18.7 Register Description

USCI Control Register (UUART_CTL)

Register	Offset	R/W	Description	Reset Value
UUART_CTL	UUARTn_BA+0x00	R/W	USCI Control Register	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**FUNMODE**	**Function Mode** This bit field selects the protocol for this USCI controller. Selecting a protocol that is not available or a reserved combination disables the USCI. When switching between two protocols, the USCI has to be disabled before selecting a new protocol. Simultaneously, the USCI will be reset when user write 000 to FUNMODE. 000 = The USCI is disabled. All protocol related state machines are set to idle state. 001 = The SPI protocol is selected. 010 = The UART protocol is selected. 100 = The I^2C protocol is selected. **Note:** Other bit combinations are reserved.

USCI Interrupt Enable Register (UUART_INTEN)

Register	Offset	R/W	Description	Reset Value
UUART_INTEN	UUARTn_BA+0x04	R/W	USCI Interrupt Enable Register	0x0000_0000

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

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**RXENDIEN**	**Receive End Interrupt Enable Bit** This bit enables the interrupt generation in case of a receive finish event. 0 = The receive end interrupt Disabled. 1 = The receive end interrupt Enabled.
[3]	**RXSTIEN**	**Receive Start Interrupt Enable BIt** This bit enables the interrupt generation in case of a receive start event. 0 = The receive start interrupt Disabled. 1 = The receive start interrupt Enabled.
[2]	**TXENDIEN**	**Transmit End Interrupt Enable Bit** This bit enables the interrupt generation in case of a transmit finish event. 0 = The transmit finish interrupt Disabled. 1 = The transmit finish interrupt Enabled.
[1]	**TXSTIEN**	**Transmit Start Interrupt Enable Bit** This bit enables the interrupt generation in case of a transmit start event. 0 = The transmit start interrupt Disabled. 1 = The transmit start interrupt Enabled.
[0]	**Reserved**	Reserved.

USCI Baud Rate Generator Register (UUART_BRGEN)

Register	Offset	R/W	Description	Reset Value
UUART_BRGEN	UUARTn_BA+0x08	R/W	USCI Baud Rate Generator Register	0x0000_3C00

31	30	29	28	27	26	25	24
Reserved						CLKDIV	
23	22	21	20	19	18	17	16
CLKDIV							
15	14	13	12	11	10	9	8
Reserved	DSCNT					PDSCNT	
7	6	5	4	3	2	1	0
Reserved		TMCNTSRC	TMCNTEN	SPCLKSEL		PTCLKSEL	RCLKSEL

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25:16]	**CLKDIV**	**Clock Divider** This bit field defines the ratio between the protocol clock frequency f_{PROT_CLK} and the clock divider frequency f_{DIV_CLK} ($f_{DIV\_CLK} = f_{PROT\_CLK} / (CLKDIV+1)$). **Note:** In UART function, it can be updated by hardware in the 4th falling edge of the input data 0x55 when the auto baud rate function (ABREN(UUART_PROTCTL[6])) is enabled. The revised value is the average bit time between bit 5 and bit 6. The user can use revised CLKDIV and new BRDETITV (UUART_PROTCTL[24:16]) to calculate the precise baud rate.
[15]	**Reserved**	Reserved.
[14:10]	**DSCNT**	**Denominator for Sample Counter** This bit field defines the divide ratio of the sample clock f_{SAMP_CLK}. The divided frequency $f_{DS\_CNT} = f_{PDS\_CNT} / (DSCNT+1)$. **Note:** The maximum value of DSCNT is 0xF on UART mode and suggest to set over 4 to confirm the receiver data is sampled in right value.
[9:8]	**PDSCNT**	**Pre-divider for Sample Counter** This bit field defines the divide ratio of the clock division from sample clock f_{SAMP_CLK}. The divided frequency $f_{PDS\_CNT} = f_{SAMP\_CLK} / (PDSCNT+1)$.
[7:6]	**Reserved**	Reserved.
[5]	**TMCNTSRC**	**Timing Measurement Counter Clock Source Selection** 0 = Timing measurement counter with f_{PROT_CLK}. 1 = Timing measurement counter with f_{DIV_CLK}.
[4]	**TMCNTEN**	**Timing Measurement Counter Enable Bit** This bit enables the 10-bit timing measurement counter. 0 = Timing measurement counter is Disabled. 1 = Timing measurement counter is Enabled.
[3:2]	**SPCLKSEL**	**Sample Clock Source Selection** This bit field used for the clock source selection of a sample clock (f_{SAMP_CLK}) for the

		protocol processor. 00 = f_{SAMP_CLK} = f_{DIV_CLK}. 01 = f_{SAMP_CLK} = f_{PROT_CLK}. 10 = f_{SAMP_CLK} = f_{SCLK}. 11 = f_{SAMP_CLK} = f_{REF_CLK}.
[1]	**PTCLKSEL**	**Protocol Clock Source Selection** This bit selects the source signal of protocol clock (f_{PROT_CLK}). 0 = Reference clock f_{REF_CLK}. 1 = f_{REF_CLK2} (its frequency is half of f_{REF_CLK}).
[0]	**RCLKSEL**	**Reference Clock Source Selection** This bit selects the source signal of reference clock (f_{REF_CLK}). 0 = Peripheral device clock f_{PCLK}. 1 = Reserved.

USCI Input Data Signal Configuration (UUART_DATIN0)

Register	Offset	R/W	Description	Reset Value
UUART_DATIN0	UUARTn_BA+0x10	R/W	USCI Input Data Signal Configuration Register 0	0x0000_0000

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

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4:3]	**EDGEDET**	**Input Signal Edge Detection Mode** This bit field selects which edge actives the trigger event of input data signal. 00 = The trigger event activation is disabled. 01 = A rising edge activates the trigger event of input data signal. 10 = A falling edge activates the trigger event of input data signal. 11 = Both edges activate the trigger event of input data signal. **Note:** In UART function mode, it is suggested to set this bit field as 10.
[2]	**ININV**	**Input Signal Inverse Selection** This bit defines the inverter enable of the input asynchronous signal. 0 = The un-synchronized input signal will not be inverted. 1 = The un-synchronized input signal will be inverted.
[1]	**Reserved**	Reserved.
[0]	**SYNCSEL**	**Input Signal Synchronization Selection** This bit selects if the un-synchronized input signal (with optionally inverted) or the synchronized (and optionally filtered) signal can be used as input for the data shift unit. 0 = The un-synchronized signal can be taken as input for the data shift unit. 1 = The synchronized signal can be taken as input for the data shift unit.

USCI Input Control Signal Configuration (UUART_CTLIN0)

Register	Offset	R/W	Description	Reset Value
UUART_CTLIN0	UUARTn_BA+0x20	R/W	USCI Input Control Signal Configuration Register 0	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**ININV**	**Input Signal Inverse Selection** This bit defines the inverter enable of the input asynchronous signal. 0 = The un-synchronized input signal will not be inverted. 1 = The un-synchronized input signal will be inverted.
[1]	**Reserved**	Reserved.
[0]	**SYNCSEL**	**Input Synchronization Signal Selection** This bit selects if the un-synchronized input signal (with optionally inverted) or the synchronized (and optionally filtered) signal can be used as input for the data shift unit. 0 = The un-synchronized signal can be taken as input for the data shift unit. 1 = The synchronized signal can be taken as input for the data shift unit.

USCI Input Clock Signal Configuration (UUART_CLKIN)

Register	Offset	R/W	Description	Reset Value
UUART_CLKIN	UUARTn_BA+0x28	R/W	USCI Input Clock Signal Configuration Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**SYNCSEL**	**Input Synchronization Signal Selection** This bit selects if the un-synchronized input signal or the synchronized (and optionally filtered) signal can be used as input for the data shift unit. 0 = The un-synchronized signal can be taken as input for the data shift unit. 1 = The synchronized signal can be taken as input for the data shift unit.

USCI Line Control Register (UUART_LINECTL)

Register	Offset	R/W	Description	Reset Value
UUART_LINECTL	UUARTn_BA+0x2C	R/W	USCI Line Control Register	0x0000_0000

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

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:8]	**DWIDTH**	**Word Length of Transmission** This bit field defines the data word length (amount of bits) for reception and transmission. The data word is always right-aligned in the data buffer. USCI support word length from 4 to 16 bits. 0x0: The data word contains 16 bits located at bit positions [15:0]. 0x1: Reserved. 0x2: Reserved. 0x3: Reserved. 0x4: The data word contains 4 bits located at bit positions [3:0]. 0x5: The data word contains 5 bits located at bit positions [4:0]. ... 0xF: The data word contains 15 bits located at bit positions [14:0]. **Note:** In UART protocol, the length can be configured as 6~13 bits.
[7]	**CTLOINV**	**Control Signal Output Inverse Selection** This bit defines the relation between the internal control signal and the output control signal. 0 = No effect. 1 = The control signal will be inverted before its output. **Note:** In UART protocol, the control signal means nRTS signal.
[6]	**Reserved**	Reserved.
[5]	**DATOINV**	**Data Output Inverse Selection** This bit defines the relation between the internal shift data value and the output data signal of USCIx_DAT1 pin. 0 = The value of USCIx_DAT1 is equal to the data shift register. 1 = The value of USCIx_DAT1 is the inversion of data shift register.
[4:1]	**Reserved**	Reserved.

[0]	**LSB**	**LSB First Transmission Selection** 0 = The MSB, which bit of transmit/receive data buffer depends on the setting of DWIDTH, is transmitted/received first. 1 = The LSB, the bit 0 of data buffer, will be transmitted/received first.

USCI Transmit Data Register (UUART_TXDAT)

Register	Offset	R/W	Description	Reset Value
UUART_TXDAT	UUARTn_BA+0x30	W	USCI Transmit Data Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**TXDAT**	**Transmit Data** Software can use this bit field to write 16-bit transmit data for transmission.

USCI Receive Data Register (UUART_RXDAT)

Register	Offset	R/W	Description	Reset Value
UUART_RXDAT	UUARTn_BA+0x34	R	USCI Receive Data Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**RXDAT**	**Received Data** This bit field monitors the received data which stored in receive data buffer. **Note:** RXDAT[15:13] indicate the same frame status of BREAK, FRMERR and PARITYERR (UUART_PROTSTS[7:5]).

USCI Transmitter/Receive Buffer Control Register (UUART_BUFCTL)

Register	Offset	R/W	Description	Reset Value
UUART_BUFCTL	UUARTn_BA+0x38	R/W	USCI Transmit/Receive Buffer Control Register	0x0000_0000

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

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**RXRST**	**Receive Reset** 0 = No effect. 1 = Reset the receive-related counters, state machine, and the content of receive shift register and data buffer. **Note 1:** It is cleared automatically after one PCLK cycle. **Note 2:** It is suggested to check the RXBUSY (UUART_PROTSTS[10]) before this bit will be set to 1.
[16]	**TXRST**	**Transmit Reset** 0 = No effect. 1 = Reset the transmit-related counters, state machine, and the content of transmit shift register and data buffer**.** **Note:** It is cleared automatically after one PCLK cycle.
[15]	**RXCLR**	**Clear Receive Buffer** 0 = No effect. 1 = The receive buffer is cleared (filling level is cleared and output pointer is set to input pointer value). Should only be used while the buffer is not taking part in data traffic. **Note:** It is cleared automatically after one PCLK cycle.
[14]	**RXOVIEN**	**Receive Buffer Overrun Error Interrupt Enable Bit** 0 = Receive overrun interrupt Disabled. 1 = Receive overrun interrupt Enabled.
[13:8]	**Reserved**	Reserved.
[7]	**TXCLR**	**Clear Transmit Buffer** 0 = No effect. 1 = The transmit buffer is cleared (filling level is cleared and output pointer is set to input pointer value). Should only be used while the buffer is not taking part in data traffic. **Note:** It is cleared automatically after one PCLK cycle.

[6:0]	**Reserved**	Reserved.

USCI Transmit/Receive Buffer Status Register (UUART_BUFSTS)

Register	Offset	R/W	Description	Reset Value
UUART_BUFSTS	UUARTn_BA+0x3C	R	USCI Transmit/Receive Buffer Status Register	0x0000_0101

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

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**TXFULL**	**Transmit Buffer Full Indicator** 0 = Transmit buffer is not full. 1 = Transmit buffer is full.
[8]	**TXEMPTY**	**Transmit Buffer Empty Indicator** 0 = Transmit buffer is not empty. 1 = Transmit buffer is empty.
[7:4]	**Reserved**	Reserved.
[3]	**RXOVIF**	**Receive Buffer Over-run Error Interrupt Status** This bit indicates that a receive buffer overrun error event has been detected. If RXOVIEN (UUART_BUFCTL[14]) is enabled, the corresponding interrupt request is activated. It is cleared by software writes 1 to this bit. 0 = A receive buffer overrun error event has not been detected. 1 = A receive buffer overrun error event has been detected.
[2]	**Reserved**	Reserved.
[1]	**RXFULL**	**Receive Buffer Full Indicator** 0 = Receive buffer is not full. 1 = Receive buffer is full.
[0]	**RXEMPTY**	**Receive Buffer Empty Indicator** 0 = Receive buffer is not empty. 1 = Receive buffer is empty.

USCI PDMA Control Register (UUART_PDMACTL)

Register	Offset	R/W	Description	Reset Value
UUART_PDMACTL	UUARTn_BA+0x40	R/W	USCI PDMA Control Register	0x0000_0000

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

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**PDMAEN**	**PDMA Mode Enable Bit** 0 = PDMA function Disabled. 1 = PDMA function Enabled.
[2]	**RXPDMAEN**	**PDMA Receive Channel Available** 0 = Receive PDMA function Disabled. 1 = Receive PDMA function Enabled.
[1]	**TXPDMAEN**	**PDMA Transmit Channel Available** 0 = Transmit PDMA function Disabled. 1 = Transmit PDMA function Enabled.
[0]	**PDMARST**	**PDMA Reset** 0 = No effect. 1 = Reset the USCI's PDMA control logic. This bit will be cleared to 0 automatically.

USCI Wake-up Control Register (UUART_WKCTL)

Register	Offset	R/W	Description	Reset Value
UUART_WKCTL	UUARTn_BA+0x54	R/W	USCI Wake-up Control Register	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**PDBOPT**	**Power Down Blocking Option** 0 = If user attempts to enter Power-down mode by executing WFI while the protocol is in transferring, MCU will stop the transfer and enter Power-down mode immediately. 1 = If user attempts to enter Power-down mode by executing WFI while the protocol is in transferring, the on-going transfer will not be stopped and MCU will enter idle mode immediately.
[1]	**Reserved**	Reserved.
[0]	**WKEN**	**Wake-up Enable Bit** 0 = Wake-up function Disabled. 1 = Wake-up function Enabled.

USCI Wake-up Status Register (UUART_WKSTS)

Register	Offset	R/W	Description	Reset Value
UUART_WKSTS	UUARTn_BA+0x58	R/W	USCI Wake-up Status Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**WKF**	**Wake-up Flag** When chip is woken up from Power-down mode, this bit is set to 1. Software can write 1 to clear this bit.

USCI Protocol Control Register – UART (UUART_PROTCTL)

Register	Offset	R/W	Description	Reset Value
UUART_PROTCTL	UUARTn_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PROTEN	Reserved	BCEN	Reserved	Reserved	STICKEN	Reserved	BRDETITV
23	22	21	20	19	18	17	16
BRDETITV							
15	14	13	12	11	10	9	8
Reserved	WAKECNT				CTSWKEN	DATWKEN	Reserved
7	6	5	4	3	2	1	0
Reserved	ABREN	RTSAUDIREN	CTSAUTOEN	RTSAUTOEN	EVENPARITY	PARITYEN	STOPB

Bits	Description	
[31]	**PROTEN**	**UART Protocol Enable Bit** 0 = UART Protocol Disabled. 1 = UART Protocol Enabled.
[30]	**Reserved**	Reserved.
[29]	**BCEN**	**Transmit Break Control Enable Bit** 0 = Transmit Break Control Disabled. 1 = Transmit Break Control Enabled. **Note:** When this bit is set to logic 1, the serial data output (TX) is forced to the Spacing State (logic 0). This bit acts only on TX line and has no effect on the transmitter logic.
[27]	**Reserved**	Reserved.
[26]	**STICKEN**	**Stick Parity Enable Bit** 0 = Stick parity Disabled. 1 = Stick parity Enabled. **Note:** Refer to RS-485 Support section for detailed information.
[25]	**Reserved**	Reserved.
[24:16]	**BRDETITV**	**Baud Rate Detection Interval** This bit fields indicate how many clock cycle selected by TMCNTSRC (UUART_BRGEN [5]) does the slave calculates the baud rate in one bits. The order of the bus shall be 1 and 0 step by step (e.g. the input data pattern shall be 0x55). The user can read the value to know the current input baud rate of the bus whenever the ABRDETIF (UUART_PROTSTS[9]) is set. **Note:** This bit can be cleared to 0 by software writing '0' to the **BRDETITV**.
[15]	**Reserved**	Reserved.
[14:11]	**WAKECNT**	**Wake-up Counter** These bits field indicate how many clock cycle selected by f_{PDS_CNT} do the slave can get the 1st bit (start bit) when the device is wake-up from Power-down mode.
[10]	**CTSWKEN**	**nCTS Wake-up Mode Enable Bit**

		0 = nCTS wake-up mode Disabled. 1 = nCTS wake-up mode Enabled.
[9]	**DATWKEN**	**Data Wake-up Mode Enable Bit** 0 = Data wake-up mode Disabled. 1 = Data wake-up mode Enabled.
[6]	**ABREN**	**Auto-baud Rate Detect Enable Bit** 0 = Auto-baud rate detect function Disabled. 1 = Auto-baud rate detect function Enabled. **Note:** When the auto - baud rate detect operation finishes, hardware will clear this bit. The associated interrupt ABRDETIF (UUART_PROTST[9]) will be generated (If ARBIEN (UUART_PROTIEN [1]) is enabled).
[5]	**RTSAUDIREN**	**nRTS Auto Direction Enable Bit** When nRTS auto direction is enabled, if the transmitted bytes in the TX buffer is empty, the UART asserted nRTS signal automatically. 0 = nRTS auto direction control Disabled. 1 = nRTS auto direction control Enabled. **Note 1:** This bit is used for nRTS auto direction control for RS485. **Note 2:** This bit has effect only when the RTSAUTOEN is not set.
[4]	**CTSAUTOEN**	**nCTS Auto-flow Control Enable Bit** When nCTS auto-flow is enabled, the UART will send data to external device when nCTS input assert (UART will not send data to device if nCTS input is dis-asserted). 0 = nCTS auto-flow control Disabled. 1 = nCTS auto-flow control Enabled.
[3]	**RTSAUTOEN**	**nRTS Auto-flow Control Enable Bit** When nRTS auto-flow is enabled, if the receiver buffer is full (RXFULL (UUART_BUFSTS[1] =1), the UART will de-assert nRTS signal. 0 = nRTS auto-flow control Disabled. 1 = nRTS auto-flow control Enabled. **Note:** This bit has effect only when the RTSAUDIREN is not set.
[2]	**EVENPARITY**	**Even Parity Enable Bit** 0 = Odd number of logic 1's is transmitted and checked in each word. 1 = Even number of logic 1's is transmitted and checked in each word. **Note:** This bit has effect only when PARITYEN is set.
[1]	**PARITYEN**	**Parity Enable Bit** This bit defines the parity bit is enabled in an UART frame. 0 = The parity bit Disabled. 1 = The parity bit Enabled.
[0]	**STOPB**	**Stop Bits** This bit defines the number of stop bits in an UART frame. 0 = The number of stop bits is 1. 1 = The number of stop bits is 2.

USCI Protocol Interrupt Enable Register – UART (UUART_PROTIEN)

Register	Offset	R/W	Description	Reset Value
UUART_PROTIEN	UUARTn_BA+0x60	R/W	USCI Protocol Interrupt Enable Register	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**RLSIEN**	**Receive Line Status Interrupt Enable Bit** 0 = Receive line status interrupt Disabled. 1 = Receive line status interrupt Enabled. **Note:** UUART_PROTSTS[7:5] indicates the current interrupt event for receive line status interrupt.
[1]	**ABRIEN**	**Auto-baud Rate Interrupt Enable Bit** 0 = Auto-baud rate interrupt Disabled. 1 = Auto-baud rate interrupt Enabled.
[0]	**Reserved**	Reserved.

USCI Protocol Status Register – UART (UUART_PROTSTS)

Register	Offset	R/W	Description	Reset Value
UUART_PROTSTS	UUARTn_BA+0x64	R/W	USCI Protocol Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						CTSLV	CTSSYNCLV
15	14	13	12	11	10	9	8
Reserved				ABERRSTS	RXBUSY	ABRDETIF	Reserved
7	6	5	4	3	2	1	0
BREAK	FRMERR	PARITYERR	RXENDIF	RXSTIF	TXENDIF	TXSTIF	Reserved

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**CTSLV**	**nCTS Pin Status (Read Only)** This bit used to monitor the current status of nCTS pin input. 0 = nCTS pin input is low level voltage logic state. 1 = nCTS pin input is high level voltage logic state.
[16]	**CTSSYNCLV**	**nCTS Synchronized Level Status (Read Only)** This bit used to indicate the current status of the internal synchronized nCTS signal. 0 = The internal synchronized nCTS is low. 1 = The internal synchronized nCTS is high.
[15:12]	**Reserved**	Reserved.
[11]	**ABERRSTS**	**Auto-baud Rate Error Status** This bit is set when auto-baud rate detection counter overrun. When the auto-baud rate counter overrun, the user shall revise the CLKDIV (UUART_BRGEN[25:16]) value and enable ABREN (UUART_PROTCTL[6]) to detect the correct baud rate again. 0 = Auto-baud rate detect counter is not overrun. 1 = Auto-baud rate detect counter is overrun. **Note 1:** This bit is set at the same time of ABRDETIF. **Note 2:** This bit can be cleared by writing “1” to ABRDETIF or ABERRSTS.
[10]	**RXBUSY**	**RX Bus Status Flag (Read Only)** This bit indicates the busy status of the receiver. 0 = The receiver is Idle. 1 = The receiver is BUSY.
[9]	**ABRDETIF**	**Auto-baud Rate Interrupt Flag** This bit is set when auto-baud rate detection is done among the falling edge of the input

		data. If the ABRIEN (UUART_PROTCTL[6]) is set, the auto-baud rate interrupt will be generated. This bit can be set 4 times when the input data pattern is 0x55 and it is cleared before the next falling edge of the input bus. 0 = Auto-baud rate detect function is not done. 1 = One Bit auto-baud rate detect function is done. **Note:** This bit can be cleared by writing “1” to it.
[8]	**Reserved**	Reserved.
[7]	**BREAK**	**Break Flag** This bit is set to logic 1 whenever the received data input (RX) is held in the “spacing state” (logic 0) for longer than a full word transmission time (that is, the total time of “start bit” + data bits + parity + stop bits). 0 = No Break is generated. 1 = Break is generated in the receiver bus. **Note:** This bit can be cleared by writing “1” among the BREAK, FRMERR and PARITYERR bits.
[6]	**FRMERR**	**Framing Error Flag** This bit is set to logic 1 whenever the received character does not have a valid “stop bit” (that is, the stop bit following the last data bit or parity bit is detected as logic 0). 0 = No framing error is generated. 1 = Framing error is generated. **Note:** This bit can be cleared by writing “1” among the BREAK, FRMERR and PARITYERR bits.
[5]	**PARITYERR**	**Parity Error Flag** This bit is set to logic 1 whenever the received character does not have a valid “parity bit”. 0 = No parity error is generated. 1 = Parity error is generated. **Note:** This bit can be cleared by writing “1” among the BREAK, FRMERR and PARITYERR bits.
[4]	**RXENDIF**	**Receive End Interrupt Flag** 0 = A receive finish interrupt status has not occurred. 1 = A receive finish interrupt status has occurred. **Note:** It is cleared by software writing 1 into this bit.
[3]	**RXSTIF**	**Receive Start Interrupt Flag** 0 = A receive start interrupt status has not occurred. 1 = A receive start interrupt status has occurred. **Note:** It is cleared by software writing 1 into this bit.
[2]	**TXENDIF**	**Transmit End Interrupt Flag** 0 = A transmit end interrupt status has not occurred. 1 = A transmit end interrupt status has occurred. **Note:** It is cleared by software writing 1 into this bit.
[1]	**TXSTIF**	**Transmit Start Interrupt Flag** 0 = A transmit start interrupt status has not occurred. 1 = A transmit start interrupt status has occurred. **Note 1:** It is cleared by software writing one into this bit. **Note 2:** Used for user to load next transmit data when there is no data in transmit buffer.
[0]	**Reserved**	Reserved.

6.19 USCI - SPI Mode

6.19.1 Overview

The SPI protocol of USCI controller applies to synchronous serial data communication and allows full duplex transfer. It supports both master and Slave operation mode with the 4-wire bi-direction interface. SPI mode of USCI controller performs a serial-to-parallel conversion on data received from a peripheral device, and a parallel-to-serial conversion on data transmitted to a peripheral device. The SPI mode is selected by FUNMODE (USPI_CTL[2:0]) = 0x1

This SPI protocol can operate as Master or Slave mode by setting the SLAVE (USPI_PROTCTL[0]) to communicate with the off-chip SPI Slave or master device. The application block diagrams in Master and Slave mode are shown below.

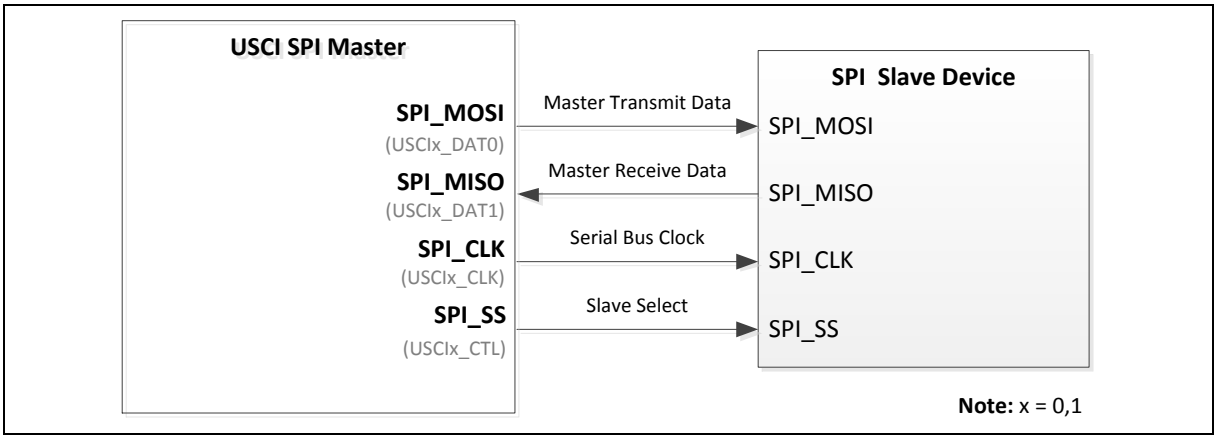


Figure 6.19-1 SPI Master Mode Application Block Diagram

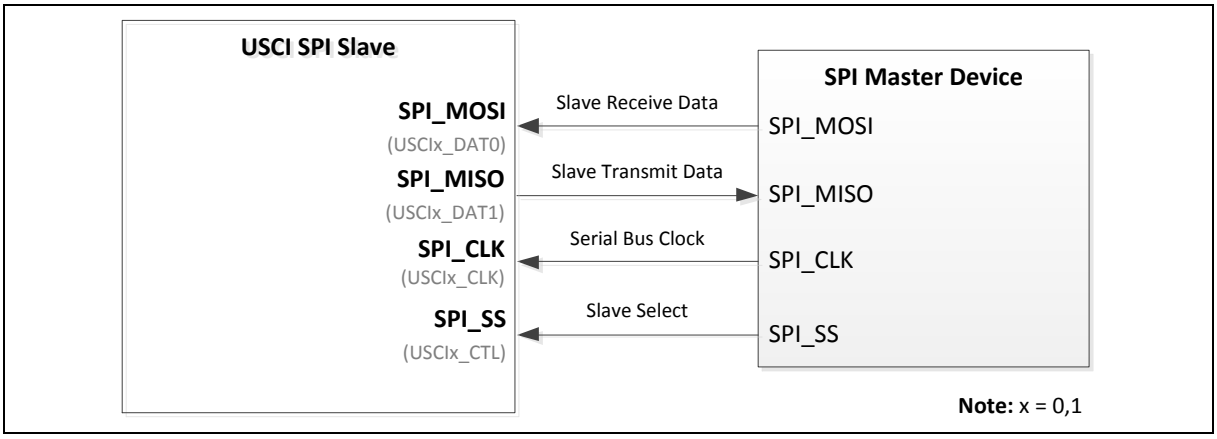


Figure 6.19-2 SPI Slave Mode Application Block Diagram

6.19.2 Features

- Supports Master or Slave mode operation (the maximum frequency -- Master = f_{PCLK} / 2, Slave < f_{PCLK} / 5)
- Configurable bit length of a transfer word from 4 to 16-bit
- Supports one transmit buffer and two receive buffers for data payload
- Supports MSB first or LSB first transfer sequence

- Supports Word Suspend function
- Supports PDMA transfer
- Supports 3-wire, no slave select signal, bi-direction interface
- Supports wake-up function by slave select signal in Slave mode
- Supports one data channel half-duplex transfer

6.19.3 Block Diagram

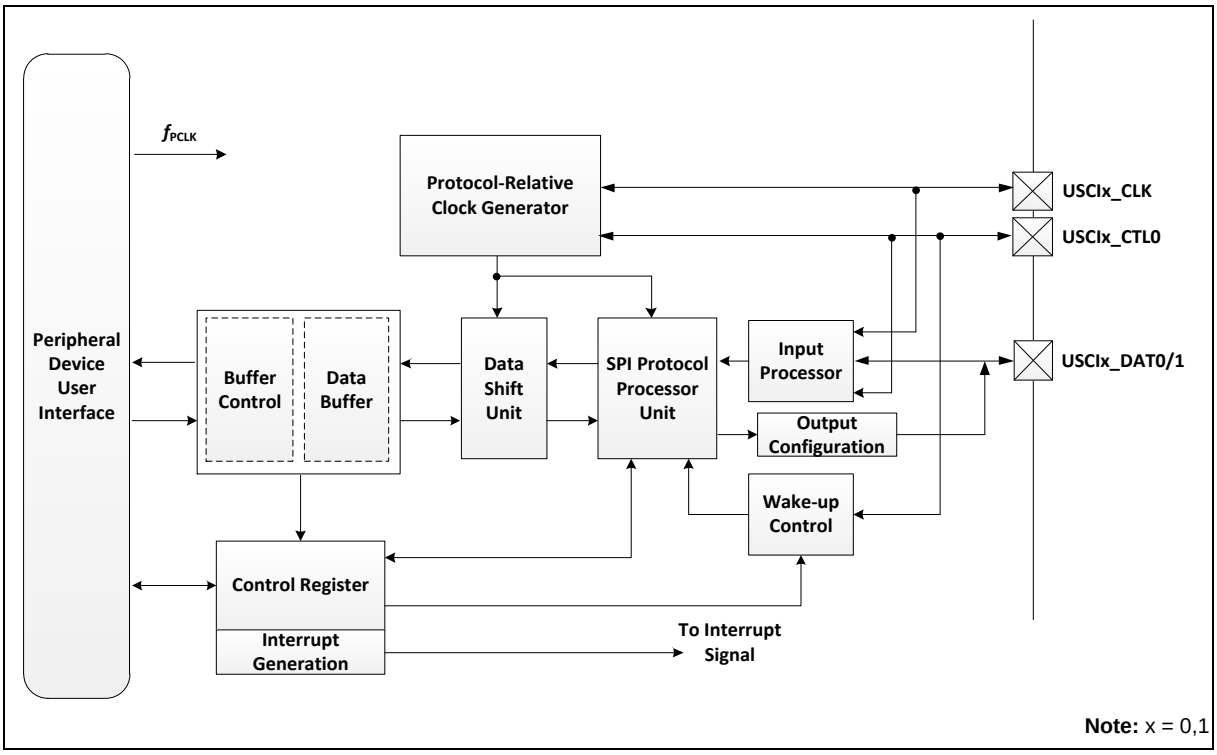


Figure 6.19-3 USCI SPI Mode Block Diagram

6.19.4 Basic Configuration

6.19.4.1 USCI0 SPI Basic Configurations

- Clock Source Configuration
  - Enable USCI0 peripheral clock in USCI0CKEN (CLK_APBCLK1[8]).
  - Enable USCI0_SPI functi on USPI_CTL[2:0] register, USPI_CTL[2:0]=3'b001
- Reset Configuration
  - Reset USCI0 controller in USCI0RST (SYS_IPRST2[8]).

6.19.4.2 USCI1 SPI Basic Configurations

- Clock Source Configuration
  - Enable USCI1 peripheral clock in USCI1CKEN (CLK_APBCLK1[9]).
  - Enable USCI1_SPI functi on USPI_CTL[2:0] register, USPI_CTL[2:0]=3'b001
- Reset Configuration
  - Reset USCI1 controller in USCI1RST (SYS_IPRST2[9]).

6.19.5 Functional Description

6.19.5.1 USCI Common Function Description

Please refer to section **6.17.4** for detailed information.

6.19.5.2 Signal Description

A device operating in Master mode controls the start and end of a data transfer, as well as the generation of the SPI bus clock and slave select signal. The slave select signal indicates the start and the end of a data transfer, and the master device can use it to enable the transmitting or receiving operations of Slave device. Slave device receives the SPI bus clock and optionally a slave select signal for data transaction. The signals for SPI communication are shown below.

SPI Mode	Receive Data	Transmit Data	Serial Bus Clock	Slave Select
Full-duplex SPI Master	SPI_MISO (USCIx_DAT1)	SPI_MOSI (USCIx_DAT0)	SPI_CLK (USCIx_CLK)	SPI_SS (USCIx_CTL0)
Full-duplex SPI Slave	SPI_MOSI (USCIx_DAT0)	SPI_MISO (USCIx_DAT1)	SPI_CLK (USCIx_CLK)	SPI_SS (USCIx_CTL0)
Half-duplex SPI Master/Slave	SPI_MOSI (USCIx_DAT0)	SPI_MOSI (USCIx_DAT0)	SPI_CLK (USCIx_CLK)	SPI_SS (USCIx_CTL0)

SPI Communication Signals

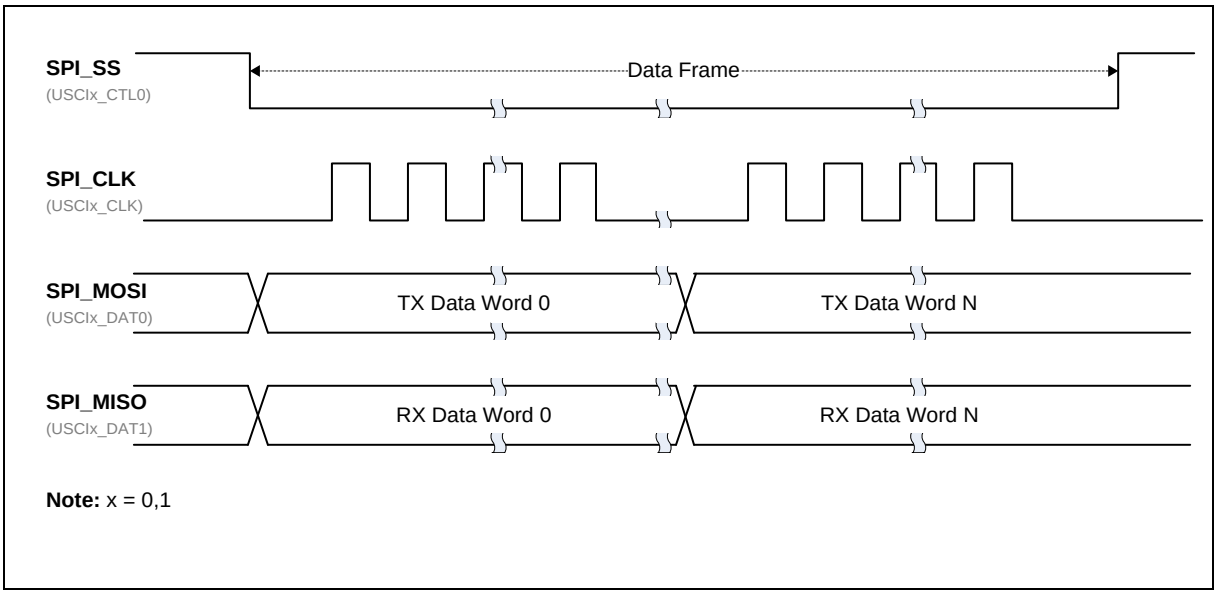


Figure 6.19-44-Wire Full-Duplex SPI Communication Signals (Master Mode)

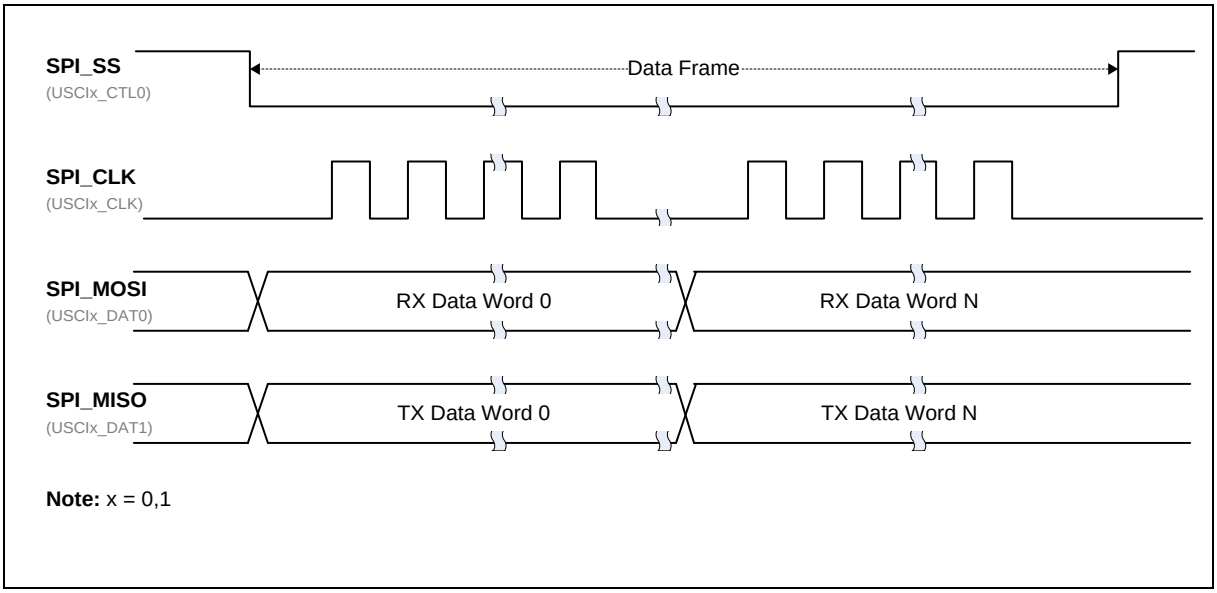


Figure 6.19-54-Wire Full-Duplex SPI Communication Signals (Slave Mode)

6.19.5.3 Serial Bus Clock Configuration

The USCI controller needs the peripheral clock to drive the USCI logic unit to perform the data transfer. The peripheral clock frequency is equal to PCLK frequency.

In Master mode, the frequency of the SPI bus clock is determined by protocol-relative clock generator. In general, the SPI bus clock is denoted as SPI clock. The frequency of SPI clock is half of f_{SAMP_CLK}, which can be selected by SPCLKSEL (USPI_BRGEN[3:2]). Refer to **6.17.4** for details of protocol-relative clock generator.

In Slave mode, the SPI bus clock is provided by an off-chip Master device. The peripheral clock frequency, f_{PCLK}, of SPI Slave device must be 5-times faster than the serial bus clock rate of the SPI Master device connected together (i.e. the clock rate of serial bus clock < 1/5 peripheral clock f_{PCLK} in Slave mode).

In SPI protocol, SCLKMODE (USPI_PROTCTL[7:6]) defines not only the idle state of serial bus clock but also the serial clock edge used for transmit and receive data. Both Master and Slave devices on the same communication bus should have the same SCLKMODE configuration. The four kinds of serial bus clock configuration are shown below.

SCLKMODE [1:0]	SPI Clock Idle State	Transmit Timing	Receive Timing
0x0	Low	Falling edge	Rising edge
0x1	Low	Rising edge	Falling edge
0x2	High	Rising edge	Falling edge
0x3	High	Falling edge	Rising edge

Table 6.19-1 Serial Bus Clock Configuration

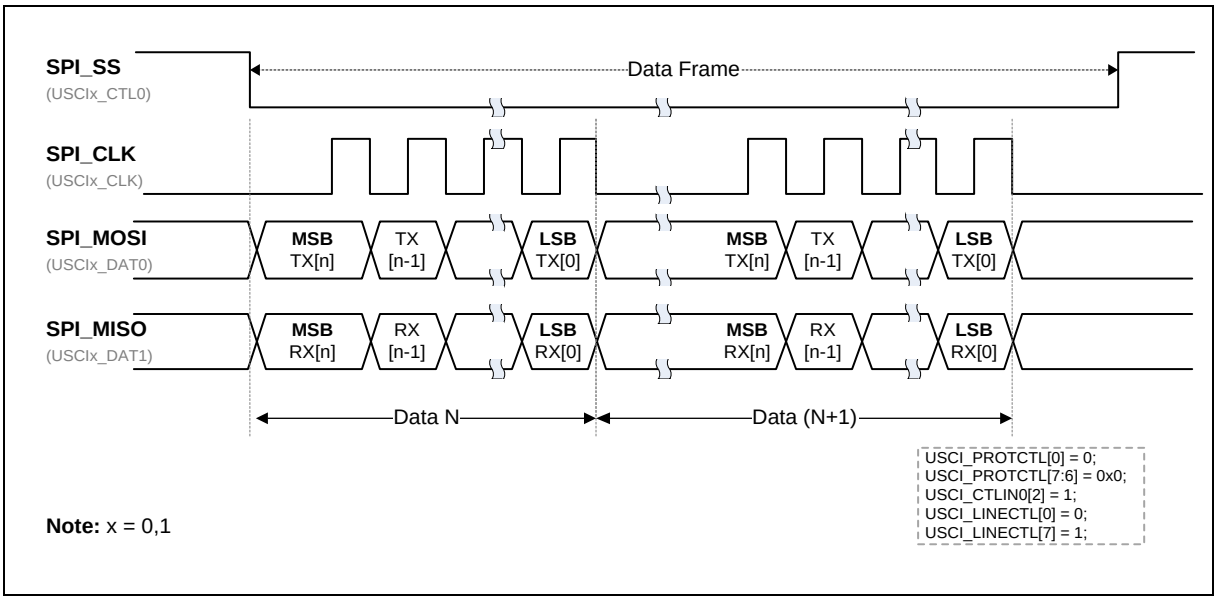


Figure 6.19-6 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x0)

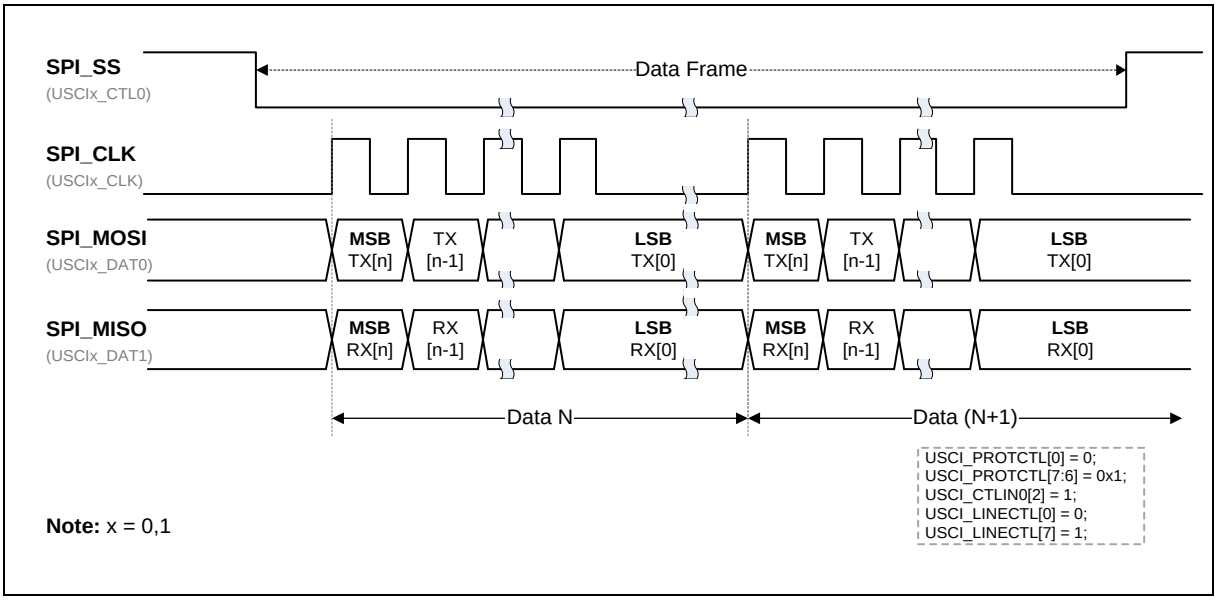


Figure 6.19-7 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x1)

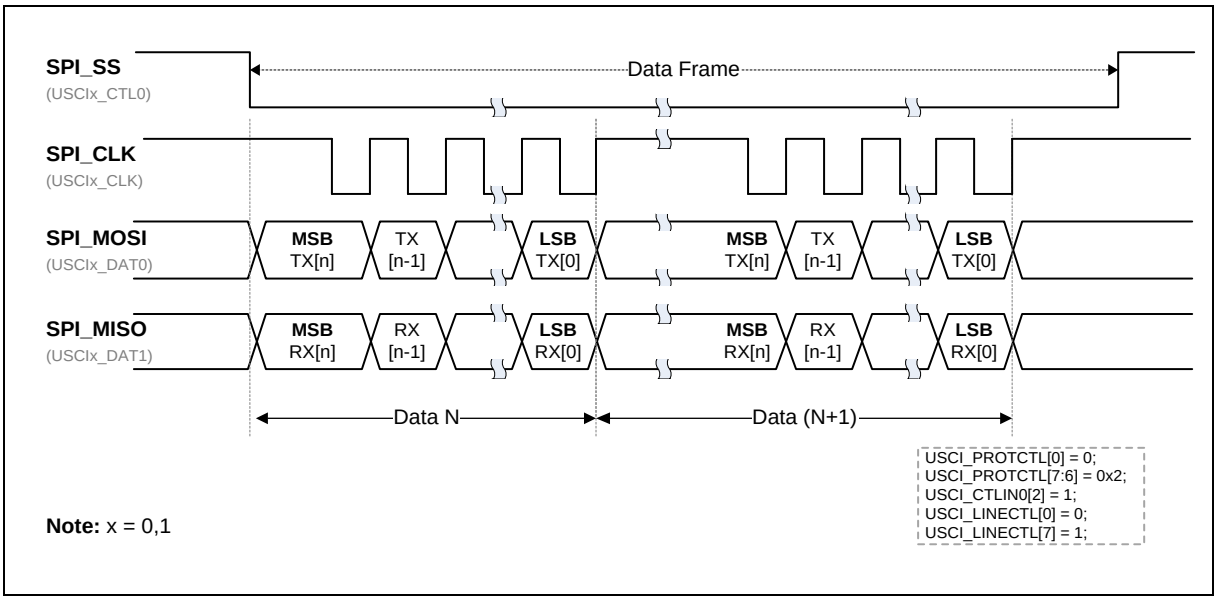


Figure 6.19-8 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x2)

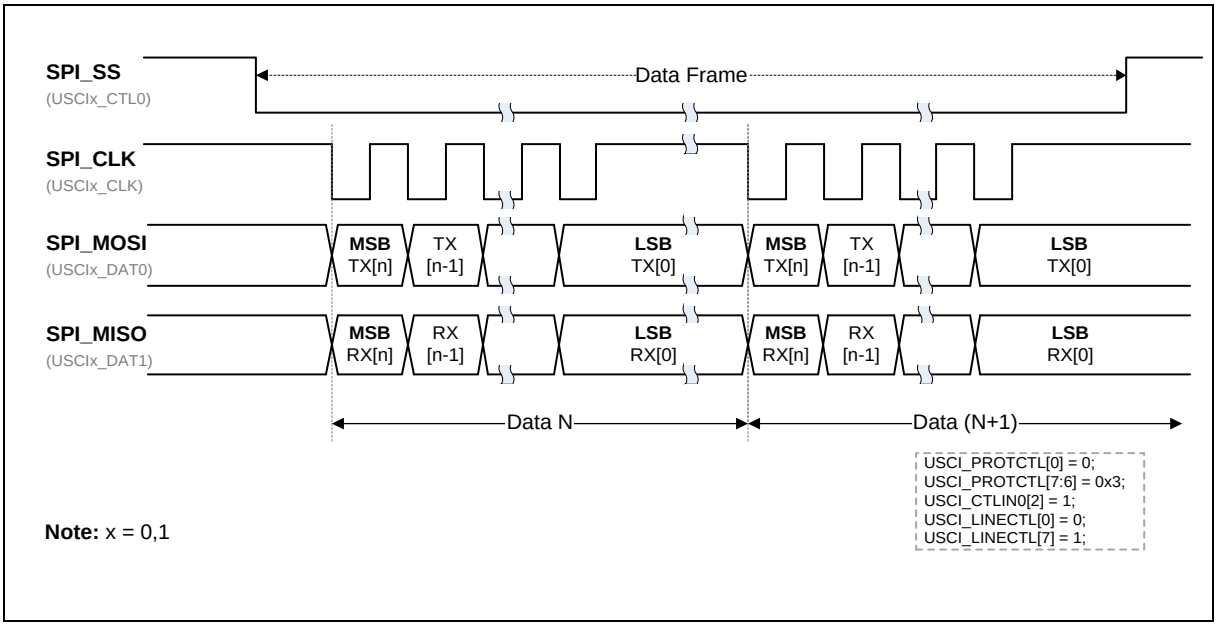


Figure 6.19-9 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x3)

6.19.5.4 Slave Select Signal

The slave selection signal of SPI protocol is active high by default. In SPI Master mode, the USCI controller can drive the control signal to off-chip SPI Slave device through slave select pin SPI_SS (USCIx_CTL0). In SPI Slave mode, the received slave select signal can be inverted by ININV (USPI_CTLIN0[2]).

If the slave select signal of external SPI Master device is low active, the ININV (USPI_CTLIN0[2]) setting of slave device should be set to 1 for the inversion of input control signal. If USCI operates as SPI Master mode, the output slave select inversion CTLOINV (USPI_LINECTL[7]) is also needed to

set as 1 for the external SPI Slave device whose slave select signal is active low.

The duration between the slave select active edge and the first SPI clock input edge shall over 2 USCI peripheral clock cycles.

The input slave select signal of SPI Slave has to be keep inactive for at least 2 USCI peripheral clock cycles between two consecutive frames in order to correctly detect the end of a frame.

6.19.5.5 Transmit and Receive Data

The bit length of a transmit/receive data word in SPI protocol of USCI controller is defined in DWIDTH (USPI_LINECTL[11:8]), and it can be configured up to 16-bit length for transmitting and receiving data in SPI communication.

The LSB bit (USPI_LINECTL[0]) defines the order of transfer data bit. If the LSB bit is set to 1, the transmission data sequence is LSB first. If the LSB bit is cleared to 0, the transmission data sequence is MSB first.

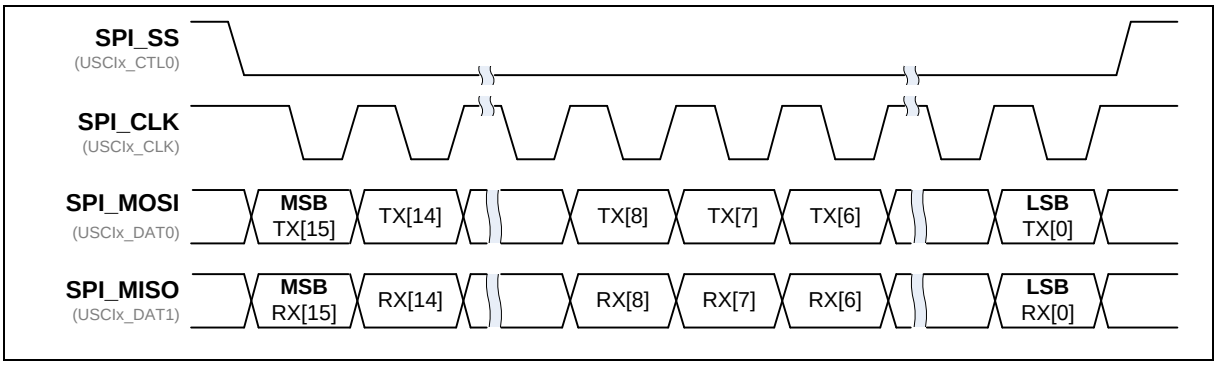


Figure 6.19-1016-bit Data Length in One Word Transaction with MSB First Format

6.19.5.6 Word Suspend

SUSPITV (USPI_PROTCTL[11:8]) 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 (USPI_PROTCTL[11:8]) is 0x3 (3.5 SPI clock cycles).

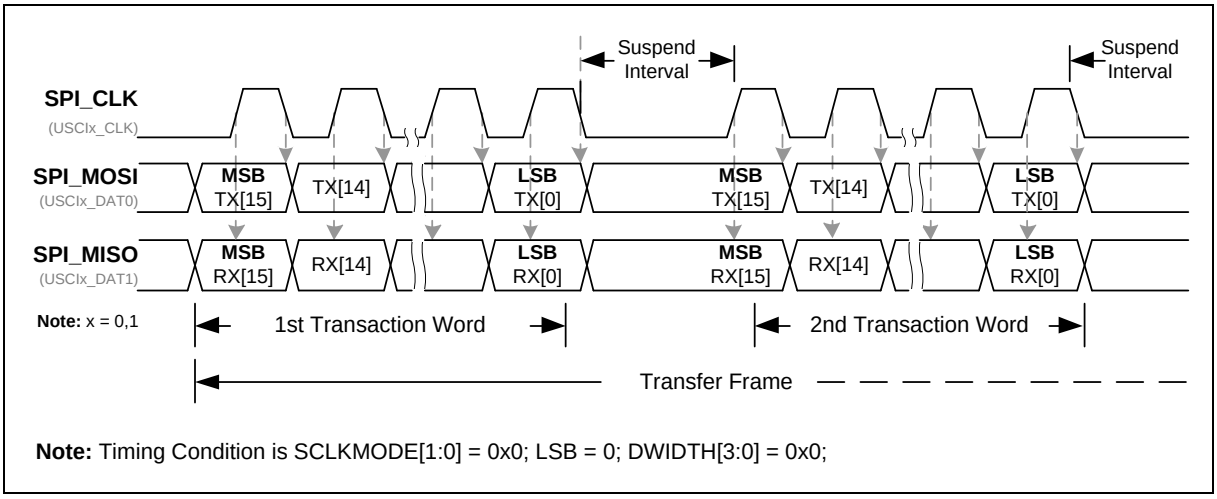


Figure 6.19-11 Word Suspend Interval between Two Transaction Words

6.19.5.7 Automatic Slave Select Function

AUTOSS (USPI_PROTCTL[3]) is used for SPI Master mode to enable the automatic slave select function. If the bit AUTOSS (USPI_PROTCTL[3]) is set, the slave select signal will be generated automatically and the setting value of SS (USPI_PROTCTL[2]) will not affect the output slave select (through USCIx_CTL0 line). This means that the slave select signal will be asserted by the USCI controller when the SPI data transfer is started by writing to the transmit buffer. And, it will be de-asserted after either all transaction is finished or one word transaction done if the value of SUSPITV (USPI_PROTCTL[11:8]) is equal to or great than 3.

If the AUTOSS bit (USPI_PROTCTL[3]) is cleared, the slave select on USCIx_CTL0 pin will be asserted/de-asserted by setting/clearing the SS (USPI_PROTCTL[2]). The internal slave select signal is active high and the CTLOINV (USPI_LINECTL[7]) can be used for the inversion of the slave select signal.

In SPI Master mode, if the value of SUSPITV (USPI_PROTCTL[11:8]) is less than 3 and the AUTOSS (USPI_PROTCTL[3]) is set as 1, the slave select signal will be kept at active state between two successive word transactions.

In SPI Slave mode, to recognize the inactive state of the slave select signal, the inactive period of the received slave select signal must be larger than 2 peripheral clock cycles between two successive transactions.

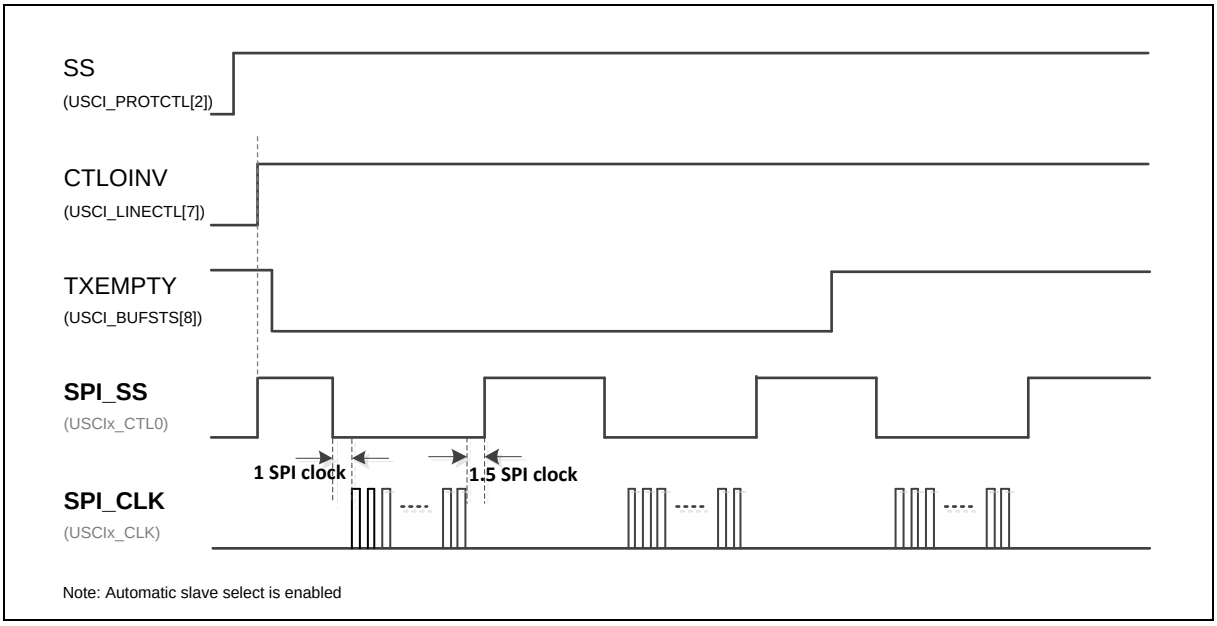


Figure 6.19-12 Auto Slave Select (SUSPITV ≧ 0x3)

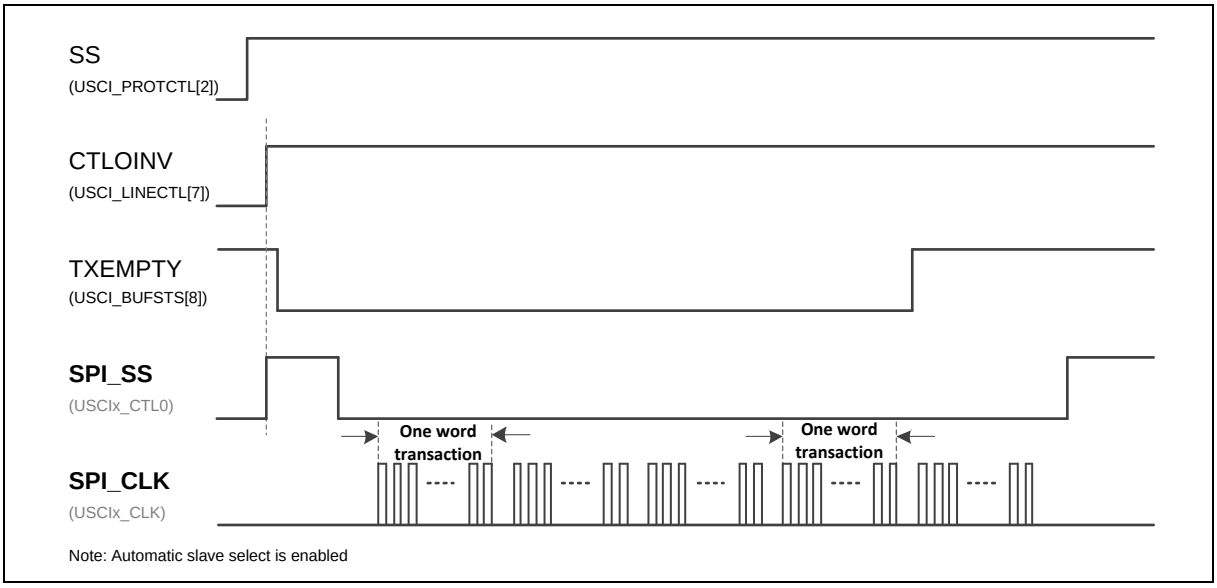


Figure 6.19-13 Auto Slave Select (SUSPITV < 0x3)

6.19.5.8 Slave 3-wire Mode

When the SLV3WIRE (USPI_PROTCTL[1]) is set by software to enable the Slave 3-wire mode, the USCI SPI communication can work with no slave select signal in Slave mode. The SLV3WIRE (USPI_PROTCTL[1]) only takes effect in SPI Slave mode. Only three pins, SPI_CLK (through USCIx_CLK line), SPI_MOSI (through USCIx_DAT0 line), and SPI_MISO (through USCIx_DAT1 line), are required to communicate with a SPI Master. When the SLV3WIRE (USPI_PROTCTL[1]) is set to 1, the SPI Slave will be ready to transmit/receive data after the SPI protocol is enabled by setting FUNMODE(USPI_CTL [2:0]) to 0x1.

6.19.5.9 Data Transfer Mode

The USCI controller supports full-duplex SPI transfer and one data channel half-duplex SPI transfer.

- Full-duplex SPI transfer

  In full-duplex SPI transfer, there are two data pins. One is used for transmitting data and the other is used for receiving data. Thus, data transmission and data reception can be performed simultaneously.

  SCLKMODE (USPI_PROTCTL[7:6]) defines the transition timing of the data shift output signal on USCIx_DAT0 pin. The transition may happen at the corresponding edge of SPI bus clock or active edge of slave select signal. The level of the last data bit of a data word is held on USCIx_DAT0 pin until the next data word begins with the next corresponding edge of the serial bus clock.

- One data channel half-duplex SPI transfer

  In one data channel half-duplex SPI transfer, there is only one data pin for data transfer. Thus, the data transmission and data reception are at different time interval. The data shift direction is determined by PORTDIR (USPI_TXDAT[16]). Refer to the register description for more detail information.

  The function of one data channel half-duplex SPI transfer is similar to the full-duplex SPI protocol. All the transfer data timing is the same as the full-duplex SPI transfer.

  Figure 6.19-14 shows the one output data channel and one input data channel half-duplex transfer diagrams with the external device.

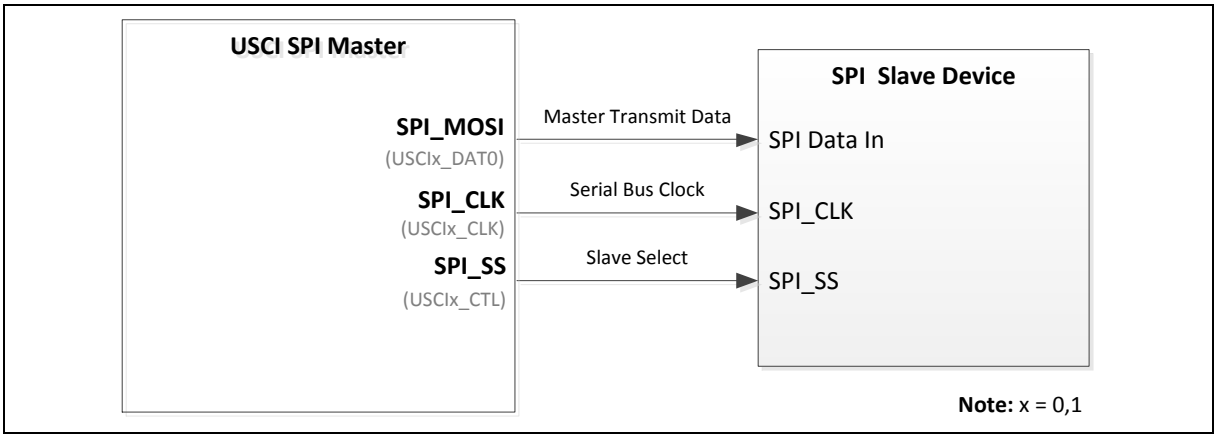


Figure 6.19-14 One Output Data Channel Half-duplex (SPI Master Mode)

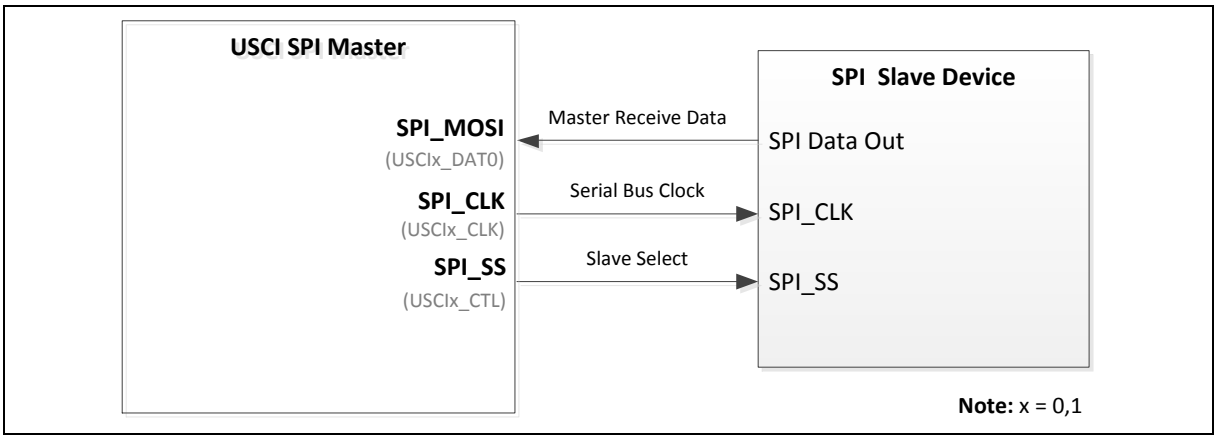


Figure 6.19-15 One Input Data Channel Half-duplex (SPI Master Mode)

The one data channel half-duplex transfer mode can be configured by TSMSEL[2:0] (USPI_PROTCTL[14:12]) and PORTDIR (USPI_TXDAT[16]) settings. When TSMSEL (USPI_PROTCTL[14:12]) is set to 0x4, one data channel half-duplex transfer mode is selected. The PORTDIR (USPI_TXDAT[16]) is used to define the direction of the corresponding transmit data. When the PORTDIR bit is set to 0, the USCI controller will send the corresponding data to external SPI device. When the PORTDIR bit is set to 1, the controller will read the corresponding data from the external SPI device.

For example, in one data channel half-duplex transfer mode with PORTDIR=0, USCI SPI transmits data through USCIx_DAT0 pin; if PORTDIR=1, USCI SPI receives data through USCIx_DAT0 pin.

6.19.5.10 Interrupt

Data Transfer Interrupts

- Transmit start interrupt

  The interrupt event TXSTIF (USPI_PROTSTS[1]) is set after the start of the first data bit of a transmit data word. It can be cleared only by writing 1 to it.

- Transmit end interrupt

  The interrupt event TXENDIF (USPI_PROTSTS[2]) is set after the start of the last data bit of the last transmit data which has been stored in transmit buffer. It can be cleared only by writing 1 to it.

- Receive start interrupt

The interrupt event RXSTIF (USPI_PROTSTS[3]) is set after the start of the first data bit of a receive data word. It can be cleared only by writing 1 to it.

- Receive end interrupt

  The interrupt event RXENDIF (USPI_PROTSTS[4]) is set after the start of the last data bit of a receive data word. It can be cleared only by writing 1 to it.

Protocol-Related Interrupts

- SPI slave select interrupt

  In SPI Slave mode, there are slave select active and in-active interrupt flags, SSACTIF (USPI_PROTSTS[9]) and SSINAIF (USPI_PROTSTS[8]), will be set to 1 when SLAVE (USPI_PROTCTL [0]) is set to 1 and Slave senses the slave select signal active or inactive. The SPI controller will issue an interrupt if SSINAIEN (USPI_PROTIEN[0]) or SSACTIEN (USPI_PROTIEN[1]), are set to 1. Because the internal slave select signal in SPI function is active high, the ININV (USPI_CTLIN0[2]) can be used for inverting the slave select signal comes from an active low device.

- Slave time-out interrupt

  In SPI Slave mode, there is Slave time-out function for user to know that there is no serial clock input during the period of one word transaction. The Slave time-out function uses the timing measurement counter for the calculation of Slave time-out period which is defined by SLVTOCNT (USPI_PROTCTL[25:16]). TMCNTSRC (USPI_BRGEN[5]) can be used for clock frequency selection of timing measurement counter to calculate the Slave time-out period.

  When the timing measurement counter is enabled by TMCNTEN (USPI_BRGEN[4]) and the setting value of SLVTOCNT (USPI_PROTCTL[25:16]) is not 0 in SPI Slave mode, the timing measurement counter will start counting after the first input serial clock of each received word data. This counter will be reset while receiving the following input serial clock and then keep counting. Finally, the timing measurement counter will be cleared and stopped after the finish of the current word transaction. If the value of the time-out counter is equal to or greater than the value of SLVTOCNT (USPI_PROTCTL[25:16]) before one word transaction is done, the Slave time-out interrupt event occurs and the SLVTOIF (USPI_PROTSTS[5]) will be set to 1.

Buffer-Related Interrupts

The buffer-related interrupts are available if there is transmit/receive buffer in USCI controller.

- Receive buffer overrun interrupt

  If there is receive buffer overrun event, RXOVIF (USPI_BUFSTS[3]) will be set as 1. It can be cleared by write 1 into it.

- Transmit buffer under-run interrupt

  If there is transmit buffer under-run event, TXUDRIF (USPI_BUFSTS[11]) will be set as 1. It can be cleared by write 1 into it.

6.19.5.11 Timing Diagram

The slave select signal of USCI SPI protocol is active high by default, and it can be inverted by CTLOINV (USPI_LINECTL[7]) setting.

The idle state of serial bus clock and the serial bus clock edge used for transmit/receive data can be configured by setting SCLKMODE (USPI_PROTCTL[7:6]). The bit length of a transaction word data is determined by DWIDTH (USPI_LINECTL[11:8]), and data bit transfer sequence is determined by LSB (USPI_LINECTL[0]). Four SPI timing diagrams for Master/Slave operations and the related settings are shown below.

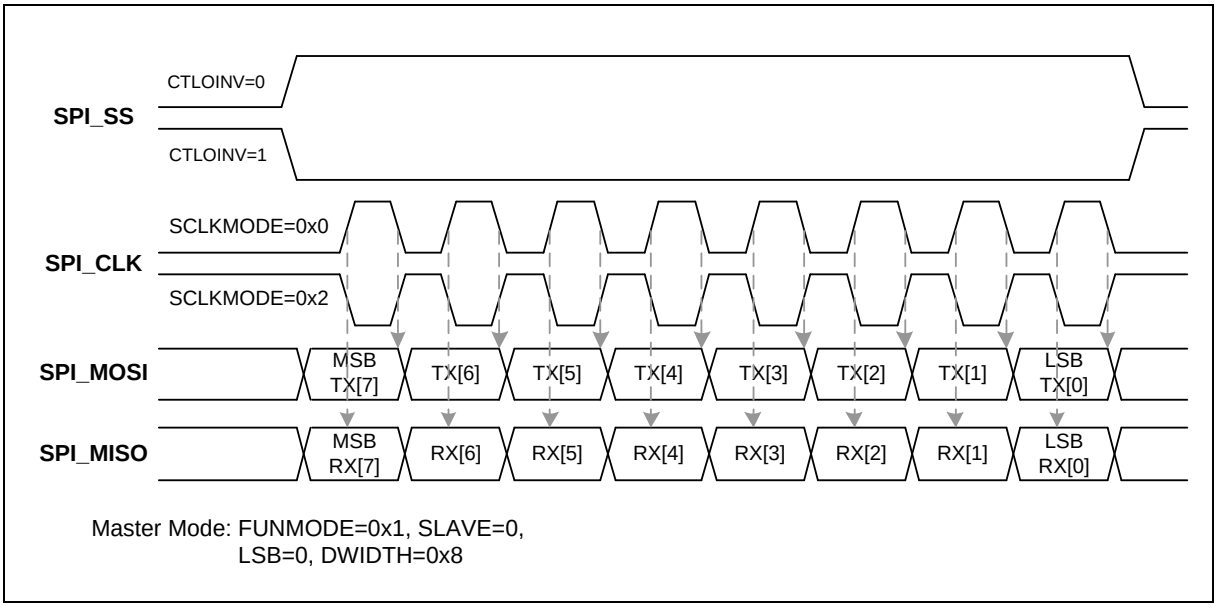


Figure 6.19-16 SPI Timing in Master Mode

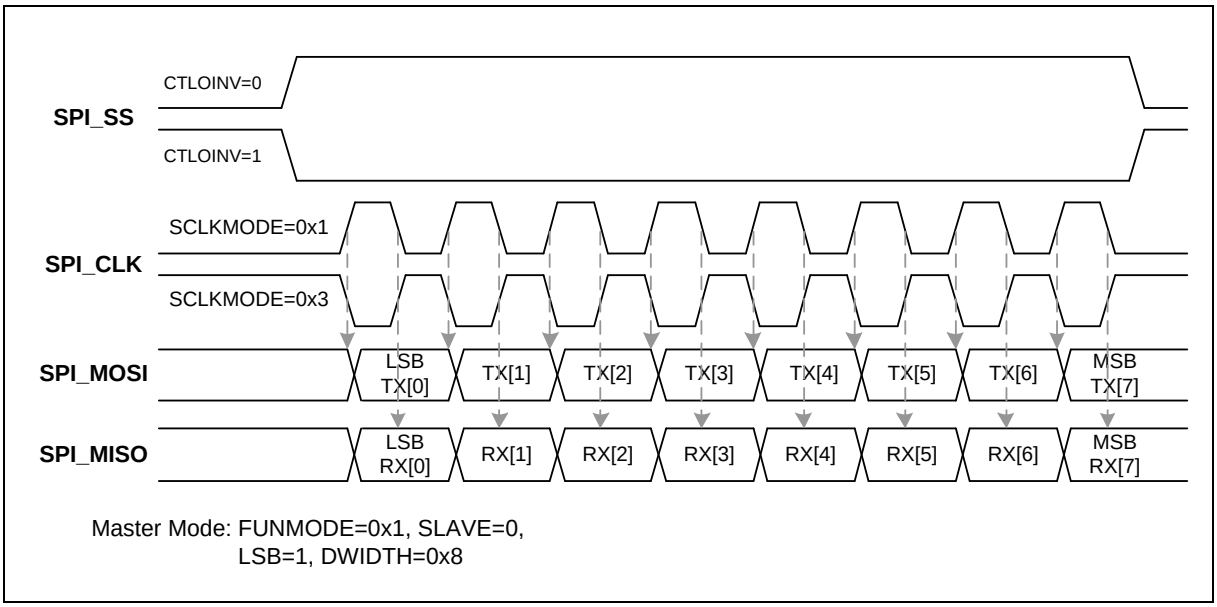


Figure 6.19-17 SPI Timing in Master Mode (Alternate Phase of Serial Bus Clock)

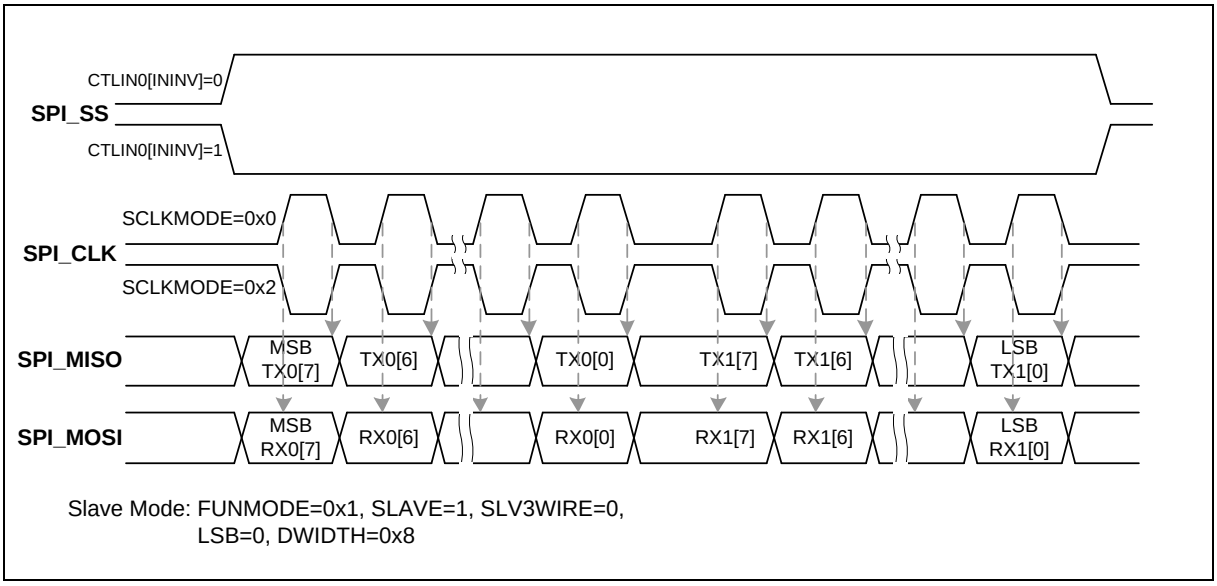


Figure 6.19-18 SPI Timing in Slave Mode

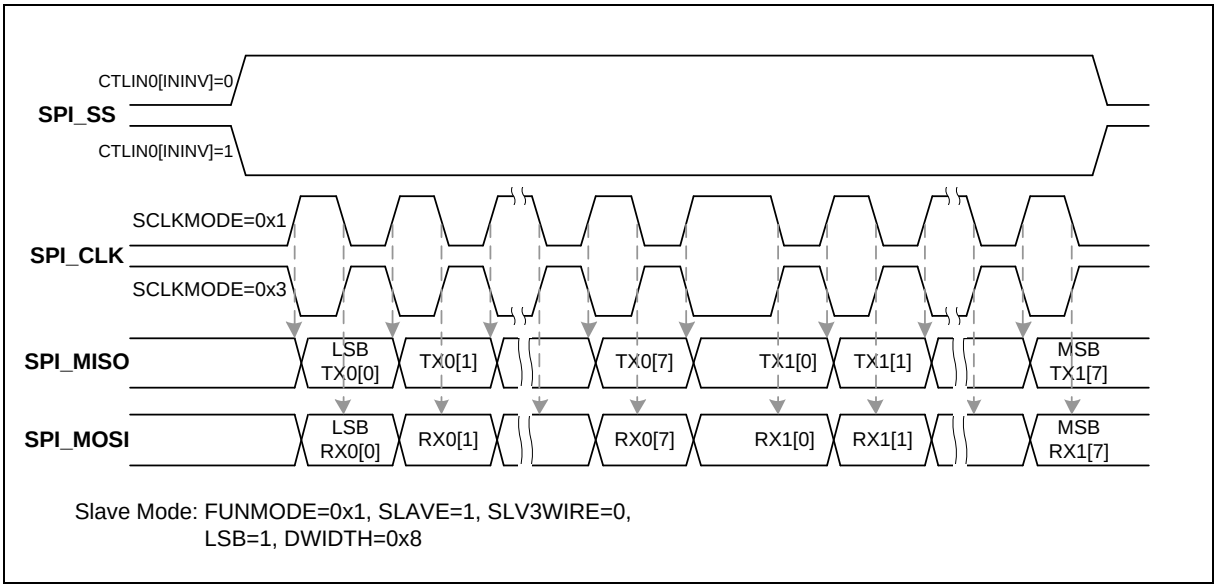


Figure 6.19-19 SPI Timing in Slave Mode (Alternate Phase of Serial Bus Clock)

6.19.5.12 Programming Flow

This section describes the programming flow for USCI SPI data transfer.

For Master mode:

1. Enable USCI peripheral clock by setting CLK_APBCLK1 register.
2. Configure user-specified pins as USCI function pins by setting corresponding multiple function control registers.
3. Set FUNMODE (USPI_CTL[2:0]) to 1 to select SPI mode.
4. Set USPI_BRGEN register to determine the SPI bus clock frequency.
5. According to the requirements of user's application, configure the settings as follows.
   - CTLOINV (USPI_LINECTL[7]): If the slave selection signal is active low, set this bit to 1;

otherwise, set it to 0.

- DWIDTH (USPI_LINECTL[11:8]): Data width setting.
- LSB (USPI_LINECTL[0]): LSB first or MSB first.
- TSMSEL (USPI_PROTCTL[14:12]): Full-duplex SPI transfer or one channel half-duplex SPI transfer.
- SCLKMODE (USPI_PROTCTL[7:6]): Determine the clock timing.
- AUTOSS (USPI_PROTCTL[3]): Enable automatic slave select function or not.
- SLAVE (USPI_PROTCTL[0]): Set to 0 for Master mode.
- Set PROTEN (USPI_PROTCTL[31]) to 1 to enable SPI protocol.

6. If automatic slave select function is disabled (AUTOSS=0), set SS (USPI_PROTCTL[2]) to 1 before data transfer; set SS to 0 to inactivate the slave selection signal by user's application.
7. Write USPI_TXDAT register to trigger SPI transfer. In half-duplex SPI transfer, the data pin direction is determined by PORTDIR (USPI_TXDAT[16]) setting.
8. User can get the received data by reading USPI_RXDAT register as long as RXEMPTY (USPI_BUFSTS[0]) is 0. The SPI data transfer can be triggered by writing USPI_TXDAT register as long as TXFULL (USPI_BUFSTS[9]) is 0.

For Slave mode:

1. Enable USCI peripheral clock by setting CLK_APBCLK1 register.
2. Configure user-specified pins as USCI function pins by setting corresponding multiple function control registers.
3. Set FUNMODE (USPI_CTL[2:0]) to 1 to select SPI mode.
4. According to the requirements of user's application, configure the settings as follows.
   - ININV (USPI_CTLIN0[2]): If the slave selection signal is active low, set this bit to 1; otherwise, set it to 0.
   - DWIDTH (USPI_LINECTL[11:8]): Data width setting.
   - LSB (USPI_LINECTL[0]): LSB first or MSB first.
   - TSMSEL (USPI_PROTCTL[14:12]): Full-duplex SPI transfer or one channel half-duplex SPI transfer.
   - SCLKMODE (USPI_PROTCTL[7:6]): Determine the clock timing.
   - SLAVE (USPI_PROTCTL[0]): Set to 1 for Slave mode.
5. Set PROTEN (USPI_PROTCTL[31]) to 1 to enable SPI protocol.
6. Write USPI_TXDAT register for transmission. In half-duplex SPI transfer, the data pin direction is determined by PORTDIR (USPI_TXDAT[16]) setting.
7. User can get the received data by reading USPI_RXDAT register as long as RXEMPTY (USPI_BUFSTS[0]) is 0. The next datum for transmission can be written to USPI_TXDAT register as long as TXFULL (USPI_BUFSTS[9]) is 0.

6.19.5.13 Wake-up Function

The USCI Controller in SPI mode supports wake-up system function. The wake-up source in SPI protocol is the transition of input slave select signal.

6.19.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
USCI_SPI Base Address: **USPIn_BA = 0x400D_0000 + (0x1000 * n)** **n= 0,1**				
USPI_CTL	USPIn_BA+0x00	R/W	USCI Control Register	0x0000_0000
USPI_INTEN	USPIn_BA+0x04	R/W	USCI Interrupt Enable Register	0x0000_0000
USPI_BRGEN	USPIn_BA+0x08	R/W	USCI Baud Rate Generator Register	0x0000_3C00
USPI_DATIN0	USPIn_BA+0x10	R/W	USCI Input Data Signal Configuration Register 0	0x0000_0000
USPI_CTLIN0	USPIn_BA+0x20	R/W	USCI Input Control Signal Configuration Register 0	0x0000_0000
USPI_CLKIN	USPIn_BA+0x28	R/W	USCI Input Clock Signal Configuration Register	0x0000_0000
USPI_LINECTL	USPIn_BA+0x2C	R/W	USCI Line Control Register	0x0000_0000
USPI_TXDAT	USPIn_BA+0x30	W	USCI Transmit Data Register	0x0000_0000
USPI_RXDAT	USPIn_BA+0x34	R	USCI Receive Data Register	0x0000_0000
USPI_BUFCTL	USPIn_BA+0x38	R/W	USCI Transmit/Receive Buffer Control Register	0x0000_0000
USPI_BUFSTS	USPIn_BA+0x3C	R	USCI Transmit/Receive Buffer Status Register	0x0000_0101
USPI_PDMACTL	USPIn_BA+0x40	R/W	USCI PDMA Control Register	0x0000_0000
USPI_WKCTL	USPIn_BA+0x54	R/W	USCI Wake-up Control Register	0x0000_0000
USPI_WKSTS	USPIn_BA+0x58	R/W	USCI Wake-up Status Register	0x0000_0000
USPI_PROTCTL	USPIn_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0300
USPI_PROTIEN	USPIn_BA+0x60	R/W	USCI Protocol Interrupt Enable Register	0x0000_0000
USPI_PROTSTS	USPIn_BA+0x64	R/W	USCI Protocol Status Register	0x0000_0000

6.19.7 Register Description

USCI Control Register (USPI_CTL)

Register	Offset	R/W	Description	Reset Value
USPI_CTL	USPIn_BA+0x00	R/W	USCI Control Register	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**FUNMODE**	**Function Mode** This bit field selects the protocol for this USCI controller. Selecting a protocol that is not available or a reserved combination disables the USCI. When switching between two protocols, the USCI has to be disabled before selecting a new protocol. Simultaneously, the USCI will be reset when user write 000 to FUNMODE. 000 = The USCI is disabled. All protocol related state machines are set to idle state. 001 = The SPI protocol is selected. 010 = The UART protocol is selected. 100 = The I^2C protocol is selected. **Note:** Other bit combinations are reserved.

USCI Interrupt Enable Register (USPI_INTEN)

Register	Offset	R/W	Description	Reset Value
USPI_INTEN	USPIn_BA+0x04	R/W	USCI Interrupt Enable Register	0x0000_0000

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

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**RXENDIEN**	**Receive End Interrupt Enable Bit** This bit enables the interrupt generation in case of a receive finish event. 0 = The receive end interrupt Disabled. 1 = The receive end interrupt Enabled. **Note:** The receive finish event happens when hardware receives the last bit of RX data into shift data unit.
[3]	**RXSTIEN**	**Receive Start Interrupt Enable Bit** This bit enables the interrupt generation in case of a receive start event. 0 = The receive start interrupt Disabled. 1 = The receive start interrupt Enabled. **Note:** For SPI master mode, the receive start event happens when SPI master sends slave select active and spi clock to the external SPI slave. For SPI slave mode, the receive start event happens when slave select of SPI slave is active and spi clock of SPI slave is inputed from the external SPI master.
[2]	**TXENDIEN**	**Transmit End Interrupt Enable Bit** This bit enables the interrupt generation in case of a transmit finish event. 0 = The transmit finish interrupt Disabled. 1 = The transmit finish interrupt Enabled. **Note:** The transmit finish event happens when hardware sends the last bit of TX data from shift data unit.
[1]	**TXSTIEN**	**Transmit Start Interrupt Enable Bit** This bit enables the interrupt generation in case of a transmit start event. 0 = The transmit start interrupt Disabled. 1 = The transmit start interrupt Enabled. **Note:** The transmit start event happens when hardware starts to move TX data from data buffer to shift data unit.
[0]	**Reserved**	Reserved.

USCI Baud Rate Generator Register (USPI_BRGEN)

Register	Offset	R/W	Description	Reset Value
USPI_BRGEN	USPIn_BA+0x08	R/W	USCI Baud Rate Generator Register	0x0000_3C00

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

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25:16]	**CLKDIV**	**Clock Divider** This bit field defines the ratio between the protocol clock frequency f_{PROT_CLK} and the clock divider frequency f_{DIV_CLK} (f_{DIV_CLK} = f_{PROT_CLK} / (CLKDIV+1)).
[15:6]	**Reserved**	Reserved.
[5]	**TMCNTSRC**	**Time Measurement Counter Clock Source Selection** 0 = Time measurement counter with f_{PROT_CLK}. 1 = Time measurement counter with f_{DIV_CLK}.
[4]	**TMCNTEN**	**Time Measurement Counter Enable Bit** This bit enables the 10-bit timing measurement counter. 0 = Time measurement counter Disabled. 1 = Time measurement counter Enabled.
[3:2]	**SPCLKSEL**	**Sample Clock Source Selection** This bit field used for the clock source selection of sample clock (f_{SAMP_CLK}) for the protocol processor. 00 = f_{DIV_CLK}. 01 = f_{PROT_CLK}. 10 = f_{SCLK}. 11 = f_{REF_CLK}.
[1]	**PTCLKSEL**	**Protocol Clock Source Selection** This bit selects the source of protocol clock (f_{PROT_CLK}). 0 = Reference clock f_{REF_CLK}. 1 = f_{REF_CLK2} (its frequency is half of f_{REF_CLK}).
[0]	**RCLKSEL**	**Reference Clock Source Selection** This bit selects the source of reference clock (f_{REF_CLK}). 0 = Peripheral device clock f_{PCLK}.

		1 = Reserved.

USCI Input Data Signal Configuration (USPI_DATIN0)

Register	Offset	R/W	Description	Reset Value
USPI_DATIN0	USPIn_BA+0x10	R/W	USCI Input Data Signal Configuration Register 0	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**ININV**	**Input Signal Inverse Selection** This bit defines the inverter enable of the input asynchronous signal. 0 = The un-synchronized input signal will not be inverted. 1 = The un-synchronized input signal will be inverted. **Note:** In SPI protocol, it is suggested this bit should be set as 0.
[1]	**Reserved**	Reserved.
[0]	**SYNCSEL**	**Input Signal Synchronization Selection** This bit selects if the un-synchronized input signal (with optionally inverted) or the synchronized (and optionally filtered) signal, which is synchronized with PCLK, can be used as input for the data shift unit. 0 = The un-synchronized signal can be taken as input for the data shift unit. 1 = The synchronized signal can be taken as input for the data shift unit. **Note:** In SPI protocol, it is suggested this bit should be set as 0.

USCI Input Control Signal Configuration (USPI_CTLIN0)

Register	Offset	R/W	Description	Reset Value
USPI_CTLIN0	USPIn_BA+0x20	R/W	USCI Input Control Signal Configuration Register 0	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**ININV**	**Input Signal Inverse Selection** This bit defines the inverter enable of the input asynchronous signal. 0 = The un-synchronized input signal will not be inverted. 1 = The un-synchronized input signal will be inverted.
[1]	**Reserved**	Reserved.
[0]	**SYNCSEL**	**Input Synchronization Signal Selection** This bit selects if the un-synchronized input signal (with optionally inverted) or the synchronized (and optionally filtered) signal, which is synchronized with PCLK, can be used as input for the data shift unit. 0 = The un-synchronized signal can be taken as input for the data shift unit. 1 = The synchronized signal can be taken as input for the data shift unit. **Note:** In SPI protocol, it is suggested this bit should be set as 0.

USCI Input Clock Signal Configuration (USPI_CLKIN)

Register	Offset	R/W	Description	Reset Value
USPI_CLKIN	USPIn_BA+0x28	R/W	USCI Input Clock Signal Configuration Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**SYNCSEL**	**Input Synchronization Signal Selection** This bit selects if the un-synchronized input signal or the synchronized (and optionally filtered) signal, which is synchronized with PCLK, can be used as input for the data shift unit. 0 = The un-synchronized signal can be taken as input for the data shift unit. 1 = The synchronized signal can be taken as input for the data shift unit. **Note:** In SPI protocol, it is suggested this bit should be set as 0.

USCI Line Control Register (USPI_LINECTL)

Register	Offset	R/W	Description	Reset Value
USPI_LINECTL	USPIn_BA+0x2C	R/W	USCI Line Control Register	0x0000_0000

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

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:8]	**DWIDTH**	**Word Length of Transmission** This bit field defines the data word length (amount of bits) for reception and transmission. The data word is always right-aligned in the data buffer. USCI support word length from 4 to 16 bits. 0x0: The data word contains 16 bits located at bit positions [15:0]. 0x1: Reserved. 0x2: Reserved. 0x3: Reserved. 0x4: The data word contains 4 bits located at bit positions [3:0]. 0x5: The data word contains 5 bits located at bit positions [4:0]. ... 0xF: The data word contains 15 bits located at bit positions [14:0].
[7]	**CTLOINV**	**Control Signal Output Inverse Selection** This bit defines the relation between the internal control signal and the output control signal. 0 = No effect. 1 = The control signal will be inverted before its output. **Note:** The control signal has different definitions in different protocol. In SPI protocol, the control signal means slave select signal.
[6]	**Reserved**	Reserved.
[5]	**DATOINV**	**Data Output Inverse Selection** This bit defines the relation between the internal shift data value and the output data signal of USCIx_DAT0/1 pins. 0 = Data output values of USCIx_DAT0/1 pins are not inverted. 1 = Data output values of USCIx_DAT0/1 pins are inverted.
[4:1]	**Reserved**	Reserved.
[0]	**LSB**	**LSB First Transmission Selection**

		0 = The MSB, which bit of transmit/receive data buffer depends on the setting of DWIDTH, is transmitted/received first. 1 = The LSB, the bit 0 of data buffer, will be transmitted/received first.

USCI Transmit Data Register (USPI_TXDAT)

Register	Offset	R/W	Description	Reset Value
USPI_TXDAT	USPIn_BA+0x30	W	USCI Transmit Data Register	0x0000_0000

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

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**PORTDIR**	**Port Direction Control** This bit field is only available while USCI operates in SPI protocol (FUNMODE = 0x1) with half-duplex transfer. It is used to define the direction of the data port pin. When software writes USPI_TXDAT register, the transmit data and its port direction are settled simultaneously. 0 = The data pin is configured as output mode. 1 = The data pin is configured as input mode.
[15:0]	**TXDAT**	**Transmit Data** Software can use this bit field to write 16-bit transmit data for transmission. In order to avoid overwriting the transmit data, user have to check TXEMPTY (USPI_BUFSTS[8]) status before writing transmit data into this bit field.

USCI Receive Data Register (USPI_RXDAT)

Register	Offset	R/W	Description	Reset Value
USPI_RXDAT	USPIn_BA+0x34	R	USCI Receive Data Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**RXDAT**	**Received Data** This bit field monitors the received data which stored in receive data buffer.

USCI Transmitter/Receive Buffer Control Register (USPI_BUFCTL)

Register	Offset	R/W	Description	Reset Value
USPI_BUFCTL	USPIn_BA+0x38	R/W	USCI Transmit/Receive Buffer Control Register	0x0000_0000

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

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**RXRST**	**Receive Reset** 0 = No effect. 1 = Reset the receive-related counters, state machine, and the content of receive shift register and data buffer. **Note:** It is cleared automatically after one PCLK cycle.
[16]	**TXRST**	**Transmit Reset** 0 = No effect. 1 = Reset the transmit-related counters, state machine, and the content of transmit shift register and data buffer**.** **Note 1:** It is cleared automatically after one PCLK cycle. **Note 2:** Write 1 to this bit will set the output data pin to zero if USPI_PROTCTL[28]=0.
[15]	**RXCLR**	**Clear Receive Buffer** 0 = No effect. 1 = The receive buffer is cleared. Should only be used while the buffer is not taking part in data traffic. **Note:** It is cleared automatically after one PCLK cycle.
[14]	**RXOVIEN**	**Receive Buffer Overrun Interrupt Enable Bit** 0 = Receive overrun interrupt Disabled. 1 = Receive overrun interrupt Enabled.
[13:8]	**Reserved**	Reserved.
[7]	**TXCLR**	**Clear Transmit Buffer** 0 = No effect. 1 = The transmit buffer is cleared. Should only be used while the buffer is not taking part in data traffic. **Note:** It is cleared automatically after one PCLK cycle.
[6]	**TXUDRIEN**	**Slave Transmit Under Run Interrupt Enable Bit**

		0 = Transmit under-run interrupt Disabled. 1 = Transmit under-run interrupt Enabled.
[5:0]	**Reserved**	Reserved.

USCI Transmit/Receive Buffer Status Register (USPI_BUFSTS)

Register	Offset	R/W	Description	Reset Value
USPI_BUFSTS	USPIn_BA+0x3C	R	USCI Transmit/Receive Buffer Status Register	0x0000_0101

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

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11]	**TXUDRIF**	**Transmit Buffer Under-run Interrupt Status** This bit indicates that a transmit buffer under-run event has been detected. If enabled by TXUDRIEN (USPI_BUFCTL[6]), the corresponding interrupt request is activated. It is cleared by software writes 1 to this bit 0 = A transmit buffer under-run event has not been detected. 1 = A transmit buffer under-run event has been detected.
[10]	**Reserved**	Reserved.
[9]	**TXFULL**	**Transmit Buffer Full Indicator** 0 = Transmit buffer is not full. 1 = Transmit buffer is full.
[8]	**TXEMPTY**	**Transmit Buffer Empty Indicator** 0 = Transmit buffer is not empty. 1 = Transmit buffer is empty and available for the next transmission datum.
[7:4]	**Reserved**	Reserved.
[3]	**RXOVIF**	**Receive Buffer Over-run Interrupt Status** This bit indicates that a receive buffer overrun event has been detected. If RXOVIEN (USPI_BUFCTL[14]) is enabled, the corresponding interrupt request is activated. It is cleared by software writes 1 to this bit. 0 = A receive buffer overrun event has not been detected. 1 = A receive buffer overrun event has been detected.
[2]	**Reserved**	Reserved.
[1]	**RXFULL**	**Receive Buffer Full Indicator** 0 = Receive buffer is not full. 1 = Receive buffer is full.
[0]	**RXEMPTY**	**Receive Buffer Empty Indicator**

		0 = Receive buffer is not empty. 1 = Receive buffer is empty.

USCI PDMA Control Register (USPI_PDMACTL)

Register	Offset	R/W	Description	Reset Value
USPI_PDMACTL	USPIn_BA+0x40	R/W	USCI PDMA Control Register	0x0000_0000

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

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**PDMAEN**	**PDMA Mode Enable Bit** 0 = PDMA function Disabled. 1 = PDMA function Enabled.
[2]	**RXPDMAEN**	**PDMA Receive Channel Available** 0 = Receive PDMA function Disabled. 1 = Receive PDMA function Enabled.
[1]	**TXPDMAEN**	**PDMA Transmit Channel Available** 0 = Transmit PDMA function Disabled. 1 = Transmit PDMA function Enabled.
[0]	**PDMARST**	**PDMA Reset** 0 = No effect. 1 = Reset the USCI's PDMA control logic. This bit will be cleared to 0 automatically.

USCI Wake-up Control Register (USPI_WKCTL)

Register	Offset	R/W	Description	Reset Value
USPI_WKCTL	USPIn_BA+0x54	R/W	USCI Wake-up Control Register	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**PDBOPT**	**Power Down Blocking Option** 0 = If user attempts to enter Power-down mode by executing WFI while the protocol is in transferring, MCU will stop the transfer and enter Power-down mode immediately. 1 = If user attempts to enter Power-down mode by executing WFI while the protocol is in transferring, the on-going transfer will not be stopped and MCU will enter idle mode immediately.
[1]	**Reserved**	Reserved.
[0]	**WKEN**	**Wake-up Enable Bit** 0 = Wake-up function Disabled. 1 = Wake-up function Enabled.

USCI Wake-up Status Register (USPI_WKSTS)

Register	Offset	R/W	Description	Reset Value
USPI_WKSTS	USPIn_BA+0x58	R/W	USCI Wake-up Status Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**WKF**	**Wake-up Flag** When chip is woken up from Power-down mode, this bit is set to 1. Software can write 1 to clear this bit.

USCI Protocol Control Register – USPI_PROTCTL (SPI)

Register	Offset	R/W	Description	Reset Value
USPI_PROTCTL	USPIn_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0300

31	30	29	28	27	26	25	24
PROTEN	Reserved		TXUDRPOL	Reserved		SLVTOCNT	
23	22	21	20	19	18	17	16
SLVTOCNT							
15	14	13	12	11	10	9	8
Reserved	TSMSEL			SUSPITV			
7	6	5	4	3	2	1	0
SCLKMODE		Reserved		AUTOSS	SS	SLV3WIRE	SLAVE

Bits	Description	
[31]	**PROTEN**	**SPI Protocol Enable Bit** 0 = SPI Protocol Disabled. 1 = SPI Protocol Enabled.
[30:29]	**Reserved**	Reserved.
[28]	**TXUDRPOL**	**Transmit Under-run Data Polarity (for Slave)** This bit defines the transmitting data value of USCIx_DAT1 when no data is available for transferring. 0 = The output data value is 0 if TX under run event occurs. 1 = The output data value is 1 if TX under run event occurs.
[27:26]	**Reserved**	Reserved.
[25:16]	**SLVTOCNT**	**Slave Mode Time-out Period (Slave Only)** In Slave mode, this bit field is used for Slave time-out period. This bit field indicates how many clock periods (selected by TMCNTSRC, USPI_BRGEN[5]) between the two edges of input SCLK will assert the Slave time-out event. Writing 0x0 into this bit field will disable the Slave time-out function. Example: Assume SLVTOCNT is 0x0A and TMCNTSRC (USPI_BRGEN[5]) is 1, it means the time-out event will occur if the state of SPI bus clock pin is not changed more than (10+1) periods of f_{DIV_CLK}.
[15]	**Reserved**	Reserved.
[14:12]	**TSMSEL**	**Transmit Data Mode Selection** This bit field describes how receive and transmit data is shifted in and out. TSMSEL = 000b: Full-duplex SPI. TSMSEL = 100b: Half-duplex SPI. Other values are reserved. **Note:** Changing the value of this bit field will produce the TXRST and RXRST to clear the TX/RX data buffer automatically.
[11:8]	**SUSPITV**	**Suspend Interval (Master Only)** This bit field provides the configurable suspend interval between two successive transmit/receive transaction in a transfer. The definition of the suspend interval is the

		interval between the last clock edge of the preceding transaction word and the first clock edge of the following transaction word. The default value is 0x3. The period of the suspend interval is obtained according to the following equation. (SUSPITV[3:0] + 0.5) * period of SPI_CLK clock cycle Example: SUSPITV = 0x0 … 0.5 SPI_CLK clock cycle. SUSPITV = 0x1 … 1.5 SPI_CLK clock cycle. …… SUSPITV = 0xE … 14.5 SPI_CLK clock cycle. SUSPITV = 0xF … 15.5 SPI_CLK clock cycle.
[7:6]	**SCLKMODE**	**Serial Bus Clock Mode** This bit field defines the SCLK idle status, data transmit, and data receive edge. MODE0 = The idle state of SPI clock is low level. Data is transmitted with falling edge and received with rising edge. MODE1 = The idle state of SPI clock is low level. Data is transmitted with rising edge and received with falling edge. MODE2 = The idle state of SPI clock is high level. Data is transmitted with rising edge and received with falling edge. MODE3 = The idle state of SPI clock is high level. Data is transmitted with falling edge and received with rising edge.
[5:4]	**Reserved**	Reserved.
[3]	**AUTOSS**	**Automatic Slave Select Function Enable (Master Only)** 0 = Slave select signal will be controlled by the setting value of SS (USPI_PROTCTL[2]) bit. 1 = Slave select signal will be generated automatically. The slave select signal will be asserted by the SPI controller when transmit/receive is started, and will be de-asserted after each transmit/receive is finished.
[2]	**SS**	**Slave Select Control (Master Only)** If AUTOSS bit is cleared, setting this bit to 1 will set the slave select signal to active state, and setting this bit to 0 will set the slave select back to inactive state. If the AUTOSS function is enabled (AUTOSS = 1), the setting value of this bit will not affect the current state of slave select signal. **Note:** In SPI protocol, the internal slave select signal is active high.
[1]	**SLV3WIRE**	**Slave 3-wire Mode Selection (Slave Only)** The SPI protocol can work with 3-wire interface (without slave select signal) in Slave mode. 0 = 4-wire bi-direction interface. 1 = 3-wire bi-direction interface.
[0]	**SLAVE**	**Slave Mode Selection** 0 = Master mode. 1 = Slave mode.

USCI Protocol Interrupt Enable Register – USPI_PROTIEN (SPI)

Register	Offset	R/W	Description	Reset Value
USPI_PROTIEN	USPIn_BA+0x60	R/W	USCI Protocol Interrupt Enable Register	0x0000_0000

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

Bits	Description	
[31:5]	**Reserved**	Reserved.
[3]	**SLVBEIEN**	**Slave Mode Bit Count Error Interrupt Enable Bit** If data transfer is terminated by slave time-out or slave select inactive event in Slave mode, so that the transmit/receive data bit count does not match the setting of DWIDTH (USPI_LINECTL[11:8]). Bit count error event occurs. 0 = The Slave mode bit count error interrupt Disabled. 1 = The Slave mode bit count error interrupt Enabled.
[2]	**SLVTOIEN**	**Slave Time-out Interrupt Enable Bit** In SPI protocol, this bit enables the interrupt generation in case of a Slave time-out event. 0 = The Slave time-out interrupt Disabled. 1 = The Slave time-out interrupt Enabled.
[1]	**SSACTIEN**	**Slave Select Active Interrupt Enable Bit** This bit enables/disables the generation of a slave select interrupt if the slave select changes to active. 0 = Slave select active interrupt generation Disabled. 1 = Slave select active interrupt generation Enabled.
[0]	**SSINAIEN**	**Slave Select Inactive Interrupt Enable Bit** This bit enables/disables the generation of a slave select interrupt if the slave select changes to inactive. 0 = Slave select inactive interrupt generation Disabled. 1 = Slave select inactive interrupt generation Enabled.

USCI Protocol Status Register – USPI_PROTSTS (SPI)

Register	Offset	R/W	Description	Reset Value
USPI_PROTSTS	USPIn_BA+0x64	R/W	USCI Protocol Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					SLVUDR	BUSY	SSLINE
15	14	13	12	11	10	9	8
Reserved						SSACTIF	SSINAIF
7	6	5	4	3	2	1	0
Reserved	SLVBEIF	SLVTOIF	RXENDIF	RXSTIF	TXENDIF	TXSTIF	Reserved

Bits	Description	
[31:19]	**Reserved**	Reserved.
[18]	**SLVUDR**	**Slave Mode Transmit Under-run Status (Read Only)** In Slave mode, if there is no available transmit data in buffer while transmit data shift out caused by input serial bus clock, this status flag will be set to 1. This bit indicates whether the current shift-out data of word transmission is switched to TXUDRPOL (USPI_PROTCTL[28]) or not. 0 = Slave transmit under run event does not occur. 1 = Slave transmit under run event occurs.
[17]	**BUSY**	**Busy Status (Read Only)** 0 = SPI is in idle state. 1 = SPI is in busy state. The following listing are the bus busy conditions: a. USPI_PROTCTL[31] = 1 and the TXEMPTY = 0. b. For SPI Master mode, the TXEMPTY = 1 but the current transaction is not finished yet. c. For SPI Slave mode, the USPI_PROTCTL[31] = 1 and there is serial clock input into the SPI core logic when slave select is active. d. For SPI Slave mode, the USPI_PROTCTL[31] = 1 and the transmit buffer or transmit shift register is not empty even if the slave select is inactive.
[16]	**SSLINE**	**Slave Select Line Bus Status (Read Only)** This bit is only available in Slave mode. It used to monitor the current status of the input slave select signal on the bus. 0 = The slave select line status is 0. 1 = The slave select line status is 1.
[15:10]	**Reserved**	Reserved.
[9]	**SSACTIF**	**Slave Select Active Interrupt Flag (for Slave Only)** This bit indicates that the internal slave select signal has changed to active. It is cleared by software writes one to this bit 0 = The slave select signal has not changed to active.

		1 = The slave select signal has changed to active. **Note:** The internal slave select signal is active high.
[8]	**SSINAIF**	**Slave Select Inactive Interrupt Flag (for Slave Only)** This bit indicates that the internal slave select signal has changed to inactive. It is cleared by software writes 1 to this bit 0 = The slave select signal has not changed to inactive. 1 = The slave select signal has changed to inactive. **Note:** The internal slave select signal is active high.
[7]	**Reserved**	Reserved.
[6]	**SLVBEIF**	**Slave Bit Count Error Interrupt Flag (for Slave Only)** 0 = Slave bit count error event did not occur. 1 = Slave bit count error event occurred. **Note:** It is cleared by software write 1 to this bit. If the transmit/receive data bit count does not match the setting of DWIDTH (USPI_LINECTL[11:8]), bit count error event occurs.
[5]	**SLVTOIF**	**Slave Time-out Interrupt Flag (for Slave Only)** 0 = Slave time-out event did not occur. 1 = Slave time-out event occurred. **Note:** It is cleared by software write 1 to this bit
[4]	**RXENDIF**	**Receive End Interrupt Flag** 0 = Receive end event did not occur. 1 = Receive end event occurred. **Note:** It is cleared by software write 1 to this bit. The receive end event happens when hardware receives the last bit of RX data into shift data unit.
[3]	**RXSTIF**	**Receive Start Interrupt Flag** 0 = Receive start event did not occur. 1 = Receive start event occurred. **Note:** It is cleared by software write 1 to this bit. For SPI master mode, the receive start event happens when SPI master sends slave select active and spi clock to the external SPI slave. For SPI slave mode, the receive start event happens when slave select of SPI slave is active and spi clock of SPI slave is inputed from the external SPI master.
[2]	**TXENDIF**	**Transmit End Interrupt Flag** 0 = Transmit end event did not occur. 1 = Transmit end event occurred. **Note:** It is cleared by software write 1 to this bit. The transmit end event happens when hardware sends the last bit of TX data from shift data unit.
[1]	**TXSTIF**	**Transmit Start Interrupt Flag** 0 = Transmit start event did not occur. 1 = Transmit start event occurred. **Note:** It is cleared by software write 1 to this bit. The transmit start event happens when hardware starts to move TX data from data buffer to shift data unit.
[0]	**Reserved**	Reserved.

6.20 USCI - I²C Mode

6.20.1 Overview

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. 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 6.20-1 for more detailed I²C BUS Timing.

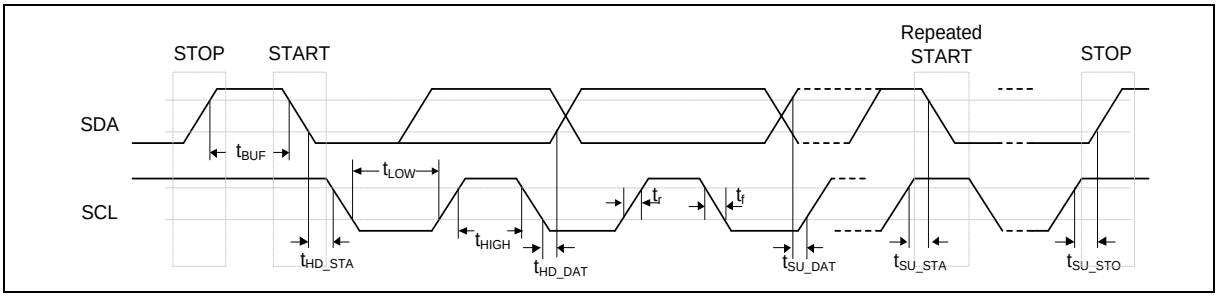


Figure 6.20-1 I²C Bus Timing

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. The I²C mode is selected by FUNMODE (UI2C_CTL [2:0]) = 100B. When enable this port, the USCI 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 because the SDA and SCL are set to open-drain pins when USCI is selected to I²C operation mode .

6.20.2 Features

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

6.20.3 Block Diagram

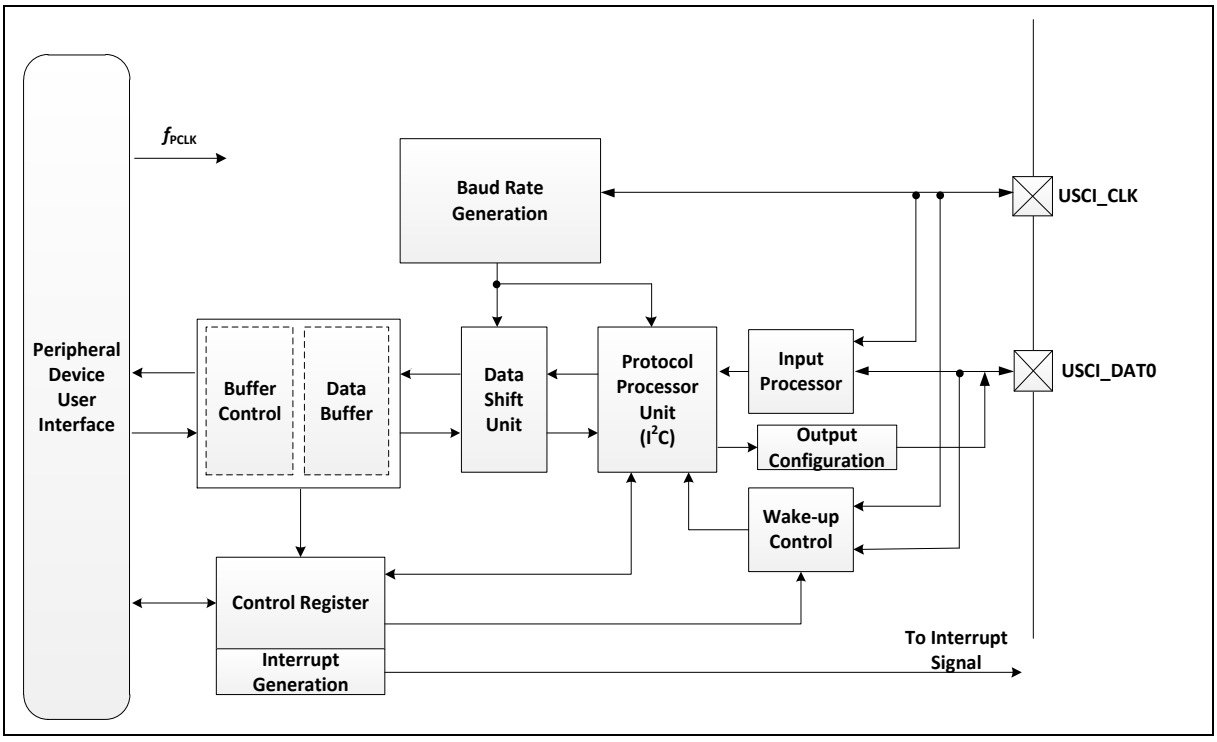


Figure 6.20-2 USCI I²C Mode Block Diagram

6.20.4 Basic Configuration

6.20.4.1 USCI0 I²C Basic Configurations

The basic configurations of USCI0_I2C are as follows:

- Clock Source Configuration
  - Enable USCI0 clock (USCI0CKEN) on CLK_APBCLK1[8] register.
  -
  - Enable USCI0_I2C function UI2C_CTL[2:0] register, UI2C_CTL[2:0]=3'b100.
- Reset Configuration
  - Reset USCI0 controller in USCI0RST (SYS_IPRST2[8]).

6.20.4.2 USCI1 I²C Basic Configurations

The basic configurations of USCI1_I2C are as follows:

- Clock Source Configuration
  - Enable USCI1 clock (USCI1CKEN) on CLK_APBCLK1[9] register.
  - Enable USCI1_I2C function UI2C_CTL[2:0] register, UI2C_CTL[2:0]=3'b100.
- Reset Configuration
  - Reset USCI1 controller in USCI1RST (SYS_IPRST2[9]).

6.20.5 Functional Description

6.20.5.1 START or Repeated START Signal

Figure 6.20-3 shows the typical I^{2}C protocol. Normally, a standard communication consists of four parts:

- START or Repeated START signal generation
- Slave address and R/W bit transfer
- Data transfer
- STOP signal generation

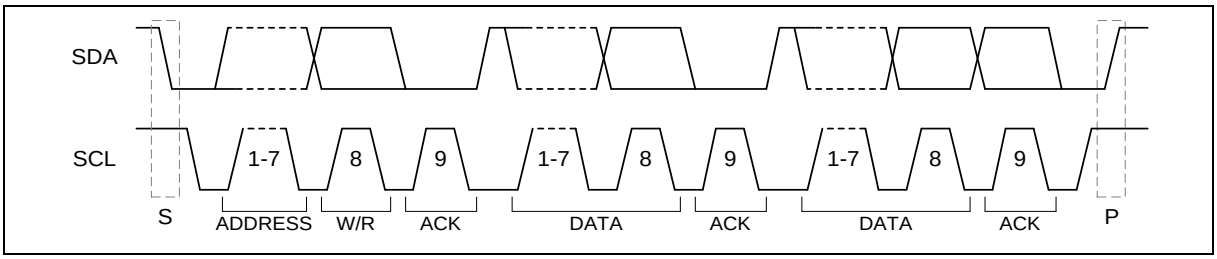


Figure 6.20-3 I^{2}C Protocol

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

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

6.20.5.2 STOP Signal

The master can terminate the communication by generating a STOP signal. A STOP signal, usually referred to as the “P” bit, is defined as a LOW to HIGH transition on the SDA line while SCL is HIGH. The section between STOP and START is called bus free.

Figure 6.20-4 shows the waveform of START, Repeat START and STOP.

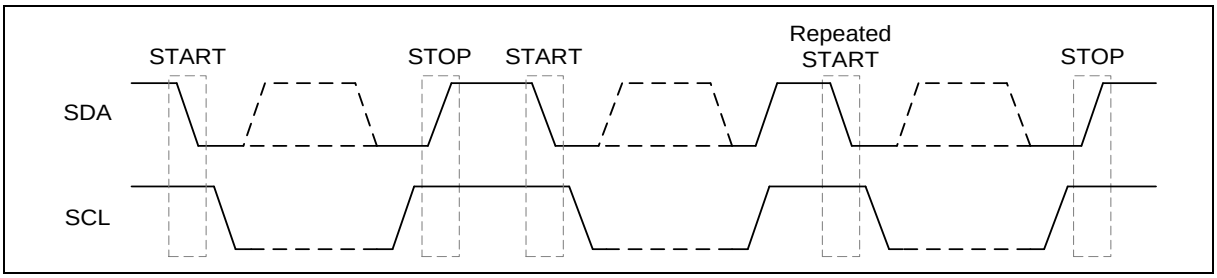


Figure 6.20-4 START and STOP Conditions

6.20.5.3 Slave Address Transfer

After a (repeated) start condition, the master sends a slave address to identify the target device of the communication. The start address can comprise one or two address bytes (for 7-bit or for 10-bit addressing schemes). After an address byte, a slave sensitive to the transmitted address has to acknowledge the reception.

Therefore, the slave’s address can be programmed in the device, where it is compared to the received

address. In case of a match, the slave answers with an acknowledge (SDA = 0). Slaves that are not targeted answer with a non-acknowledge (SDA = 1). In addition to the match of the programmed address, another address byte value has to be answered with an acknowledge if the slave is capable to handle the corresponding requests. The address byte 00H indicates a general call address that can be acknowledged.

In order to allow selective acknowledges for the different values of the address byte(s), the following control mechanism is implemented:

- If the GCFUNC bit (UI2C_PROTCTL [0]) is set the I^2C port hardware will respond to General Call address (00H). Clear GC bit to disable general call function.
- The I^2C port is equipped with one device address registers, UI2C_DEVADDRn (n = 0~1). In 7-bit address mode, the first 7 bits of a received first address byte are compared to the programmed slave address (UI2C_DEVADDRn [6:0]). If these bits match, the slave sends an acknowledge.
- For 10 bit addressing mode, if the slave address is programmed to 1111 0XXB, the XX bits are compared to the bits UI2C_DEVADDR [9:8] to check for address match and also sends an acknowledge when ADDR10EN (UI2C_PROTCTL [4]) is set. The slave waits for a second address byte compares it with UI2C_DEVADDR [7:0] and sends an acknowledge accordingly to cover the 10 bit addressing mode. The user has to take care about reserved addresses (refer to I^2C specification for more detailed description). Only the address 1111 0XXB is supported. Under each of these conditions, bit SLASEL (UI2C_PROTSTS [14]) will be set when the addressing delivered a match. This SLASEL (UI2C_PROTSTS [14]) bit is cleared automatically by a (Repeated) START or STOP condition.
- The I^2C port is equipped multiple address recognition with one address mask registers UI2C_ADDRMSKn (n = 0~1). When the bit in the address mask register is set to 1, it means the received corresponding address bit is "Don't care". If the bit is set to 0, it means the received corresponding register bit should be exactly the same as address register.

6.20.5.4 Data Transfer

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

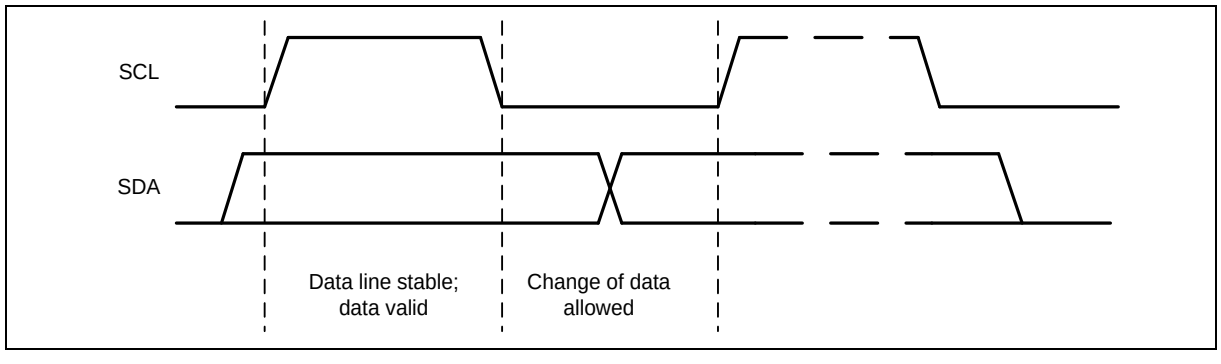


Figure 6.20-5 Bit Transfer on the I^2C Bus

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

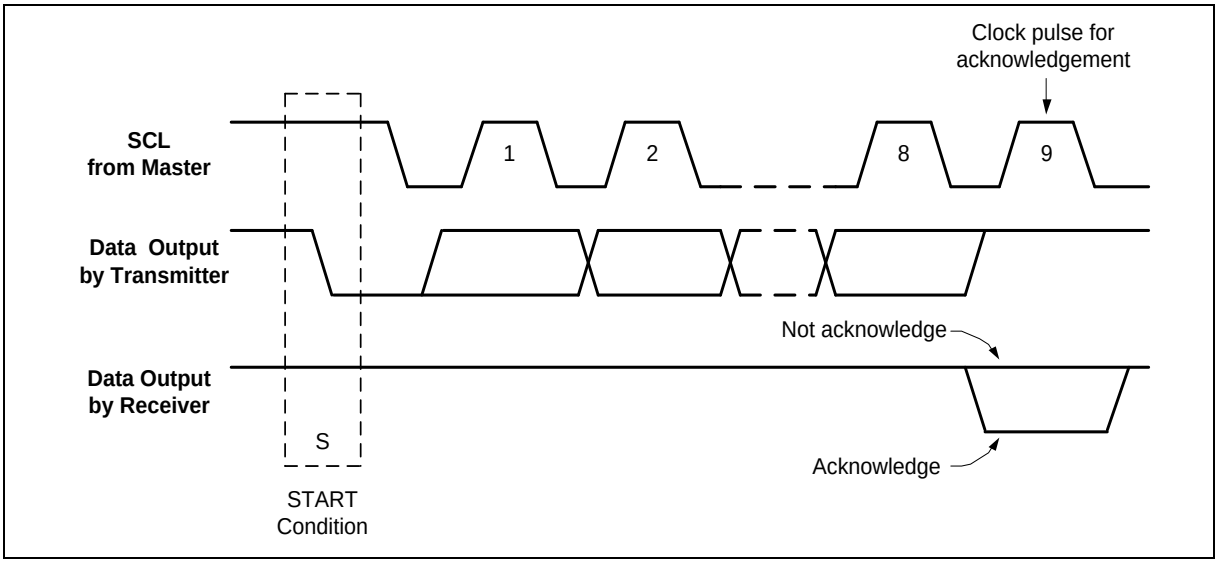


Figure 6.20-6 Acknowledge on the I²C Bus

6.20.5.5 Clock Baud Rate Bits

The data baud rate of I²C is determined by UI2C_BRGEN 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. The bits RCLKSEL, SPCLKSEL, PDSCNT, and DSCNT define the baud rate setting:

- RCLKSEL (UI2C_BRGEN [0])

  to define the input frequency f_{REF_CLK}

- SPCLKSEL (UI2C_BRGEN[3:2])

  to define the multiple source of the sample clock f_{SAMP_CLK}

- PDSCNT (UI2C_BRGEN [9:8])

  to define the length of a data sample time (division of f_{REF_CLK} by 1, 2, 3, or 4)

- DSCNT (UI2C_BRGEN [14:10])

  to define the number of data sample time per bit time

  The standard setting is given by RCLKSEL = 0 ($f_{REF\_CLK} = f_{PCLK}$), PTCLKSEL = 0 ($f_{PROT\_CLK} = f_{REF\_CLK}$) and SPCLKSEL = 2'b00 ($f_{SAMP\_CLK} = f_{DIV\_CLK}$). Under these conditions, the baud rate is given by:

$$f_{I2C} = f_{REF\_CLK} \times \frac{1}{\text{CLKDIV} + 1} \times \frac{1}{\text{PDSCNT} + 1} \times \frac{1}{\text{DSCNT} + 1}$$

  In order to generate slower frequencies, additional divide-by-2 stages can be selected by PTCLKSEL = 1 ($f_{PROT\_CLK} = f_{REF\_CLK2}$), leading to:

$$f_{I2C} = \frac{f_{REF\_CLK}}{2} \times \frac{1}{\text{CLKDIV} + 1} \times \frac{1}{\text{PDSCNT} + 1} \times \frac{1}{\text{DSCNT} + 1}$$

  If SPCLKSEL = 2'b10 ($f_{SAMP\_CLK} = f_{SCLK}$), and RCLKSEL = 0 ($f_{REF\_CLK} = f_{PCLK}$), PTCLKSEL = 0 ($f_{PROT\_CLK} = f_{REF\_CLK}$). The baud rate is given by:

$$f_{I2C} = f_{REF\_CLK} \times \frac{1}{\text{CLKDIV} + 1} \times \frac{1}{2} \times \frac{1}{\text{PDSCNT} + 1} \times \frac{1}{\text{DSCNT} + 1}$$

6.20.5.6 Byte Stretching

If a device is selected as master/slave transmit mode and should transmit a data byte but the transmit buffer TXDAT does not contain valid data to be transmitted, the device ties down SCL = 0 at the end of the previous acknowledge bit. The waiting period is finished if software writes 1 to PTRG (UI2C_PROTCTL [5]).

6.20.5.7 Multi-master Arbitration

In some applications, there are two or more masters on the same I^2C bus to access slaves, and the masters may transmit data simultaneously. The I^2C supports multi-master by including collision detection and arbitration to prevent data corruption.

If two masters sometimes initiate I^2C command at the same time, the arbitration procedure determines which master wins and can continue with the command. Arbitration is performed on the SDA signal while the SCL signal is high. Each master checks if the SDA signal on the bus corresponds to the generated SDA signal. If the SDA signal on the bus is low but it should be high, then this master has lost arbitration. Master I^2C device that has lost arbitration can generate SCL pulses until the byte ends and must then release the bus and go into slave mode. The arbitration procedure can continue until all the data is transferred. This means that in multi-master system each I^2C master must monitor the I^2C bus for collisions and act accordingly. Figure 6.20-7 describes master1 data and master2 data are compete arbitration.

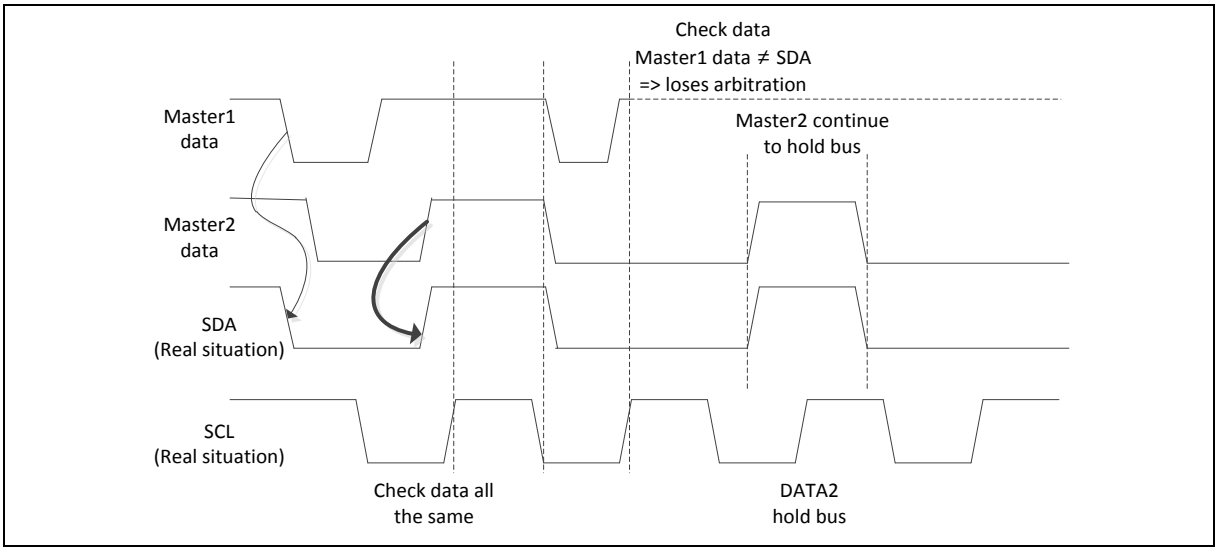


Figure 6.20-7 Arbitration Lost

In this case, during the address and data transmission, the master transmitter checks at the rising edge of SCL for each data bit if the value it is sending is equal to the value read on the SDA line. If yes, master can hold bus continuously. If this is not the case (transmitted value = 1, value read = 0), the master has lost the transmit arbitration. This is indicated by interrupt flag ARBLOIF (UI2C_PROTSTS [11]) and can generate a protocol interrupt if enabled by ARBLOIEN (UI2C_PROTIEN [4]).

When the transmit arbitration has been lost, the software has to initialize the complete frame again, starting with the first address byte together with the STARTcondition for a new master transmit attempt. Arbitration also takes place for the ACK bit. If master arbitration lost and match the device address, then master will turn to slave.

6.20.5.8 Transmission Chain

The I^2C bus protocol requiring a kind of in-bit-response during the arbitration phase and while a slave is transmitting, the resulting loop delay of the transmission chain can limit the reachable maximal baud rate, strongly depending on the bus characteristics (bus load, module frequency, etc.).

The shift clock SCL is generated by the master device, output on the wire, then it passes through the input stage and the input filter. Now, the edges can be detected and the SDA data signal can be generated accordingly. The SDA signal passes through the output stage and the wire to the master receiver part. There, it passes through the input stage and the input filter before it is sampled.

This complete loop has to be finished (including all settling times to obtain stable signal levels) before the SCL signal changes again. The delays in this path have to be taken into account for the calculation of the baud rate as a function of f_{PCLK} and f_{PROT_CLK}. We suggest user adopt f_{PCLK}.

6.20.5.9 Non-Acknowledge and Error Conditions

In case of a non-acknowledge (NACKIF (UI2C_PROTSTS [10])) or an error (ERRIF(UI2C_PROTSTS [12])), no further transmission will take place. User software doesn't invalidate the transmit buffer and disable transmissions, before configuring the transmission (by writing TXDAT) again with appropriate values to react on the previous event.

6.20.5.10 I^2C Protocol Interrupt Events

The following protocol-related events are generated in I^2C mode and can lead to a protocol interrupt.

Please note that the bits in register UI2C_PROTSTS are not all automatically cleared by hardware and have to be cleared by software in order to monitor new incoming events.

- START condition received at a correct position in a frame (STARIF (UI2C_PROTSTS [8]))
- STOP condition transferred at a correct position in a frame (STORIF (UI2C_PROTSTS [9]))
- Master arbitration lost (ARBLOIF (UI2C_PROTSTS [11]))
- Slave read requested (SLAREAD (UI2C_PROTSTS [15]))
- Acknowledge received (ACKIF (UI2C_PROTSTS [13]))
- Non-acknowledge received (NACKIF (UI2C_PROTSTS [10]))
- START condition not at the expected position in a frame (ERRIF (UI2C_PROTSTS [12]))
- STOP condition not at the expected position in a frame (ERRIF (UI2C_PROTSTS [12]))

6.20.5.11 Operating the I2C

To operate the I^2C protocol, the following issues have to be considered:

Select I^2C Mode

It is recommended to configure all parameters of the I^2C that do not change during run time while FUNMODE (UI2C_CTL [2:0]) = 000B. The I^2C control flow has to be done while FUNMODE (UI2C_CTL [2:0]) = 000B to avoid unintended edges of the input signals and the I^2C mode can be enabled by FUNMODE (UI2C_CTL [2:0]) = 100B afterwards.

Step 1. Set FUNMODE (UI2C_CTL [2:0]) = 000B

Step 2. Set FUNMODE (UI2C_CTL [2:0]) = 100B

Pin Connections

The pins used for SDA and SCL have to be set to open-drain mode by USCI controller to support the wired-AND structure of the I^2C bus lines.

Note: The step to enable the alternate output port functions should only be done after the I^2C mode is enabled, to avoided unintended spikes on the output.

Bit Timing Configuration

In standard mode (100 kBit/s) a minimum module frequency of 2 MHz is necessary, whereas in fast mode (400 kBit/s) a minimum of 10 MHz is required. Additionally, if the digital filter stage should be

used to eliminate spikes up to 50 ns, a filter frequency of 20 MHz is necessary. There could be an uncertainty in the SCL high phase timing of maximum $1/f_{PROT\_CLK}$ if another I²C participant lengthens the SCL low phase on the bus. Note that the SCL maximum frequency is $f_{SAMP\_CLK}/2$ and the SPCLKSEL (UI2C_BRGEN [3:2]) must be set to 0 for selecting $f_{SAMP\_CLK} = f_{DIV\_CLK}$.

Data Format Configuration

The data format has to be configured for 8 data bits (DWIDTH (UI2C_LINECTL [11:8]) = 8), and MSB shifted first (LSB (UI2C_LINECTL [0]) = 0). As a result, UI2C_LINECTL has to be set to 0x800.

Control Flow

The on-chip 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 bit), acknowledge pulse will be transmitted out on the 9th clock, hence an interrupt is requested on both master and slave devices if interrupt is enabled. When the microcontroller wishes to become the bus master, hardware waits until the bus is free before entering Master mode so that a possible slave action is not be interrupted. If address arbitration is lost in Master mode, 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 UI2C_PROTCTL, UI2C_PROTIEN, TXDAT registers according to current status of UI2C_PROTSTS register. In other words, for each I²C bus action, user needs to check current status by UI2C_PROTSTS register, and then set UI2C_PROTCTL, UI2C_PROTIEN, TXDAT registers to take bus action. Finally, check the response status by UI2C_PROTSTS.

The bits, STA, STO and AA in UI2C_PROTCTL register are used to control the next state of the I²C hardware after interrupt signal is cleared. Upon completion of the new action, a new status will be updated in UI2C_PROTSTS register will be set. If the I²C interrupt control bit of UI2C_PROTIEN is set, appropriate action or software branch of the new status can be performed in the Interrupt service routine.

Figure 6.20-8 shows the current I²C STARIF (UI2C_PROTSTS [8]) is set to 1 by hardware, and then set TXDAT = SLA+W (Slave address + Write bit), (PTRG, STA, STO, AA) = (1, 0, 0, x) to send the address to I²C bus, and write 1 to STARIF (UI2C_PROTSTS [8]) to clear flag. If a slave on the bus matches the address and response ACK, the UI2C_PROTSTS will be updated by ACKIF (UI2C_PROTSTS [13]) setting.

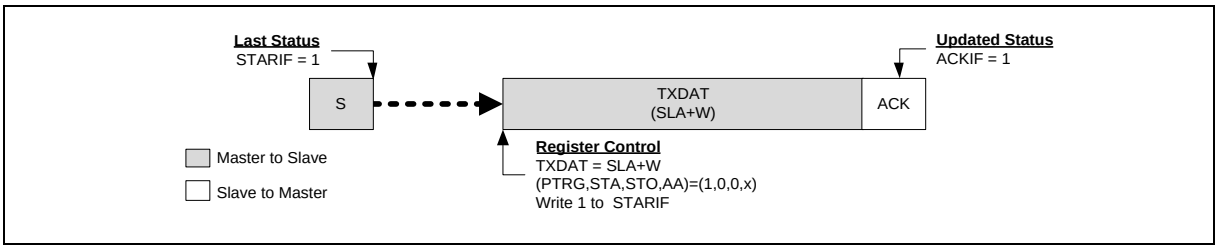


Figure 6.20-8 Control I²C Bus according to Current I²C Status

Data Transfer on the I²C Bus

Figure 6.20-9 shows a master transmits data to slave. A master addresses a slave with a 7-bit address and 1-bit write index to denote that the master wants to transmit data to the slave. The master keeps transmitting data after the slave returns acknowledge to the master.

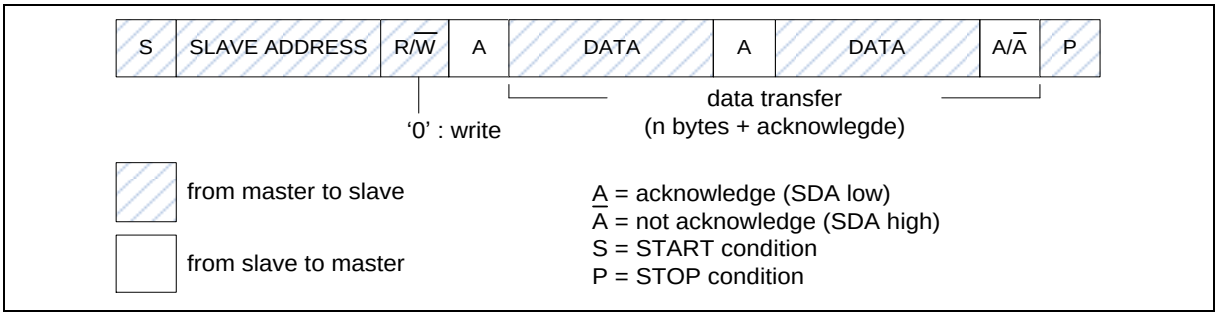


Figure 6.20-9 Master Transmits Data to Slave with a 7-bit Address

Figure 6.20-10 shows a master read data from slave. A master addresses a slave with a 7-bit address and 1-bit read index to denote that the master wants to read data from the slave. The slave will start transmitting data after the slave returns acknowledge to the master.

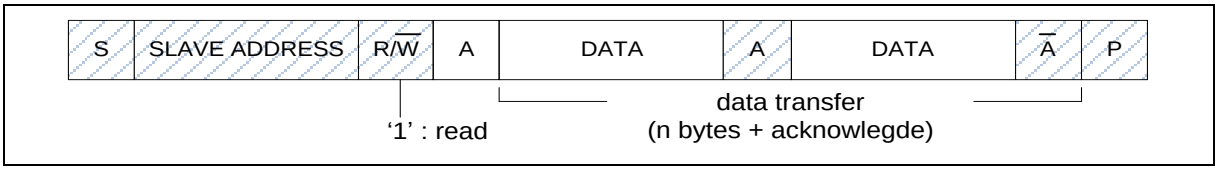


Figure 6.20-10 Master Reads Data from Slave with a 7-bit Address

Figure 6.20-11 shows a master transmits data to slave by 10-bit address. A master addresses a slave with a 10-bit address. First byte contains 10-bit address indicator (5'b11110) and 2-bit address with write index, second byte contains 8-bit address. The master keeps transmitting data after the second byte end. Note that 7-bit and 10-bit address device can work on the same bus.

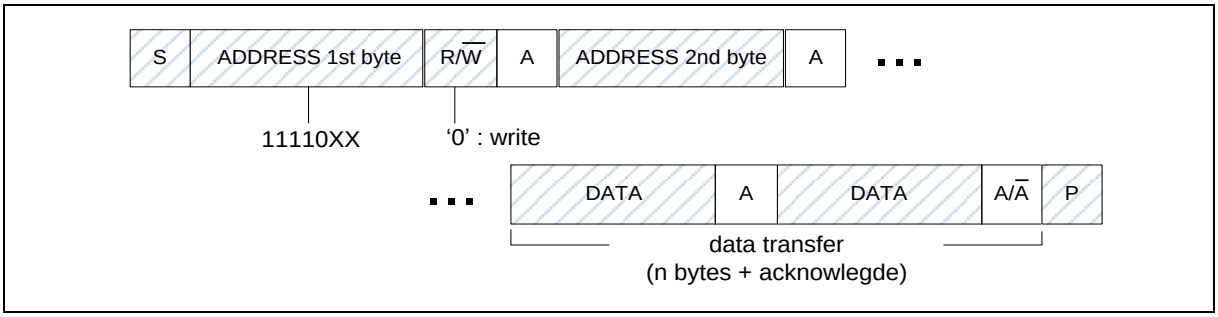


Figure 6.20-11 Master Transmits Data to Slave by 10-bit Address

Figure 6.20-12 shows a master read data from slave by 10-bit address. A master addresses a slave with a 10-bit address. First mater transmits 10-bit address to slave, after that master transmits first byte with read index. The slave will start transmitting data after the first byte with read index.

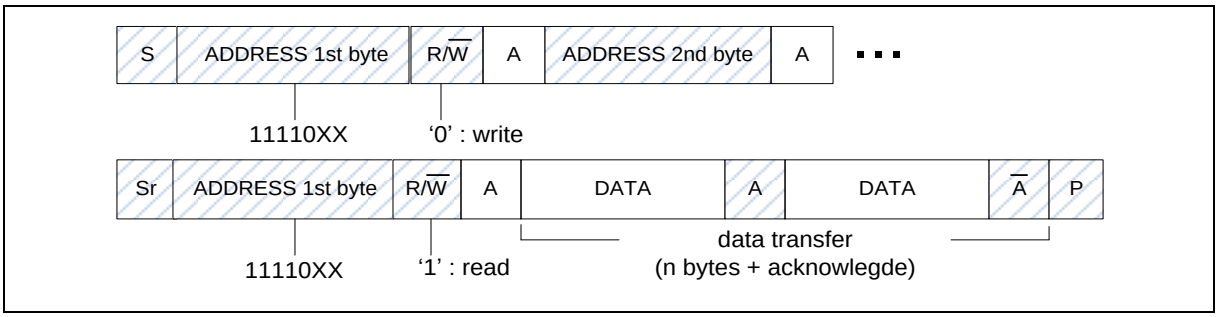


Figure 6.20-12 Master Reads Data from Slave by 10-bit Address

Master Mode

In Figure 6.20-13 and Figure 6.20-14, all possible protocols for I^{2}C master are shown. User needs to follow proper path of the flow to implement required I^{2}C protocol.

In other words, user can send a START signal to bus and I^{2}C will be in Master Transmitter mode (Figure 6.20-13) or Master receiver mode (Figure 6.20-14) after START signal has been sent successfully and new status register would be set STARIF (UI2C_PROTSTS [8]). Followed by START signal, user can send slave address, read/write bit, data and Repeat START, STOP to perform I^{2}C protocol.

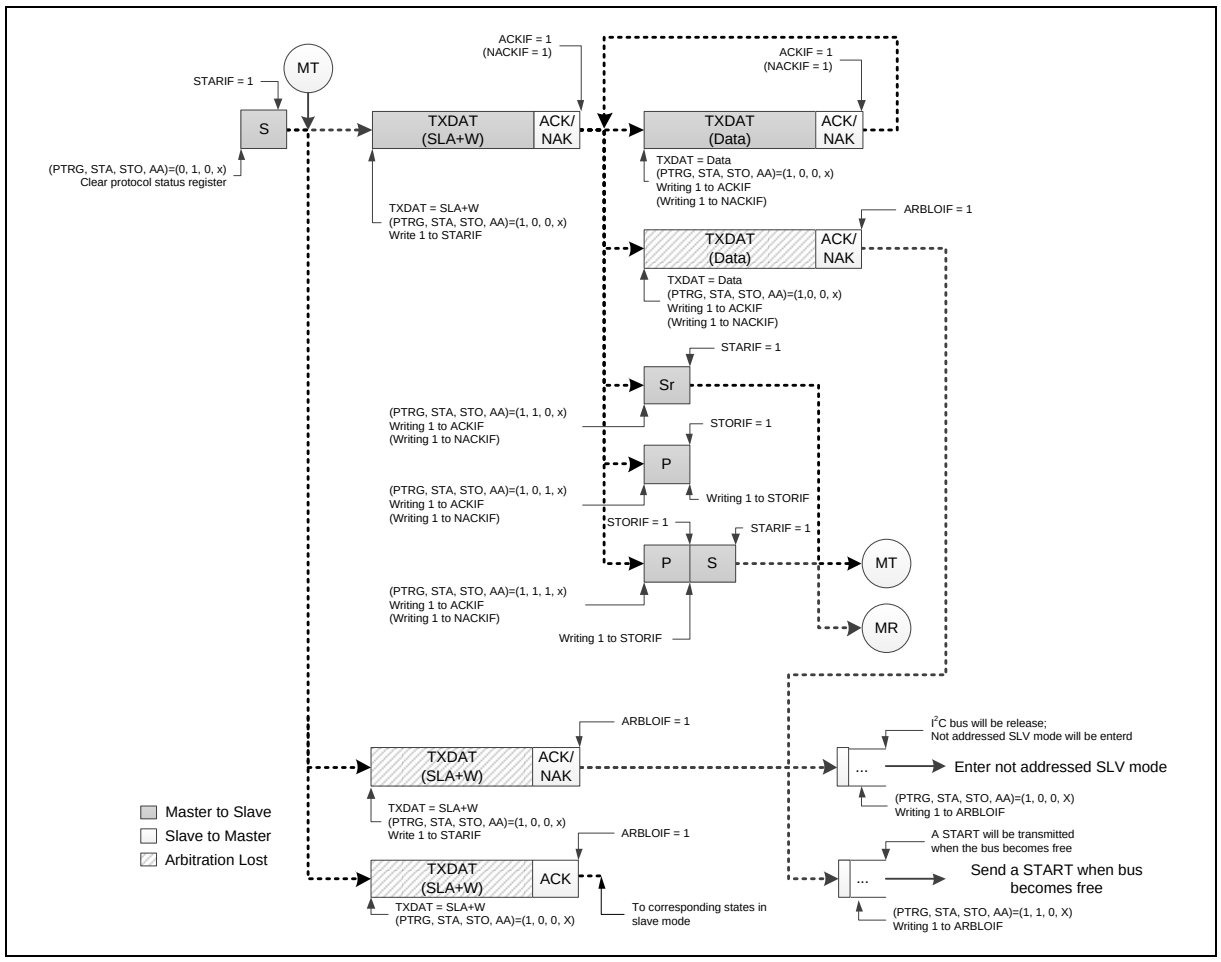


Figure 6.20-13 Master Transmitter Mode Control Flow with 7-bit Address

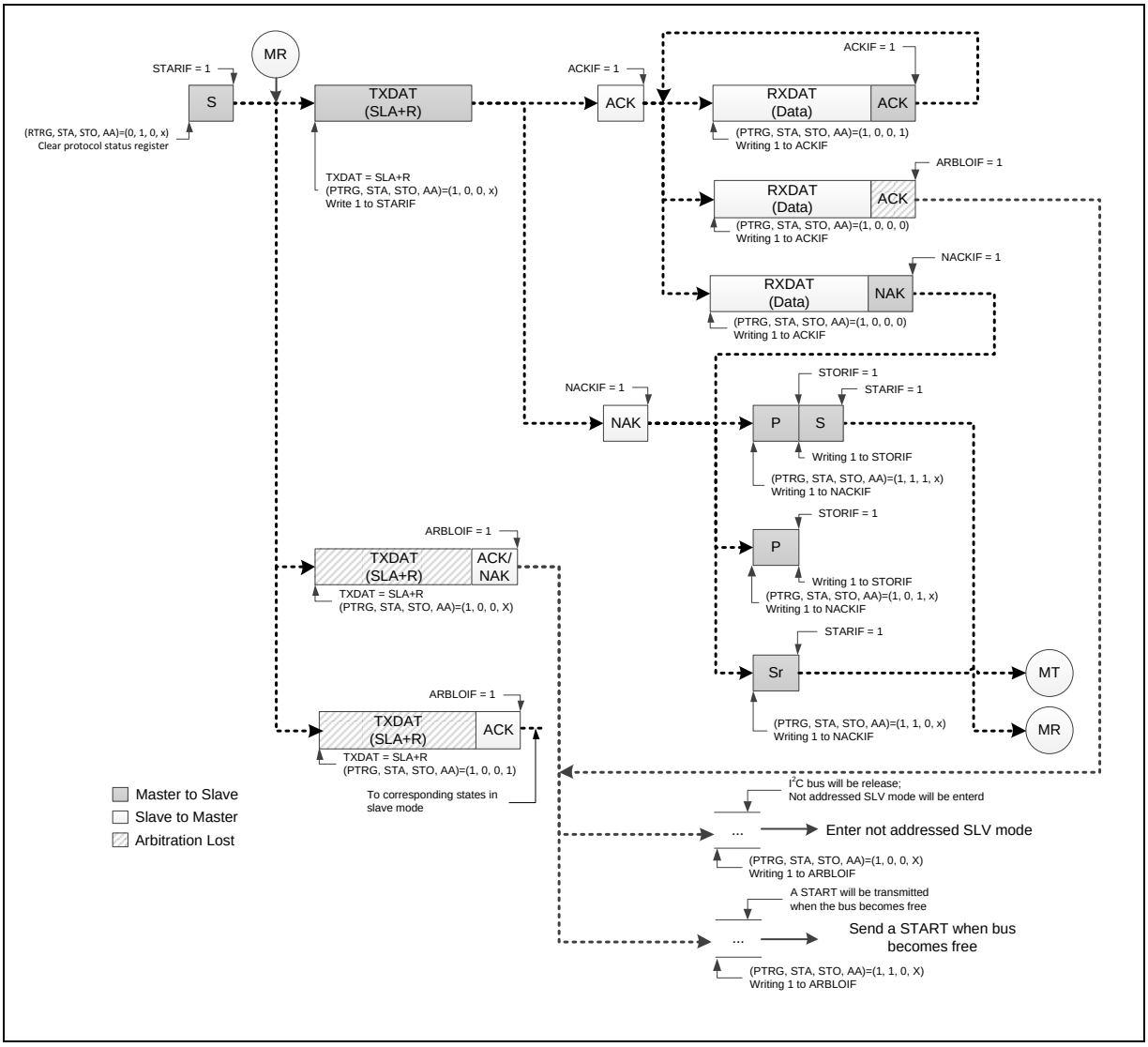


Figure 6.20-14 Master Receiver Mode Control Flow with 7-bit Address

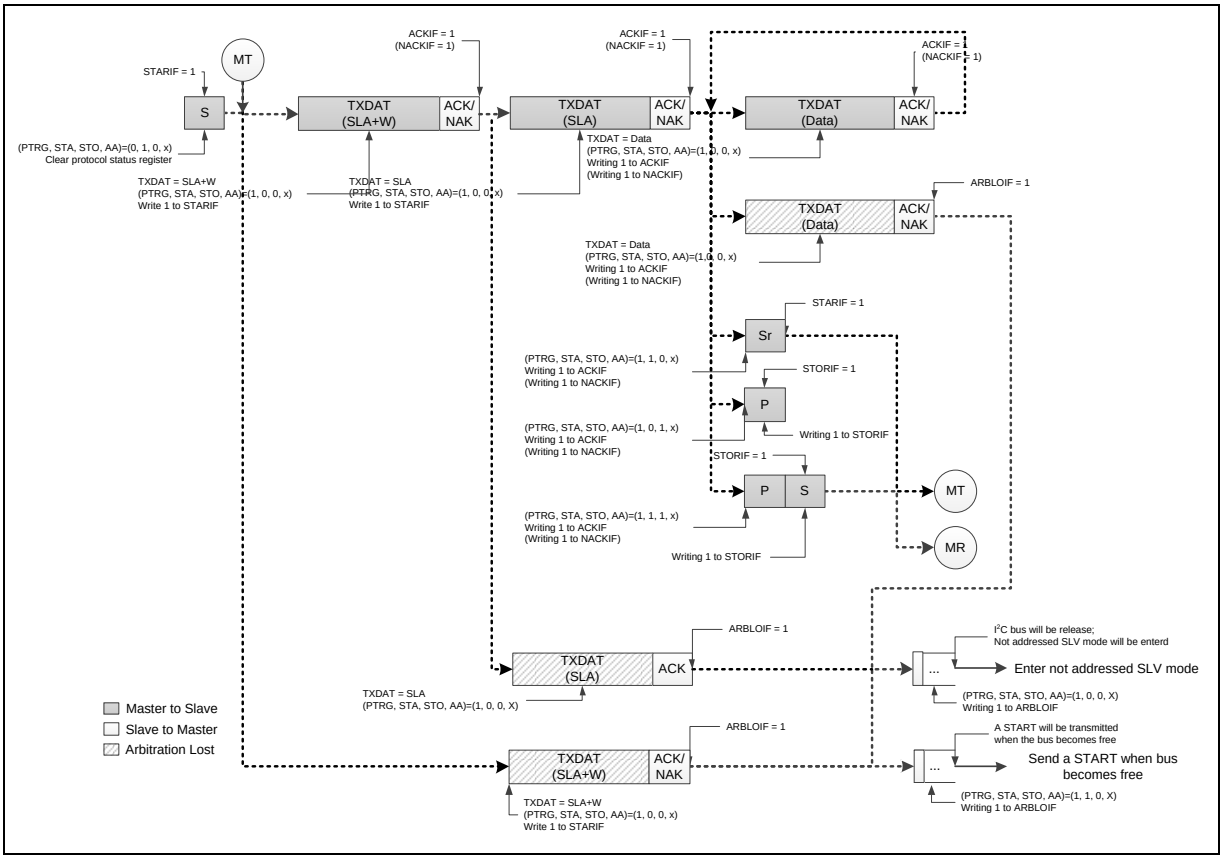


Figure 6.20-15 Master Transmitter Mode Control Flow with 10-bit Address

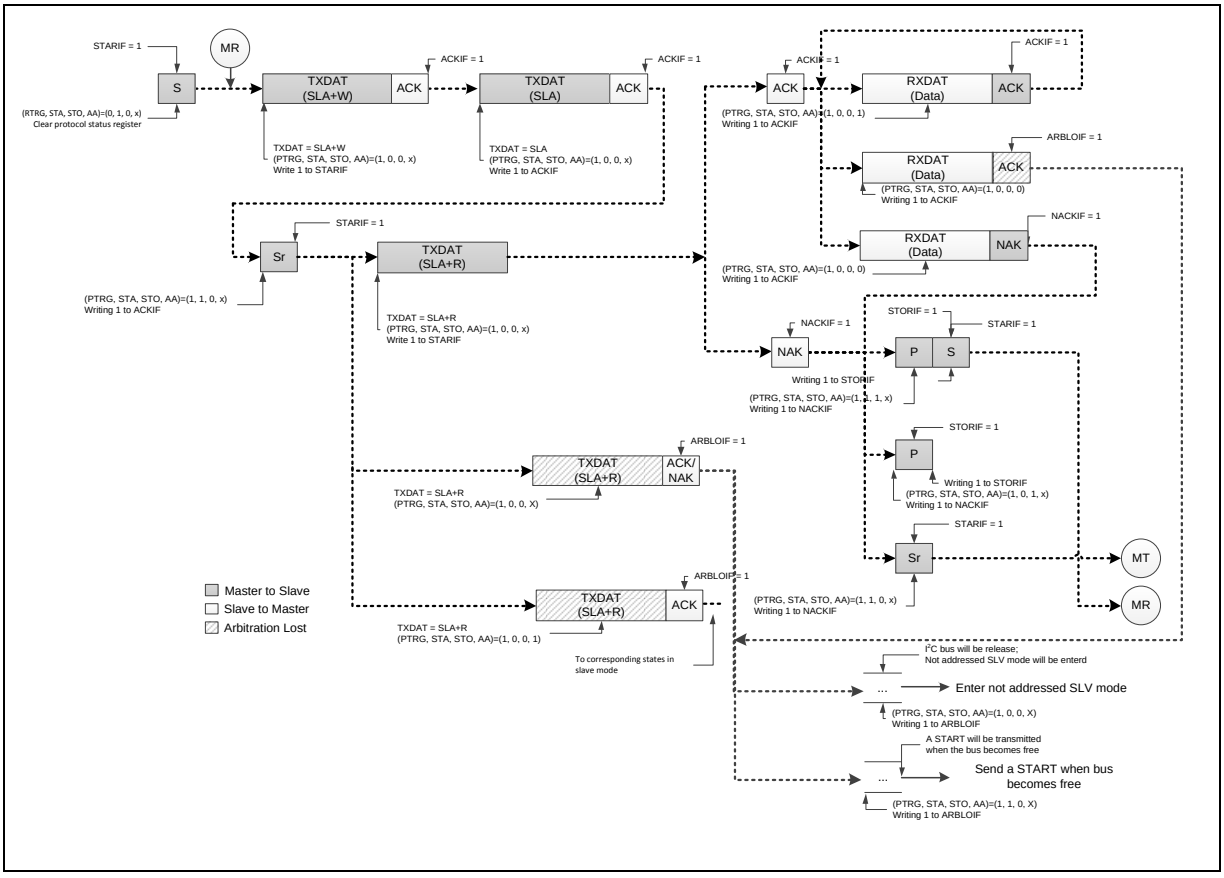


Figure 6.20-16 Master Recevier Mode Control Flow with 10-bit Address

If the I²C is in Master mode and gets arbitration lost, the bit of ARBLOIF (UI2C_PROTSTS [11]) will be set. User may writing 1 to ARBLOIF (UI2C_PROTSTS [11]) and set (PTRG, STA, STO, AA) = (1, 1, 0, X) to send START to re-start Master operation when bus become free. Otherwise, user may writing 1 to ARBLOIF (UI2C_PROTSTS [11]) and set (PTRG, STA, STO, AA) = (1, 0, 0, X) to release I²C bus and enter not addressed Slave mode.

Slave Mode

When reset, I²C is not addressed and will not recognize the address on I²C bus. User can set device address by UI2C_DEVADDRn and set (PTRG, STA, STO, AA) = (1, 0, 0, 1) to let I²C recognize the address sent by master. Figure 6.20-17 shows all the possible flow for I²C in Slave mode. Users need to follow a proper flow (as shown in Figure 6.20-17 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) or SLA+R (Master want to read data from Slave) after arbitration lost, the ARBLOIF will be set to 1.

The I²C controller supports two slave address match flags, are ADMAT0 and ADMAT1 on UI2C_ADMAT[1:0] register. Every control register represent which address is used and set 1 to inform software.

Note: During I²C communication, the SCL clock will be released when writing '1' to PTRG (UI2C_PROTCTL [5]) in Slave mode.

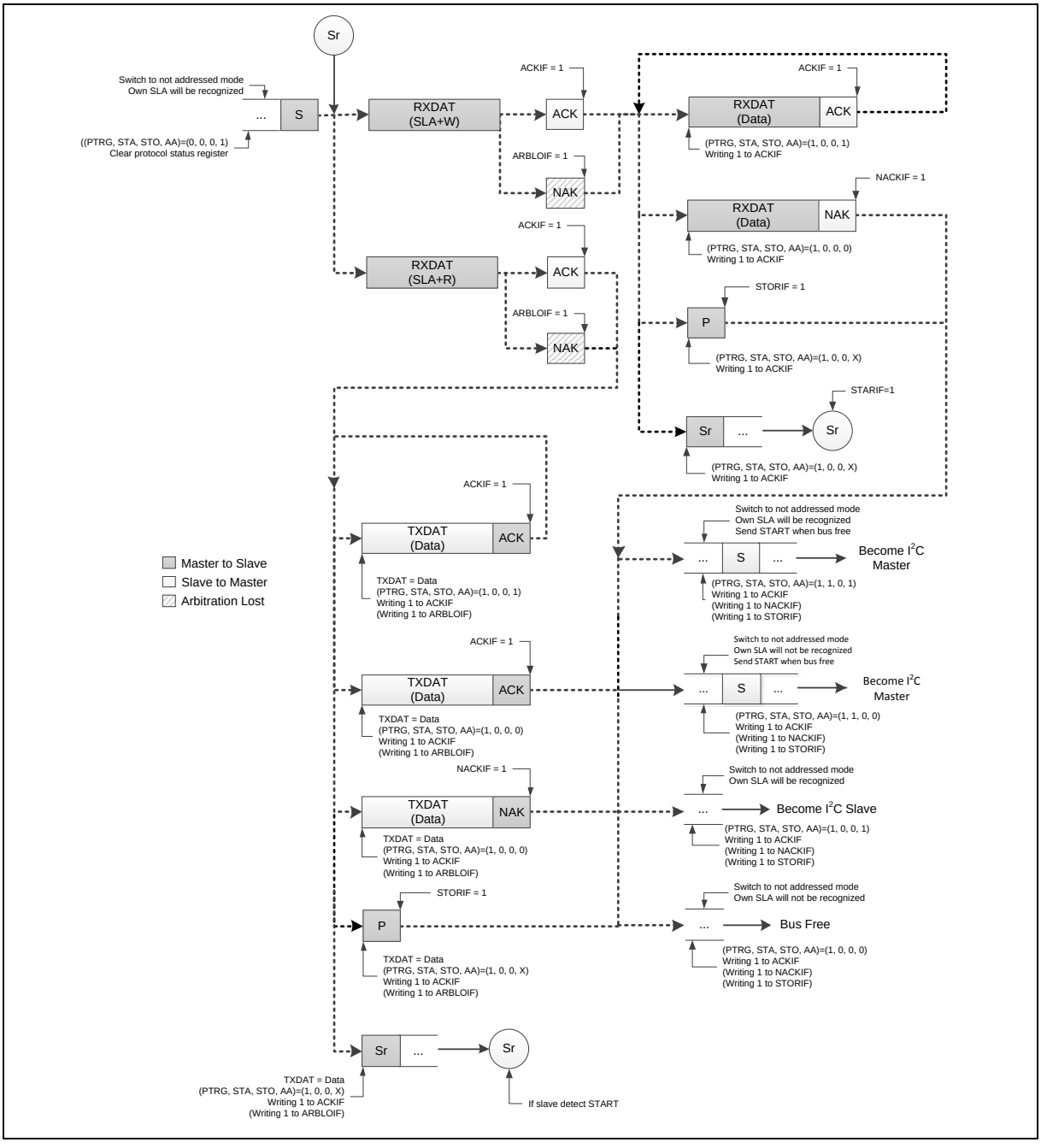


Figure 6.20-17 Save Mode Control Flow with 7-bit Address

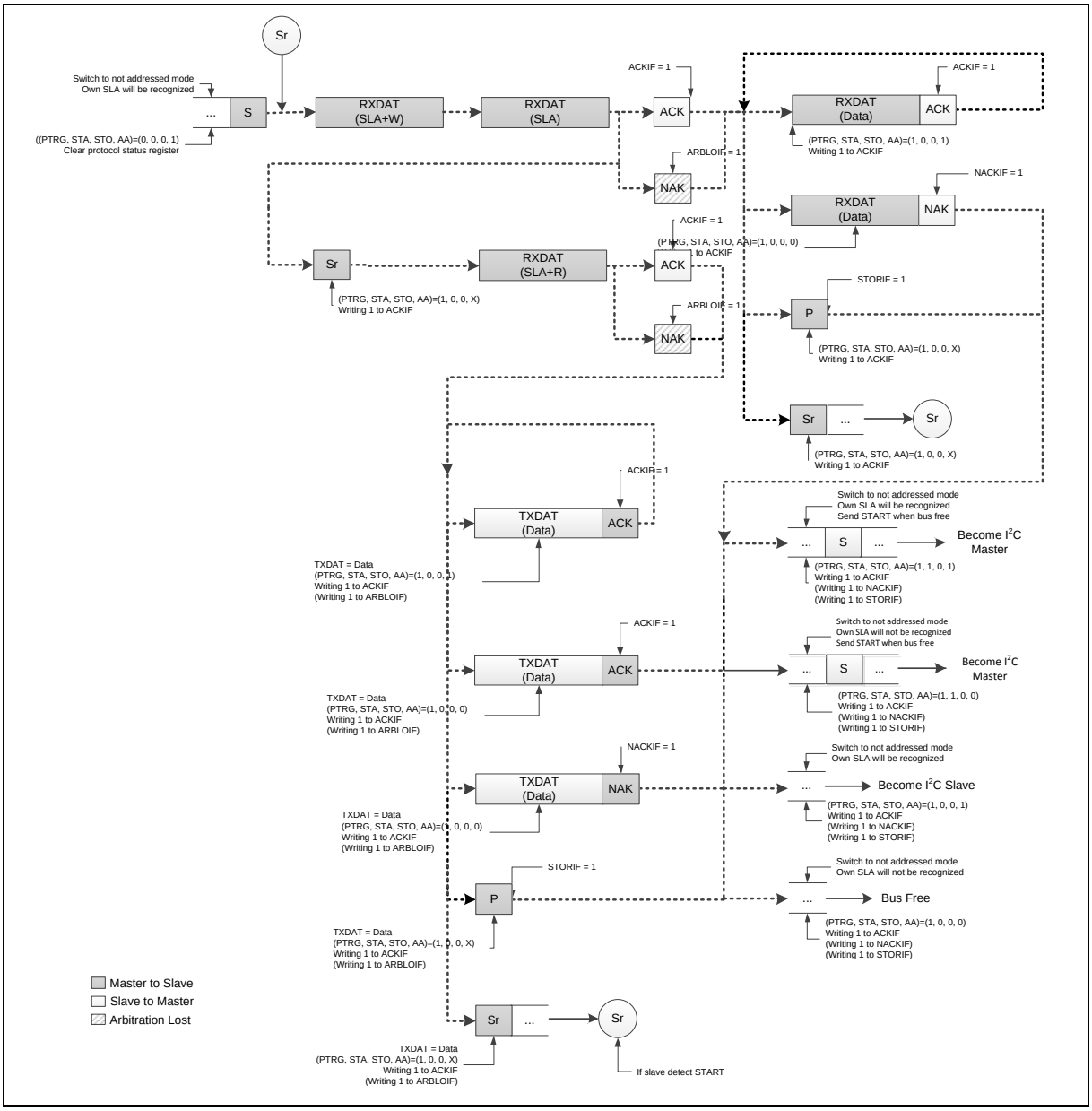


Figure 6.20-18 Save Mode Control Flow with 10-bit Address

If I^{2}C is still transmitting and receiving data in addressed Slave mode but got a STOP or Repeat START, the register STORIF (UI2C_PROTSTS [9]) or STARIF (UI2C_PROTSTS [8]) will be set. User could follow the action for NACKIF (UI2C_PROTSTS [10]) as shown in the above figure when got STARIF (UI2C_PROTSTS [8]) is set.

Note: After slave gets interrupt flag of NACKIF (UI2C_PROTSTS [10]) and start/stop symbol including STARIF (UI2C_PROTSTS [8]) and STORIF (UI2C_PROTSTS [9]), slave can switch to not address mode and own SLA will not be recognized. If setting this interrupt flag, slave will not receive any I^{2}C signal or address from master. At this status, I^{2}C should be reset by setting FUNMODE (UI2C_CTL [2:0]) = 000B to leave this status.

General Call (GC) Mode

If the GCFUNC bit (UI2C_PROTCTL [0]) is set, the I^{2}C port hardware will respond to General Call address (00H). User can clear GC bit to disable general call function. When the GC bit is set and the I^{2}C in slave mode, it can receive the general call address by 0x00 after master send general call

address to I²C bus, and then it also will follow protocol status register.

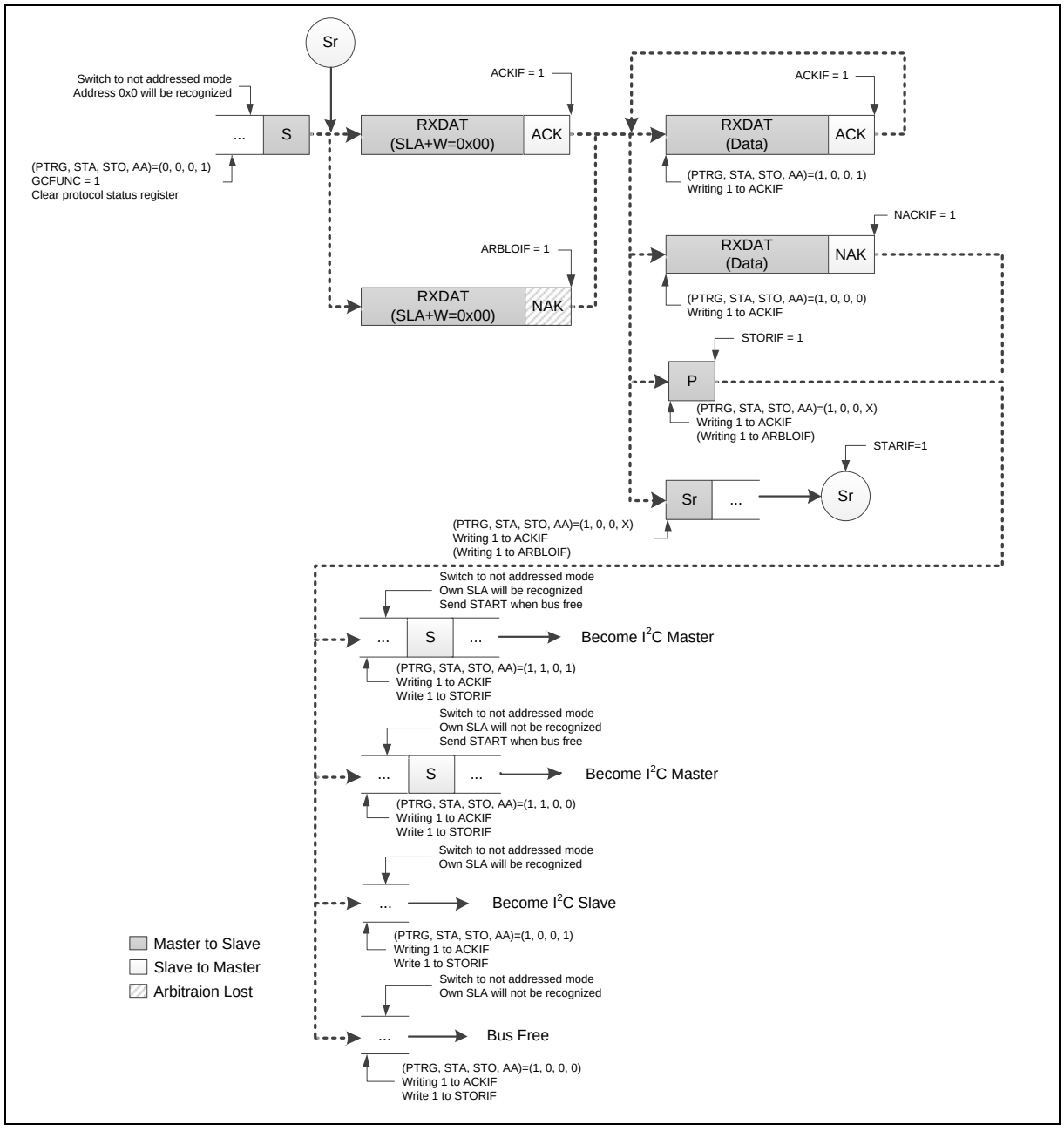


Figure 6.20-19 GC Mode with 7-bit Address

If I²C is still receiving data in GC mode but got a STOP or Repeat START, the STORIF (UI2C_PROTSTS [9]) or STARIF (UI2C_PROTSTS [8]) will be set. User could follow the action for NACKIF (UI2C_PROTSTS [10]) in above figure when got STORIF (UI2C_PROTSTS [9]) or STARIF (UI2C_PROTSTS [8]) is set.

Note: After slave gets interrupt flag of NACKIF (UI2C_PROTSTS [10]) and start/stop symbol including STARIF (UI2C_PROTSTS [8]) and STORIF (UI2C_PROTSTS [9]), slave can switch to not address mode and own SLA will not be recognized. If setting this interrupt flag, slave will not receive any I²C signal or address from master. At this time, I²C controller should be reset by setting FUNMODE (UI2C_CTL [2:0]) = 000B to leave this status.

Protocol Functional Description

Monitor Mode

When I^{2}C enters monitor mode, this device always returns NACK to master after each frame reception even address matching. Moreover, this device will store any receive data including address, command code, and data.

Interrupt in Monitor Mode

All interrupts will occur as normal process when the MONEN (UI2C_PROTCTL [9]) is set. Note that the first interrupt will occur when initial START, it not the same as I^{2}C slave, but the other interrupts are the same.

Subsequent to the address-match detection, interrupts will be generated after each data byte is received as slave mode control flow, or after each byte that the module believes it has transmitted for a slave-read transfer. In this second case, the data register will actually contain data transmitted by some other slave on the bus which was actually addressed by the master. If user wants to watch other device, user can set address mask and monitor.

If the monitor has not had time to respond to interrupt, the SCL signal will be pulled to low when SCLOUTEN (UI2C_PROTCTL [8]) is set to 1. User must set PTRG (UI2C_PROTCTL [5]) to release bus when SCLOUTEN (UI2C_PROTCTL [8]) is set to 1. If SCLOUTEN (UI2C_PROTCTL [8]) is not set to 1, user doesn't need to set PTRG (UI2C_PROTCTL [5]) to 1.

When device address match, but the device response NACK, this address will be received into buffer and NACK interrupt will be generated.

Following all of these interrupts, the processor may read the data register to see what was actually transmitted on the bus.

Loss of Arbitration in Monitor Mode

In monitor mode, the I^{2}C module will not be able to respond to a request for information by the bus master or issue an ACK. Some other slave on the bus will respond instead. Software should be aware of the fact that the module is in monitor mode and should not respond to any loss of arbitration state that is detected.

Programmable Setup and Hold Time

In order to guarantee a correct data setup and hold time, the timing must be configured. By programming HTCTL (UI2C_TMCTL[24:16]) to configure hold time and STCTL (UI2C_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 I^{2}C protocol. Once setup time configuration greater than design limitation, that means if setup time setting make SCL output less than three PCLKs, I^{2}C controller can't work normally due to SCL must sample three times. And once hold time configuration greater than I^{2}C clock limitation, I^{2}C will occur bus error. Suggest that user calculate suitable timing with baud rate and protocol before setting timing. Table 6.20-1 shows the relationship between I^{2}C baud rate and PCLK, the number of table represent one clock duty contain how many PCLKs. Setup and hold time configuration even can program some extreme values in the design, but user should follow I^{2}C protocol standard.

I^{2}C 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 6.20-1 Relationship between I²C Baud Rate and PCLK

For setup time wrong adjustment example, assuming one SCL cycle contains ten PCLKs and set STCTL (UI2C_TMCTL[8:0]) to 3 that stretch three PCLKs for setup time setting. The setup time setting limitation: ST_{limit} = (UI2C_BRGEN[25:16]+1) - 6.

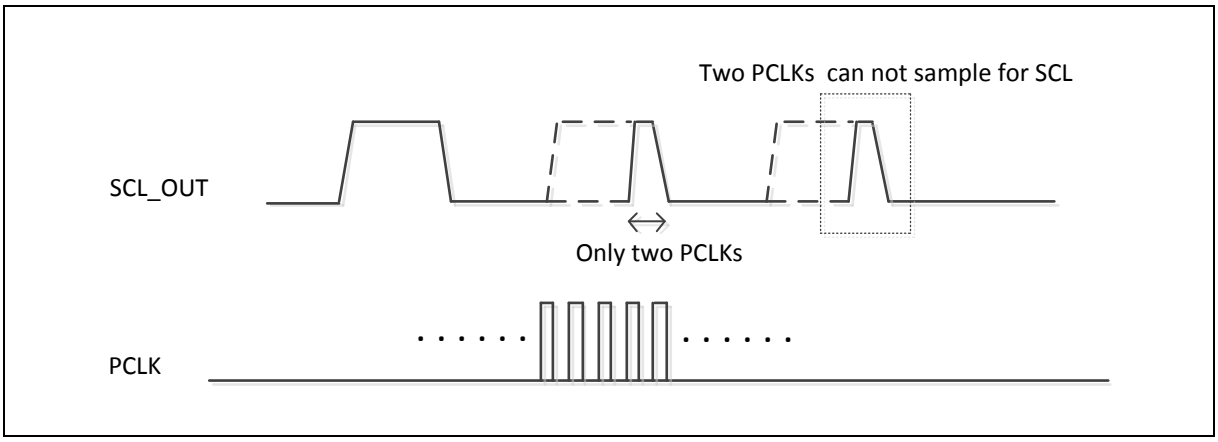


Figure 6.20-20 Setup Time Wrong Adjustment

For hold time wrong adjustment example, use I²C Baud Rate = 1200k and PCLK = 72 MHz, the SCL high/low duty = 60 PCLK. When HTCTL (UI2C_TMCTL[24:16]) is set to 63 and STCTL (UI2C_TMCTL[8:0]) is set to 0, then SDA output delay will over SCL high duty and cause bus error. The hold time setting limitation: HT_{limit} = (UI2C_BRGEN[25:16]+1) - 9.

Note: Hold time adjust function can only work in master mode, when slave mode, the USCI-I²C HTCTL (UI2C_TMCTL[24:16]) should set as 0.

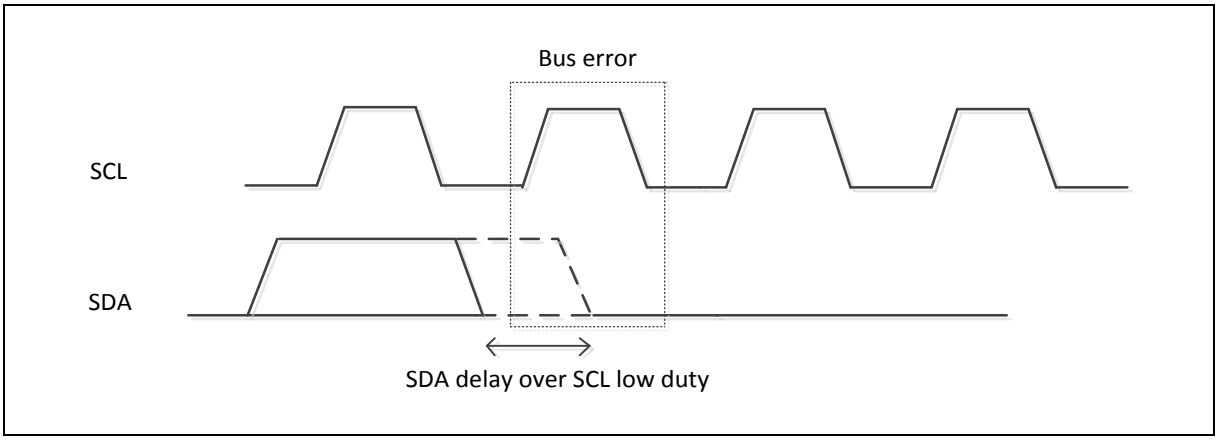


Figure 6.20-21 Hold Time Wrong Adjustment

I²C Time-out Function

There is a 10 bits time-out counter TOCNT (UI2C_PROTCTL [25:16]) 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 equals TOCNT (UI2C_PROTCTL [25:16]) and generates I²C interrupt to CPU or stops counting by clearing TOIEN (UI2C_PROTIEN [0]) to 0 or setting all I²C interrupt signal (ACKIF, ERRIF, ARBLOIF, NACKIF, STORIF, STARIF). User may write 1 to clear TOIF(UI2C_PROTSTS[5]) to 0. When time-out counter is enabled, writing 1 to the TOIF will reset counter and re-start up counting after TOIF is cleared. Refer to Figure 6.20-22 for the time-out counter TOCNT (UI2C_PROTCTL [25:16]). T_{TOCNT} = (TOCNT (UI2C_PROTCTL [25:16]) +1) x32 (5-bit) x T_{PCLK}. Note that the time counter clock source TMCNTSRC

(UI2C_ BRGEN [5]) must be set as 0.

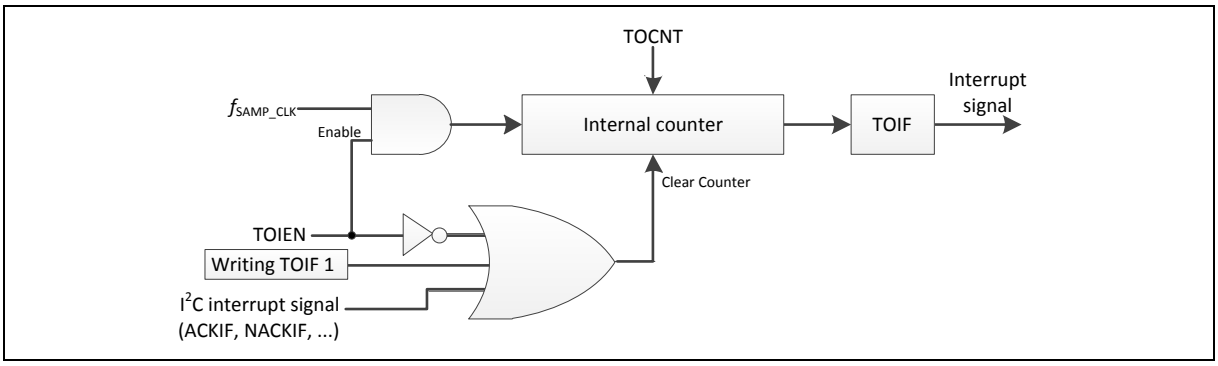


Figure 6.20-22 I²C Time-out Count Block Diagram

Wake-up Function

When chip enters Power-down mode and set WKEN (UI2C_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. The SCL is stretched until WKAKDONE bit is clear by user. If the frequency of SCL is low speed and the system has wakeup from address match frame, the user shall check this bit to confirm this frame has transaction done and then to do the wake-up procedure. Therefore, when the chip is woken up by address match with one of the device address register (UI2C_DEVADDRn), the user shall check the WKAKDONE (UI2C_PROTSTS [16]) bit is set to 1 to confirm the address wakeup frame has done. The WKAKDONE bit indicates that the ACK bit cycle of address match frame is done in power-down. Note that user must clear WKUPIF after clearing the WKAKDONE bit to 0.

The WRSTSWK (UI2C_PROTSTS [17]) bit records the Read/Write command on the address match wake-up frame. The user can use 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.

When system is woken up by other I²C master device, WKF (UI2C_WKSTS [0]) is set to indicate this event. User needs write "1" to clear this bit.

Example for Random Read on EEPROM

The following steps are used to configure the USCI0_I2C related registers when using I²C protocol to read data from EEPROM.

1. Set USCI0_I2C the multi-function pin as SCL and SDA pins. The multi-function configuration reference Basic Configuration.
2. Enable USCI0 APB clock. The multi-function configuration reference Basic Configuration.
3. Set USCI0RST=1 to reset USCI controller then set USCI0RST=0 let USCI controller to normal operation. The reset controller configuration reference Basic Configuration.
4. Set FUNMODE =100 to enable USCI0_I2C controller in the "UI2C_CTL" register.
5. Give USCI0_I2C clock a divided register value for USCI0_I2C clock rate in the "UI2C_BRGEN".
6. Enable system USCI0 IRQ in system "NVIC" control register.
7. Set ACKIEN, ERRIEN, ARBLOIEN, NACKIEN, STORIEN, STARIEN, and TOIEN to enable I²C Interrupt in the "UI2C_PROTIEN" register.
8. Set USCI address registers "UI2C_DEVADDR0 ~ UI2C_DEVADDR1".

Random read operation is one of the methods of access EEPROM. The method allows the master to

access any address of EEPROM space. Figure 6.20-23 shows the EEPROM random read operation.

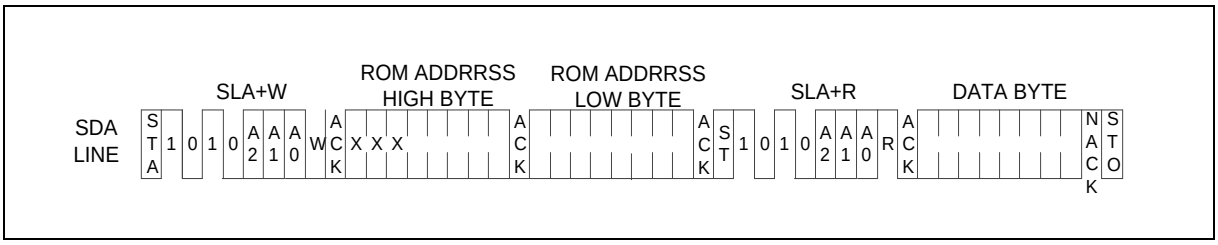


Figure 6.20-23 EEPROM Random Read

Figure 6.20-24 shows how to use I^{2}C controller to implement the protocol of EEPROM random read.

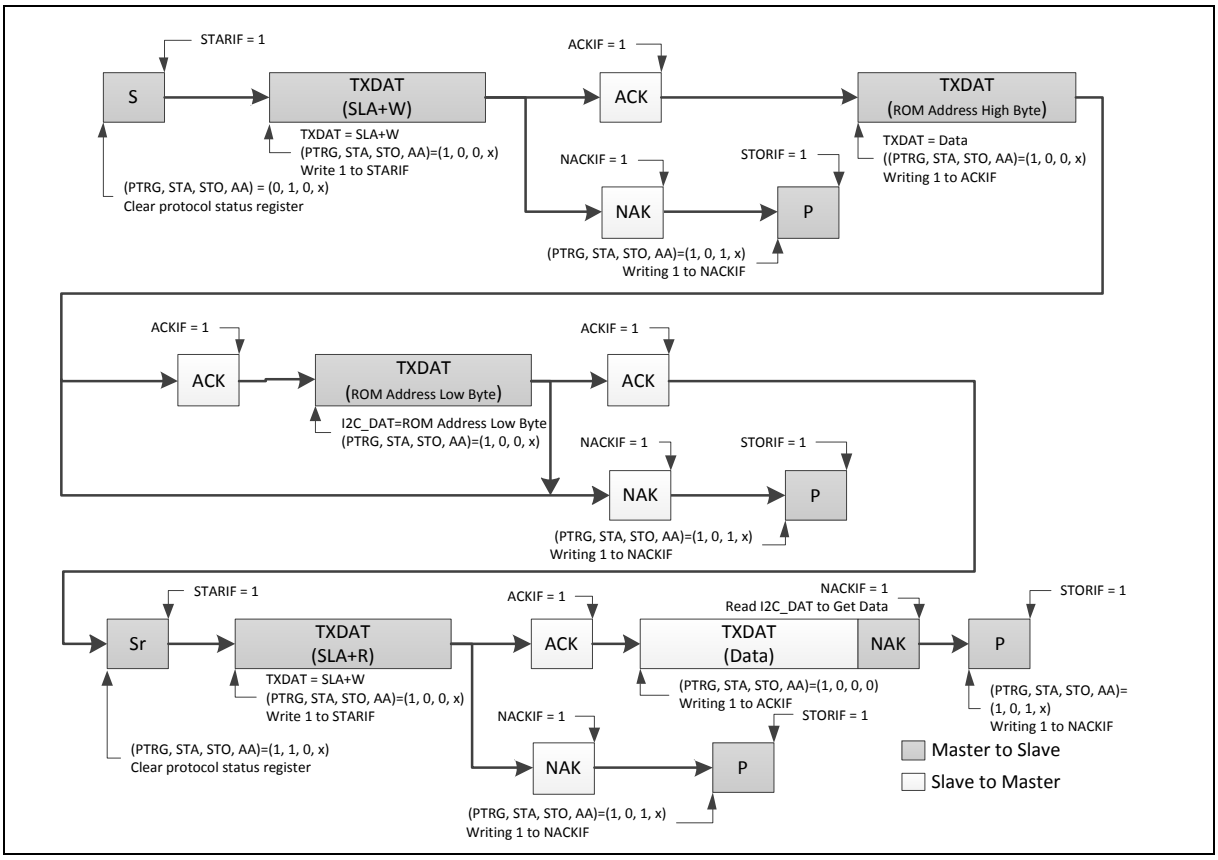


Figure 6.20-24 Protocol of EEPROM Random Read

The I^{2}C controller, which is a master, sends START to bus. Then, it sends a SLA+W (Slave address + Write bit) to EERPOM followed by two bytes data address to set the EEPROM address to read. Finally, a Repeat START followed by SLA+R is sent to read the data from EEPROM.

6.20.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
UI2C_I2C Base Address: **UI2Cn_BA = 0x400D_0000 + (0x1000 * n)** **n= 0, 1**				
UI2C_CTL	UI2Cn_BA+0x00	R/W	USCI Control Register	0x0000_0000
UI2C_BRGEN	UI2Cn_BA+0x08	R/W	USCI Baud Rate Generator Register	0x0000_3C00
UI2C_LINECTL	UI2Cn_BA+0x2C	R/W	USCI Line Control Register	0x0000_0000
UI2C_TXDAT	UI2Cn_BA+0x30	W	USCI Transmit Data Register	0x0000_0000
UI2C_RXDAT	UI2Cn_BA+0x34	R	USCI Receive Data Register	0x0000_0000
UI2C_DEVADDR0	UI2Cn_BA+0x44	R/W	USCI Device Address Register 0	0x0000_0000
UI2C_DEVADDR1	UI2Cn_BA+0x48	R/W	USCI Device Address Register 1	0x0000_0000
UI2C_ADDRMSK0	UI2Cn_BA+0x4C	R/W	USCI Device Address Mask Register 0	0x0000_0000
UI2C_ADDRMSK1	UI2Cn_BA+0x50	R/W	USCI Device Address Mask Register 1	0x0000_0000
UI2C_WKCTL	UI2Cn_BA+0x54	R/W	USCI Wake-up Control Register	0x0000_0000
UI2C_WKSTS	UI2Cn_BA+0x58	R/W	USCI Wake-up Status Register	0x0000_0000
UI2C_PROTCTL	UI2Cn_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0000
UI2C_PROTIEN	UI2Cn_BA+0x60	R/W	USCI Protocol Interrupt Enable Register	0x0000_0000
UI2C_PROTSTS	UI2Cn_BA+0x64	R/W	USCI Protocol Status Register	0x0000_0000
UI2C_ADMAT	UI2Cn_BA+0x88	R/W	I^2C Slave Match Address Register	0x0000_0000
UI2C_TMCTL	UI2Cn_BA+0x8C	R/W	I^2C Timing Configure Control Register	0x0000_0000

6.20.7 Register Description

USCI Control Register (UI2C_CTL)

Register	Offset	R/W	Description	Reset Value
UI2C_CTL	UI2Cn_BA+0x00	R/W	USCI Control Register	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**FUNMODE**	**Function Mode** This bit field selects the protocol for this USCI controller. Selecting a protocol that is not available or a reserved combination disables the USCI. When switching between two protocols, the USCI has to be disabled before selecting a new protocol. Simultaneously, the USCI will be reset when user write 000 to FUNMODE. 000 = The USCI is disabled. All protocol related state machines are set to idle state. 001 = The SPI protocol is selected. 010 = The UART protocol is selected. 100 = The I^2C protocol is selected. **Note:** Other bit combinations are reserved.

USCI Baud Rate Generator Register (UI2C_BRGEN)

Register	Offset	R/W	Description	Reset Value
UI2C_BRGEN	UI2Cn_BA+0x08	R/W	USCI Baud Rate Generator Register	0x0000_3C00

31	30	29	28	27	26	25	24
Reserved						CLKDIV	
23	22	21	20	19	18	17	16
CLKDIV							
15	14	13	12	11	10	9	8
Reserved	DSCNT					PDSCNT	
7	6	5	4	3	2	1	0
Reserved		TMCNTSRC	TMCNTEN	SPCLKSEL		PTCLKSEL	RCLKSEL

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25:16]	**CLKDIV**	**Clock Divider** This bit field defines the ratio between the protocol clock frequency f_{PROT_CLK} and the clock divider frequency f_{DIV_CLK} (fDIV_CLK = f_{PROT_CLK} / (CLKDIV+1)).
[15]	**Reserved**	Reserved.
[14:10]	**DSCNT**	**Denominator for Sample Counter** This bit field defines the divide ratio of the sample clock f_{SAMP_CLK}. The divided frequency f_{DS_CNT} = f_{PDS_CNT} / (DSCNT+1).
[9:8]	**PDSCNT**	**Pre-divider for Sample Counter** This bit field defines the divide ratio of the clock division from sample clock f_{SAMP_CLK}. The divided frequency f_{PDS_CNT} = f_{SAMP_CLK} / (PDSCNT+1).
[7:6]	**Reserved**	Reserved.
[5]	**TMCNTSRC**	**Time Measurement Counter Clock Source Selection** 0 = Time measurement counter with f_{PROT_CLK}. 1 = Time measurement counter with f_{DIV_CLK}.
[4]	**TMCNTEN**	**Time Measurement Counter Enable Bit** This bit enables the 10-bit timing measurement counter. 0 = Time measurement counter is Disabled. 1 = Time measurement counter is Enabled.
[3:2]	**SPCLKSEL**	**Sample Clock Source Selection** This bit field used for the clock source selection of a sample clock (f_{SAMP_CLK}) for the protocol processor. 00 = f_{SAMP_CLK} = f_{DIV_CLK}. 01 = f_{SAMP_CLK} = f_{PROT_CLK}. 10 = f_{SAMP_CLK} = f_{SCLK}. 11 = f_{SAMP_CLK} = f_{REF_CLK}.

[1]	**PTCLKSEL**	**Protocol Clock Source Selection** This bit selects the source signal of protocol clock (f_{PROT_CLK}). 0 = Reference clock f_{REF_CLK}. 1 = f_{REF_CLK2} (its frequency is half of f_{REF_CLK}).
[0]	**RCLKSEL**	**Reference Clock Source Selection** This bit selects the source signal of reference clock (f_{REF_CLK}). 0 = Peripheral device clock f_{PCLK}. 1 = Reserved.

USCI Line Control Register (UI2C_LINECTL)

Register	Offset	R/W	Description	Reset Value
UI2C_LINECTL	UI2Cn_BA+0x2C	R/W	USCI Line Control Register	0x0000_0000

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

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:8]	**DWIDTH**	**Word Length of Transmission** This bit field defines the data word length (amount of bits) for reception and transmission. The data word is always right-aligned in the data buffer. USCI support word length from 4 to 16 bits. 0x0: The data word contains 16 bits located at bit positions [15:0]. 0x1: Reserved. 0x2: Reserved. 0x3: Reserved. 0x4: The data word contains 4 bits located at bit positions [3:0]. 0x5: The data word contains 5 bits located at bit positions [4:0]. ... 0xF: The data word contains 15 bits located at bit positions [14:0].
[7:1]	**Reserved**	Reserved.
[0]	**LSB**	**LSB First Transmission Selection** 0 = The MSB, which bit of transmit/receive data buffer depends on the setting of DWIDTH, is transmitted/received first. 1 = The LSB, the bit 0 of data buffer, will be transmitted/received first.

USCI Transmit Data Register (UI2C_TXDAT)

Register	Offset	R/W	Description	Reset Value
UI2C_TXDAT	UI2Cn_BA+0x30	W	USCI Transmit Data Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**TXDAT**	**Transmit Data** Software can use this bit field to write 16-bit transmit data for transmission.

USCI Receive Data Register (UI2C_RXDAT)

Register	Offset	R/W	Description	Reset Value
UI2C_RXDAT	UI2Cn_BA+0x34	R	USCI Receive Data Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**RXDAT**	**Received Data** This bit field monitors the received data which stored in receive data buffer. **Note:** In I^2C protocol, RXDAT[12:8] indicate the different transmission conditions which defined in I^2C.

USCI Device Address Register (UI2C_DEVADDR)

Register	Offset	R/W	Description	Reset Value
UI2C_DEVADDR0	UI2Cn_BA+0x44	R/W	USCI Device Address Register 0	0x0000_0000
UI2C_DEVADDR1	UI2Cn_BA+0x48	R/W	USCI Device Address Register 1	0x0000_0000

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

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:0]	**DEVADDR**	**Device Address** In I^2C protocol, this bit field contains the programmed slave address. If the first received address byte are 1111 $0AAX_B$, the AA bits are compared to the bits DEVADDR[9:8] to check for address match, where the X is R/W bit. Then the second address byte is also compared to DEVADDR[7:0]. **Note 1:** The DEVADDR [9:7] must be set 3'b000 when I^2C operating in 7-bit address mode. **Note 2:** When software set 10'h000, the address can not be used.

USCI Device Address Mask Register (UI2C_ADDRMSK) – for I²C Only

Register	Offset	R/W	Description	Reset Value
UI2C_ADDRMSK0	UI2Cn_BA+0x4C	R/W	USCI Device Address Mask Register 0	0x0000_0000
UI2C_ADDRMSK1	UI2Cn_BA+0x50	R/W	USCI Device Address Mask Register 1	0x0000_0000

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

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:0]	**ADDRMSK**	**USCI Device Address Mask** 0 = Mask Disabled (the received corresponding register bit should be exact the same as address register.). 1 = Mask Enabled (the received corresponding address bit is don't care.). USCI support multiple address recognition with two address mask register. When the bit in the address mask register is set to one, it means the received corresponding address bit is don't-care. If the bit is set to zero, that means the received corresponding register bit should be exact the same as address register.

USCI Wake-up Control Register (UI2C_WKCTL)

Register	Offset	R/W	Description	Reset Value
UI2C_WKCTL	UI2Cn_BA+0x54	R/W	USCI Wake-up Control Register	0x0000_0000

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

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**WKADDREN**	**Wake-up Address Match Enable Bit** 0 = The chip is woken up according receive ‘START’ symbol. 1 = The chip is woken up according address match.
[0]	**WKEN**	**Wake-up Enable Bit** 0 = Wake-up function Disabled. 1 = Wake-up function Enabled.

USCI Wake-up Status Register (UI2C_WKSTS)

Register	Offset	R/W	Description	Reset Value
UI2C_WKSTS	UI2Cn_BA+0x58	R/W	USCI Wake-up Status Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**WKF**	**Wake-up Flag** When chip is woken up from Power-down mode, this bit is set to 1. Software can write 1 to clear this bit.

USCI Protocol Control Register – I²C (UI2C_PROTCTL)

Register	Offset	R/W	Description	Reset Value
UI2C_PROTCTL	UI2Cn_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PROTEN	Reserved					TOCNT	
23	22	21	20	19	18	17	16
TOCNT							
15	14	13	12	11	10	9	8
Reserved						MONEN	SCLOUTEN
7	6	5	4	3	2	1	0
Reserved		PTRG	ADDR10EN	STA	STO	AA	GCFUNC

Bits	Description	
[31]	**PROTEN**	**I²C Protocol Enable Bit** 0 = I²C Protocol Disabled. 1 = I²C Protocol Enabled.
[30:26]	**Reserved**	Reserved.
[25:16]	**TOCNT**	**Time-out Clock Cycle** This bit field indicates how many clock cycle selected by TMCNTSRC (UI2C_BRGEN [5]) when each interrupt flags are clear. The time-out is enable when TOCNT bigger than 0. **Note:** The TMCNTSRC (UI2C_BRGEN [5]) must be set zero on I²C mode.
[15:10]	**Reserved**	Reserved.
[9]	**MONEN**	**Monitor Mode Enable Bit** This bit enables monitor mode. In monitor mode the SDA output will be put in high impedance mode. This prevents the I²C module from outputting data of any kind (including ACK) onto the I²C data bus. 0 = The monitor mode Disabled. 1 = The monitor mode Enabled. **Note:** Depending on the state of the SCLOUTEN bit, the SCL output may be also forced high, preventing the module from having control over the I²C clock line.
[8]	**SCLOUTEN**	**SCL Output Enable Bit** This bit enables monitor pulling SCL to low. This monitor will pull SCL to low until it has had time to respond to an I²C interrupt. 0 = SCL output will be forced high due to open drain mechanism. 1 = I²C module may act as a slave peripheral just like in normal operation, the I²C holds the clock line low until it has had time to clear I²C interrupt.
[7:6]	**Reserved**	Reserved.

[5]	**PTRG**	**I²C Protocol Trigger (Write Only)** When a new state is present in the UI2C_PROTSTS register, if the related interrupt enable bits are set, the I²C interrupt is requested. It must write one by software to this bit after the related interrupt flags are set to 1 and the I²C protocol function will go ahead until the STOP is active or the PROTIEN is disabled. 0 = I2C's stretch disabled and the I²C protocol function will go ahead. 1 = I2C's stretch active.
[4]	**ADDR10EN**	**Address 10-bit Function Enable Bit** 0 = Address match 10 bit function Disabled. 1 = Address match 10 bit function Enabled.
[3]	**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.
[2]	**STO**	**I²C STOP Control** In Master mode, setting STO to transmit a STOP condition to bus then I²C hardware will check the bus condition if a STOP condition is detected this bit will be cleared by hardware automatically. In a slave mode, setting STO resets I²C hardware to the defined "not addressed" slave mode when bus error (UI2C_PROTSTS.ERRIF = 1).
[1]	**AA**	**Assert Acknowledge Control** When AA =1 prior to address or data received, an acknowledged (low level to SDA) will be returned during the acknowledge clock pulse on the SCL line when 1.) A slave is acknowledging the address sent from master, 2.) The receiver devices are acknowledging the data sent by transmitter. When AA=0 prior to address or data received, a Not acknowledged (high level to SDA) will be returned during the acknowledge clock pulse on the SCL line.
[0]	**GCFUNC**	**General Call Function** 0 = General Call Function Disabled. 1 = General Call Function Enabled.

USCI Protocol Interrupt Enable Register – I²C (UI2C_PROTIEN)

Register	Offset	R/W	Description	Reset Value
UI2C_PROTIEN	UI2Cn_BA+0x60	R/W	USCI Protocol Interrupt Enable Register	0x0000_0000

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

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**ACKIEN**	**Acknowledge Interrupt Enable Bit** This bit enables the generation of a protocol interrupt if an acknowledge is detected by a master. 0 = The acknowledge interrupt Disabled. 1 = The acknowledge interrupt Enabled.
[5]	**ERRIEN**	**Error Interrupt Enable Bit** This bit enables the generation of a protocol interrupt if an I²C error condition is detected (indicated by ERRIF (UI2C_PROTSTS [12])). 0 = The error interrupt Disabled. 1 = The error interrupt Enabled.
[4]	**ARBLOIEN**	**Arbitration Lost Interrupt Enable Bit** This bit enables the generation of a protocol interrupt if an arbitration lost event is detected. 0 = The arbitration lost interrupt Disabled. 1 = The arbitration lost interrupt Enabled.
[3]	**NACKIEN**	**Non - Acknowledge Interrupt Enable Bit** This bit enables the generation of a protocol interrupt if a Non - acknowledge is detected by a master. 0 = The non - acknowledge interrupt Disabled. 1 = The non - acknowledge interrupt Enabled.
[2]	**STORIEN**	**STOP Condition Received Interrupt Enable Bit** This bit enables the generation of a protocol interrupt if a STOP condition is detected. 0 = The stop condition interrupt Disabled. 1 = The stop condition interrupt Enabled.
[1]	**STARIEN**	**START Condition Received Interrupt Enable Bit** This bit enables the generation of a protocol interrupt if a START condition is detected.

		0 = The start condition interrupt Disabled. 1 = The start condition interrupt Enabled.
[0]	**TOIEN**	**Time-out Interrupt Enable Bit** In I^2C protocol, this bit enables the interrupt generation in case of a time-out event. 0 = The time-out interrupt Disabled. 1 = The time-out interrupt Enabled.

USCI Protocol Status Register – I²C (UI2C_PROTSTS)

Register	Offset	R/W	Description	Reset Value
UI2C_PROTSTS	UI2Cn_BA+0x64	R/W	USCI Protocol Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				ERRARBLO	BUSHANG	WRSTSWK	WKAKDONE
15	14	13	12	11	10	9	8
SLAREAD	SLASEL	ACKIF	ERRIF	ARBLOIF	NACKIF	STORIF	STARIF
7	6	5	4	3	2	1	0
Reserved	ONBUSY	TOIF	Reserved				

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19]	**ERRARBLO**	**Error Arbitration Lost** This bit indicates bus arbitration lost due to bigger noise which is can't be filtered by input processor. The I²C can send start condition when ERRARBLO is set. Thus this bit doesn't be cared on slave mode. 0 = The bus is normal status for transmission. 1 = The bus is error arbitration lost status for transmission. **Note:** This bit has no interrupt signal, and it will be cleared automatically by hardware when a START condition is present.
[18]	**BUSHANG**	**Bus Hang-up** This bit indicates bus hang-up status. There is 4-bit counter count when SCL hold high and refer f_{SAMP_CLK}. The hang-up counter will count to overflow and set this bit when SDA is low. The counter will be reset by falling edge of SCL signal. 0 = The bus is normal status for transmission. 1 = The bus is hang-up status for transmission. **Note:** This bit has no interrupt signal, and it will be cleared automatically by hardware when a START condition is present.
[17]	**WRSTSWK**	**Read/Write Status Bit in Address Wake-up Frame** 0 = Write command be record on the address match wake-up frame. 1 = Read command be record on the address match wake-up frame.
[16]	**WKAKDONE**	**Wake-up Address Frame Acknowledge Bit Done** 0 = The ACK bit cycle of address match frame isn't done. 1 = The ACK bit cycle of address match frame is done in power-down. **Note:** This bit can't release when WKUPIF is set.
[15]	**SLAREAD**	**Slave Read Request Status** This bit indicates that a slave read request has been detected. 0 = A slave R/W bit is 1 has not been detected. 1 = A slave R/W bit is 1 has been detected.

		Note: This bit has no interrupt signal, and it will be cleared automatically by hardware.
[14]	**SLASEL**	**Slave Select Status** This bit indicates that this device has been selected as slave. 0 = The device is not selected as slave. 1 = The device is selected as slave. **Note:** This bit has no interrupt signal, and it will be cleared automatically by hardware.
[13]	**ACKIF**	**Acknowledge Received Interrupt Flag** This bit indicates that an acknowledge has been received in master mode. A protocol interrupt can be generated if UI2C_PROTCTL.ACKIEN = 1. 0 = An acknowledge has not been received. 1 = An acknowledge has been received. **Note:** It is cleared by software writing 1 into this bit
[12]	**ERRIF**	**Error Interrupt Flag** This bit indicates that a Bus Error occurs when a START or STOP condition is present at an illegal position in the formation frame. Example of illegal position are during the serial transfer of an address byte, a data byte or an acknowledge bit. A protocol interrupt can be generated if UI2C_PROTCTL.ERRIEN = 1. 0 = An I^2C error has not been detected. 1 = An I^2C error has been detected. **Note 1:** It is cleared by software writing 1 into this bit **Note 2:** This bit is set for slave mode, and user must write 1 into STO register to the defined “not addressed” slave mode.
[11]	**ARBLOIF**	**Arbitration Lost Interrupt Flag** This bit indicates that an arbitration has been lost. A protocol interrupt can be generated if UI2C_PROTCTL.ARBLOIEN = 1. 0 = An arbitration has not been lost. 1 = An arbitration has been lost. **Note:** It is cleared by software writing 1 into this bit
[10]	**NACKIF**	**Non - Acknowledge Received Interrupt Flag** This bit indicates that a non - acknowledge has been received in master mode. A protocol interrupt can be generated if UI2C_PROTCTL.NACKIEN = 1. 0 = A non - acknowledge has not been received. 1 = A non - acknowledge has been received. **Note:** It is cleared by software writing 1 into this bit
[9]	**STORIF**	**Stop Condition Received Interrupt Flag** This bit indicates that a stop condition has been detected on the I^2C bus lines. A protocol interrupt can be generated if UI2C_PROTCTL.STORIEN = 1. 0 = A stop condition has not yet been detected. 1 = A stop condition has been detected. **Note 1:** It is cleared by software writing 1 into this bit
[8]	**STARIF**	**Start Condition Received Interrupt Flag** This bit indicates that a start condition or repeated start condition has been detected on master mode. However, this bit also indicates that a repeated start condition has been detected on slave mode. A protocol interrupt can be generated if UI2C_PROTCTL.STARIEN = 1. 0 = A start condition has not yet been detected. 1 = A start condition has been detected. **Note:** It is cleared by software writing 1 into this bit
[7]	**Reserved**	Reserved.

[6]	**ONBUSY**	**On Bus Busy** Indicates that a communication is in progress on the bus. It is set by hardware when a START condition is detected. It is cleared by hardware when a STOP condition is detected 0 = The bus is IDLE (both SCLK and SDA High). 1 = The bus is busy.
[5]	**TOIF**	**Time-out Interrupt Flag** 0 = A time-out interrupt status has not occurred. 1 = A time-out interrupt status has occurred. **Note:** It is cleared by software writing 1 into this bit
[4:0]	**Reserved**	Reserved.

USCI Slave Match Address Register (UI2C_ADMAT)

Register	Offset	R/W	Description	Reset Value
UI2C_ADMAT	UI2Cn_BA+0x88	R/W	I^2C Slave Match Address Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**ADMAT1**	**ADMAT0**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**ADMAT1**	**USCI Address 1 Match Status Register** When address 1 is matched, hardware will inform which address used. This bit will set to 1, and software can write 1 to clear this bit.
[0]	**ADMAT0**	**USCI Address 0 Match Status Register** When address 0 is matched, hardware will inform which address used. This bit will set to 1, and software can write 1 to clear this bit.

USCI Timing Configure Control Register (UI2C_TMCTL)

Register	Offset	R/W	Description	Reset Value
UI2C_TMCTL	UI2Cn_BA+0x8C	R/W	I^2C Timing Configure Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							**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 SDA edge in transmission mode. The delay hold time is numbers of peripheral clock = HTCTL x f_{PCLK}. **Note:** Hold time adjust function can only work in master mode, when slave mode, this field should set as 0
[15:9]	**Reserved**	Reserved.
[8:0]	**STCTL**	**Setup Time Configure Control** This field is used to generate a delay timing between SDA edge and SCL rising edge in transmission mode.. The delay setup time is numbers of peripheral clock = STCTL x f_{PCLK}.

6.21 External Bus Interface (EBI)

6.21.1 Overview

This chip is equipped with an external bus interface (EBI) for external device use. To save the connections between an external device and a chip, EBI is operating at address bus and data bus multiplex mode. The EBI supports two chip selects that can connect two external devices with different timing setting requirements.

6.21.2 Features

- Supports up to two memory banks
- Supports dedicated external chip select pin with polarity control for each bank
- Supports accessible space up to 1 Mbytes for each bank, actually external addressable space is dependent on package pin out
- Supports 8-/16-bit data width
- Supports byte write in 16-bit data width mode
- Supports Address/Data multiplexed Mode
- Supports Timing parameters individual adjustment for each memory block
- Supports LCD interface i80 mode
- Supports PDMA mode
- Supports variable external bus base clock (MCLK) which based on HCLK
- Supports configurable idle cycle for different access condition: Idle of Write command finish (W2X) and Idle of Read-to-Read (R2R)
- Supports address bus and data bus separate mode

	-	M031xB/C/D/E	M031xG/I
	M032xC/D	M032xE	M032xG/I
6.21.5.3 EBI Data Width Connection - Address Bus and Data Bus Separate Mode	-	-	●
6.21.5.4 EBI Operating Control - Continuous Data Access Mode	-	-	●

Table 6.21-1 EBI Features Comparison Table

6.21.3 Block Diagram

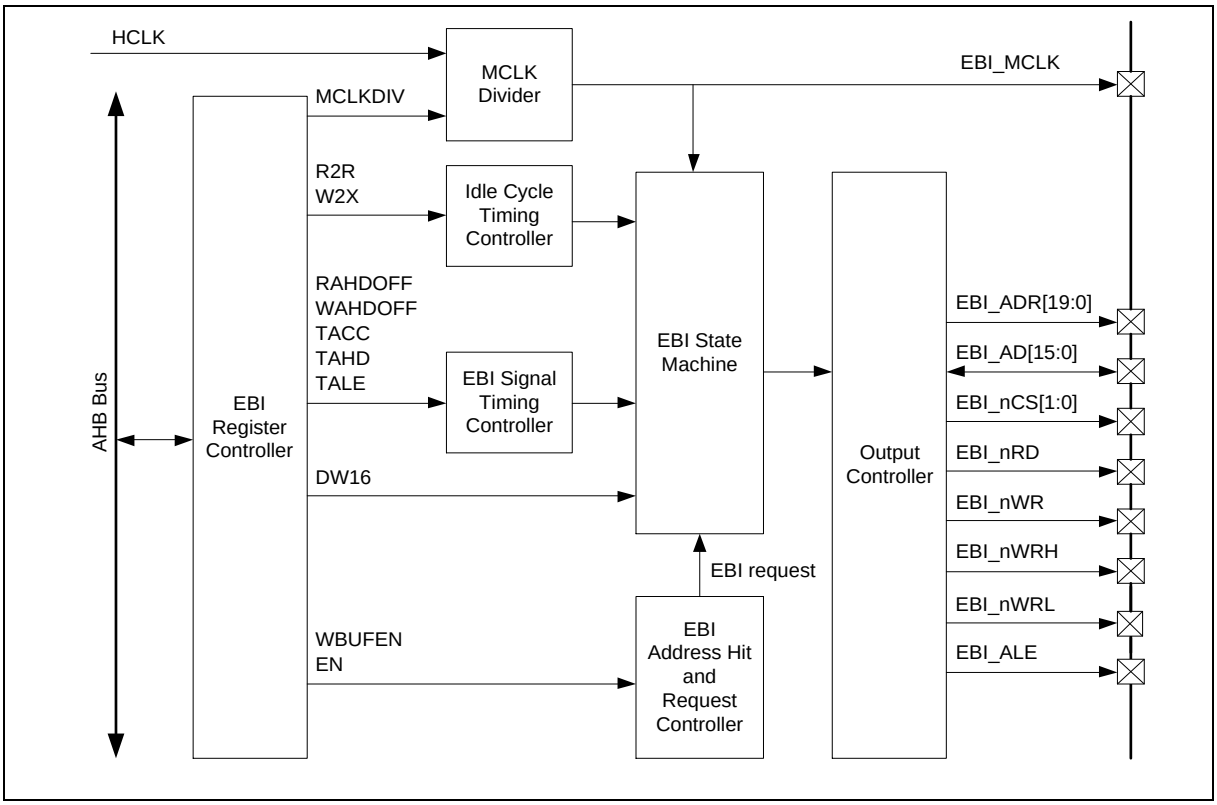


Figure 6.21-1 EBI Block Diagram

6.21.4 Basic Configuration

- Clock Source Configuration
  - Enable EBI controller clock in EBICKEN (CLK_AHBCLK[3]).
- Reset Configuration
  - Reset EBI controller in EBIRST (SYS_IPRST0[3]).

6.21.5 Functional Description

6.21.5.1 EBI Area and Address Hit

The EBI mapping address is located at 0x6000_0000 ~ 0x601F_FFFF and the total memory space is 2 Mbytes. When system request address hits EBI's memory space, the corresponding EBI chip select signal is assert and EBI state machine operates.

Chip Select	Address Mapping
EBI_nCS0	0x6000_0000 ~ 0x600F_FFFF
EBI_nCS1	0x6010_0000 ~ 0x601F_FFFF

Table 6.21-2 EBI Address Mapping

To map the whole EBI memory space, it requires 20-bit address for 8-bit data width device and 19-bit address for 16-bit data width device. For package that output less than 20-bit address, EBI will map device to mirror space. For example, the package with 18-bit EBI address, EBI will mapped external device (for Bank0/EBI_nCS0) to 0x6000_0000 ~ 0x6003_FFFF, 0x6004_0000 ~ 0x6007_FFFF,

0x6008_0000 ~ 0x600B_FFFF and 0x600C_0000 ~ 0x600F_FFFF simultaneously.

6.21.5.2 EBI Data Width Connection - Address Bus and Data Bus Multiplex Mode

The EBI supports the device whose address bus and data bus are multiplexed. For the external device with separated address and data bus, the connection to device needs additional latch device to latch the address. In this case, the pin EBI_ALE is connected to the latch device to latch the address value. Pin EBI_AD is the input of the latch device, and the output of the latch device is connected to the address of external device.

For 16-bit device, the EBI_AD [15:0] is shared by address and 16-bit data, and EBI_ADR [18:16] is dedicated for address and could be connected to 16-bit device directly. The EBI_ADR[19] will be ignored when EBI data width is set as 16-bit width. For 8-bit device, only EBI_AD [7:0] is shared by address and 8-bit data, EBI_AD[15:8] and EBI_ADR[19:16] are dedicated for address and could be connected to 8-bit device directly. Figure 6.21-2 shows the connection of 16-bit data width device and Figure 6.21-3 shows the connection of 8-bit data width device.

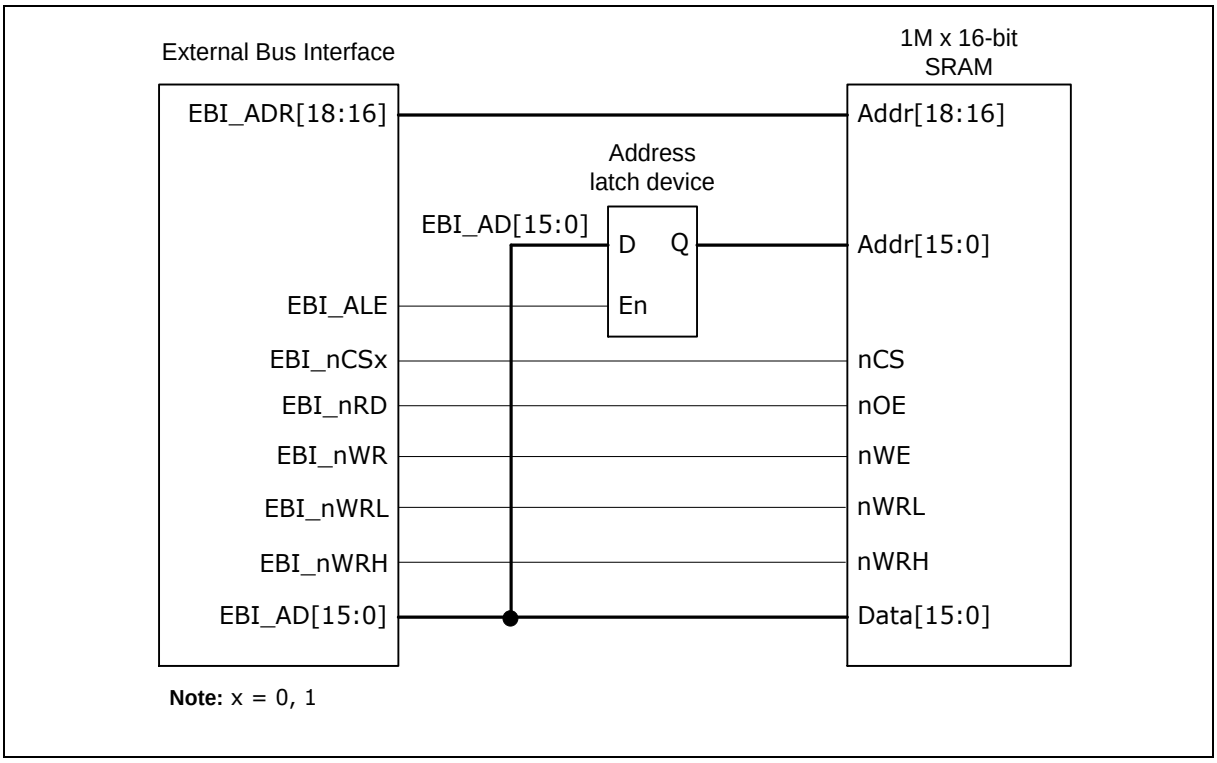


Figure 6.21-2 Connection of 16-bit EBI Data Width with 16-bit Device

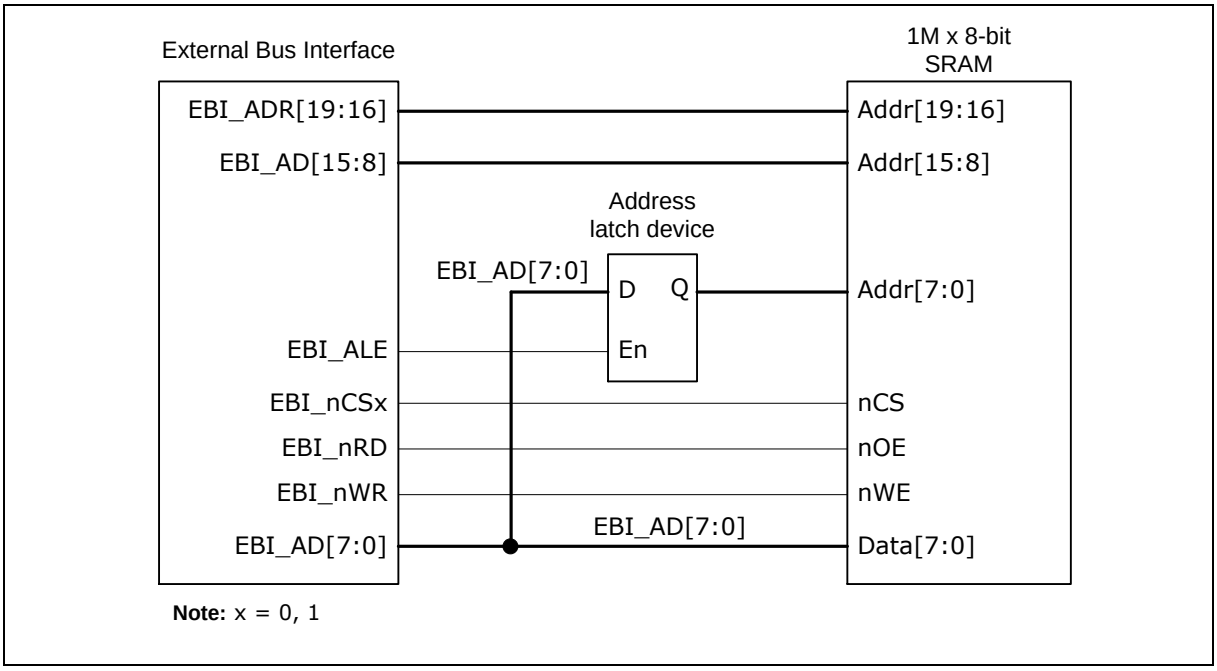


Figure 6.21-3 Connection of 8-bit EBI Data Width with 8-bit Device

When system access data width is larger than EBI data width, the EBI controller will finish a system access command by operating EBI access more than once. For example, if system requests a 32-bit data through EBI device, the EBI controller will operate accessing four times when setting EBI data width with 8-bit.

6.21.5.3 EBI Data Width Connection - Address Bus and Data Bus Separate Mode

The EBI supports address and data bus separate mode. User can enable this mode by setting ADSEPEN (EBI_CTLx[3]). When separate mode is enabled, EBI_AD is dedicated for data bus and connected directly to device data bus, EBI_ADR is dedicated for address bus.

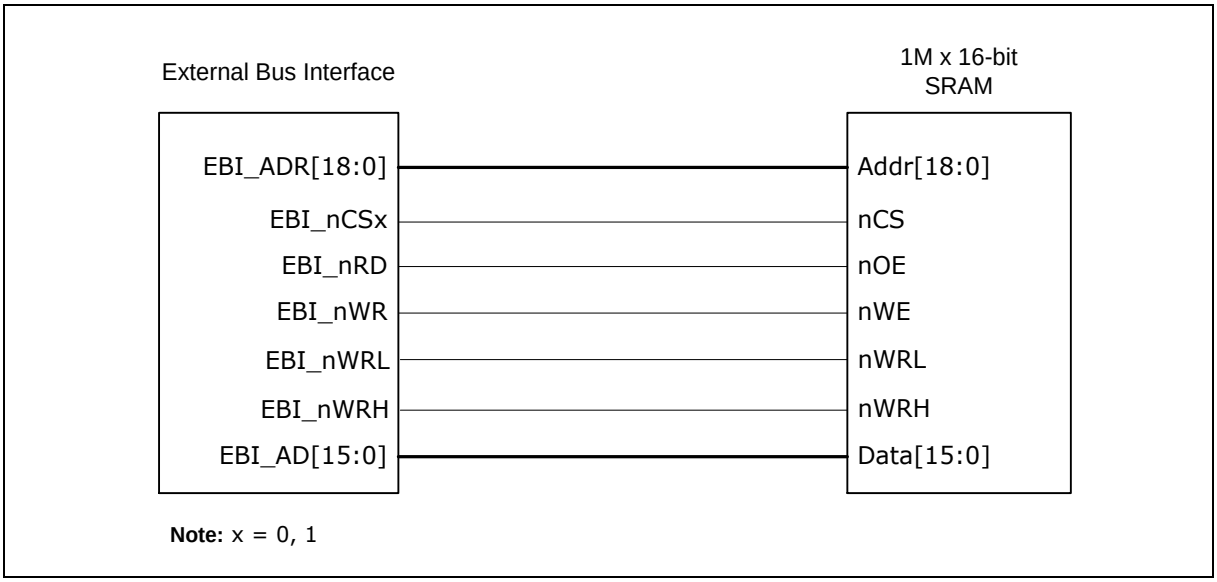


Figure 6.21-4 Connection of 16-bit EBI Data Width with 16-bit Device in Separate mode

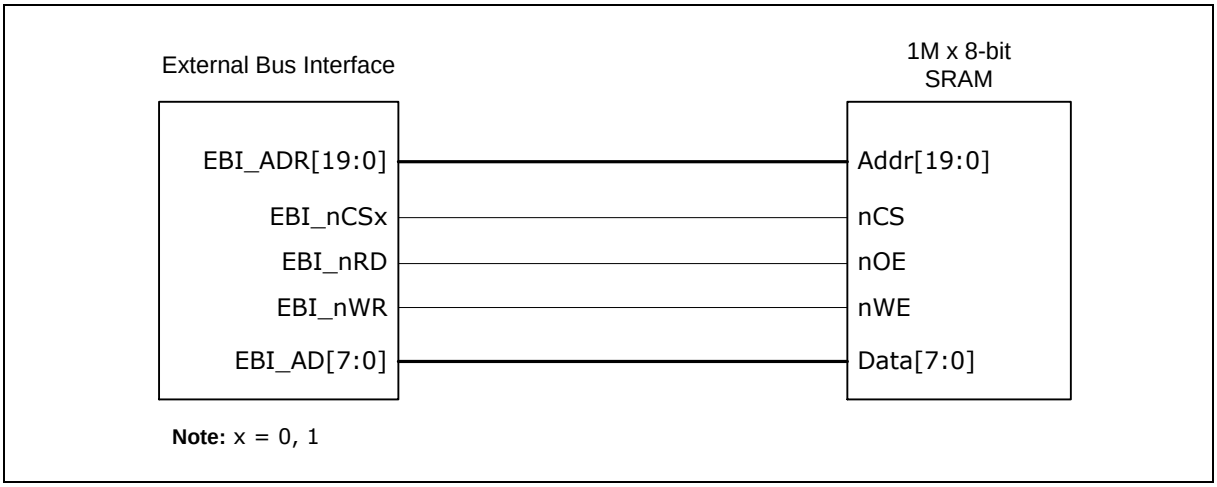


Figure 6.21-5 Connection of 8-bit EBI Data Width with 8-bit Device in Separate mode

6.21.5.4 EBI Operating Control

MCLK Control

In the chip, all EBI signals will be synchronized by EBI_MCLK when EBI is operating. When chip connects to the external device with slower operating frequency, the EBI_MCLK can divide most to HCLK/128 by setting MCLKDIV (EBI_CTLx[10:8]). Therefore, chip can be suitable for a wide frequency range of EBI device. If EBI_MCLK is set to HCLK/1, EBI signals are synchronized by positive edge of EBI_MCLK, else by negative edge of EBI_MCLK.

Operation and Access Timing Control

At the start of EBI access, chip select (EBI_nCS0, EBI_nCS1 and EBI_nCS2) asserts to low and wait one EBI_MCLK for address setup time (tASU) for address stable. Then EBI_ALE asserts to high after address is stable and keeps for a period of time (tALE) for address latch. After latch address, EBI_ALE asserts to low and wait one EBI_MCLK for latch hold time (tLHD) and another one EBI_MCLK cycle (tA2D) that is inserted behind address hold time to be the bus turn-around time for address change to data. Then EBI_nRD asserts to low when read access or EBI_nWR asserts to low when write access. Then EBI_nRD or EBI_nWR asserts to high after keeps access time (tACC) for reading output stable or writing finish. After that, EBI signals keep for data access hold time (tAHD) and chip select asserts to high, address is released by current access control.

The EBI controller provides a flexible timing control for different external device. In EBI timing control, tASU, tLHD and tA2D are fixed to 1 EBI_MCLK cycle, tAHD can modulate to 1~8 EBI_MCLK cycles by setting TAHD (EBI_TCTLx[10:8]), tACC can modulate to 1~32 EBI_MCLK cycles by setting TACC (EBI_TCTLx[7:3]), and tALE can modulate to 1~8 EBI_MCLK cycles by setting TALE (EBI_CTL0[18:16]). Some external device can support zero data access hold time accessing, the EBI controller can skipped tAHD to increase access speed by setting WAHDOFF (EBI_TCTLx[23]) and RAHDOFF (EBI_TCTLx[22]).

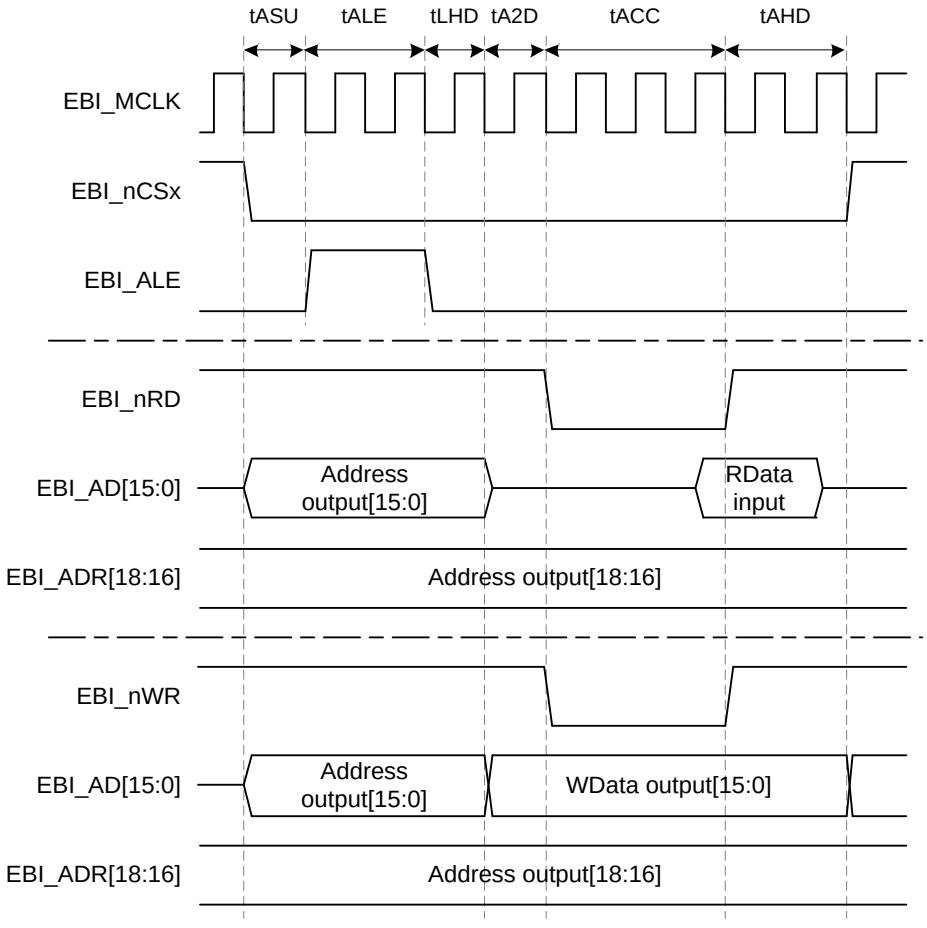
For each chip select, the EBI provides individual register with timing control except that tALE can only be controlled by EBI_CTL0.

Parameter	Value	Unit	Description
tASU	1	MCLK	Address Latch Setup Time.
tALE	1 ~ 8	MCLK	ALE High Period. Controlled by TALE (EBI_CTL0[18:16]).
tLHD	1	MCLK	Address Latch Hold Time.

tA2D	1	MCLK	Address To Data Delay (Bus Turn-Around Time).
tACC	1 ~ 32	MCLK	Data Access Time. Controlled by TACC (EBI_TCTLx[7:3]).
tAHD	1 ~ 8	MCLK	Data Access Hold Time. Controlled by TAHD (EBI_TCTLx[10:8]).
IDLE	0 ~ 15	MCLK	Idle Cycle. Controlled by R2R (EBI_TCTLx[27:24]) and W2X (EBI_TCTLx[15:12]).

Table 6.21-3 Timing Control Parameter

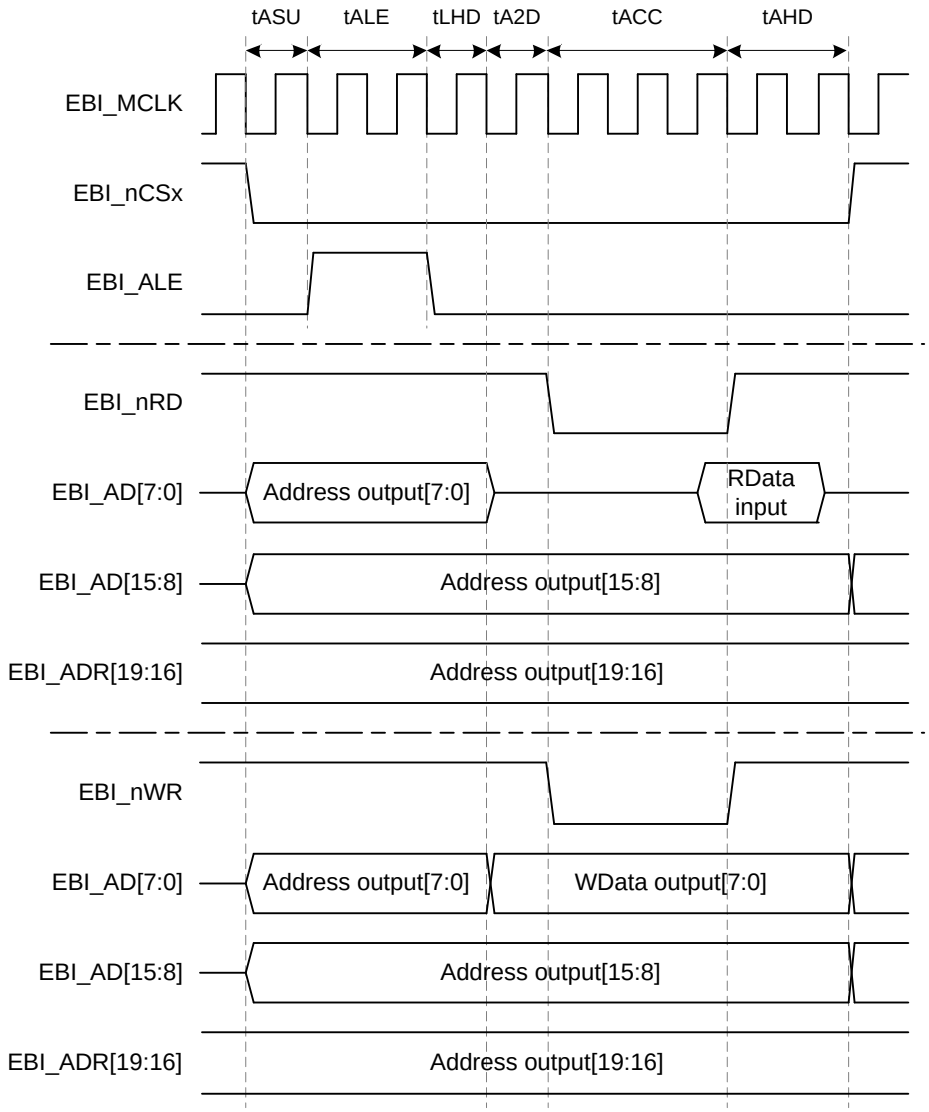
Figure 6.21-6 shows an example of setting 16-bit data width. In this example, EBI_AD bus is used for being address [15:0] and data [15:0]. When EBI_ALE assert to high, EBI_AD is address output. After address is latched, EBI_ALE asserts to low and the EBI_AD bus change to high impedance to wait device output data in read access operation, or it is used for being write data output.



Note1: The EBI_MCLK is HCLK/2, MCLKDIV (EBI_CTLx[10:8]) = 1
Note2: x = 0, 1

Figure 6.21-6 Timing Control Waveform for 16-bit Data Width

Figure 6.21-7 shows an example of setting 8-bit data width. The difference between 8-bit and 16-bit data width is EBI_AD[15:8]. In 8-bit data width setting, EBI_AD[15:8] is always Address [15:8] output so that external latch needs only 8-bit width.

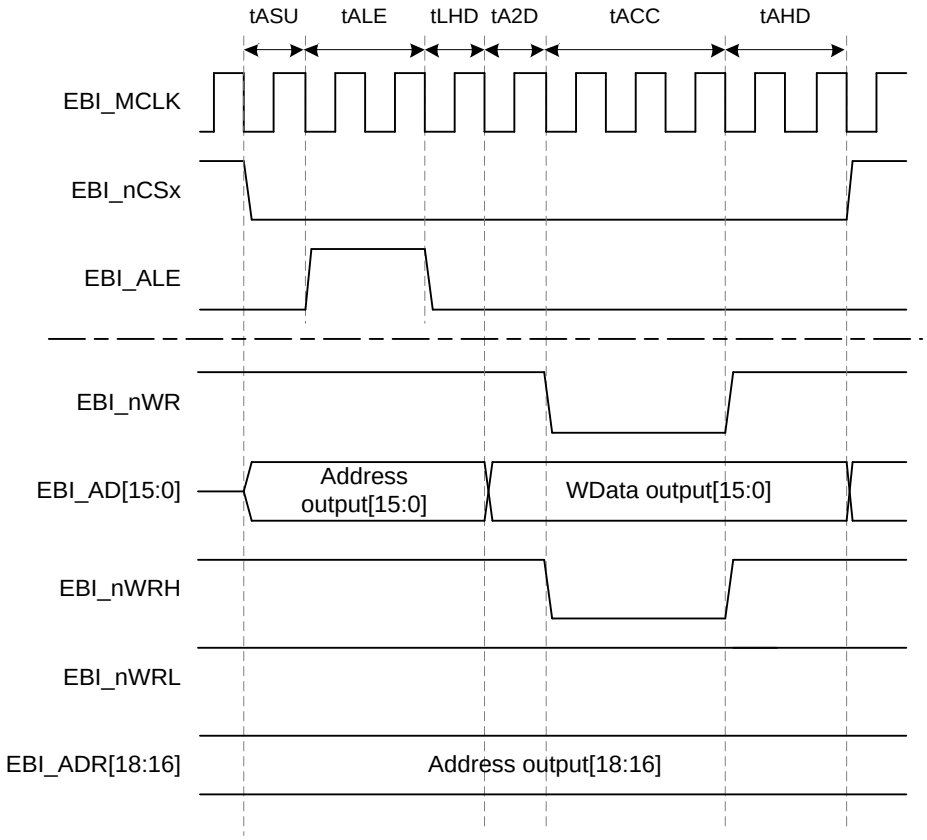


Note1: The EBI_MCLK is HCLK/2, MCLKDIV (EBI_CTLx[10:8]) = 1
Note2: x = 0, 1

Figure 6.21-7 Timing Control Waveform for 8-bit Data Width

Byte Access

The EBI supports byte access when connected to 16-bit device. The pin EBI_nWRH and EBI_nWRL assertion indicate high byte enable and low byte enable in 16-bit data bus. Figure 6.21-8 shows the write operateion of 8-bit width data in EBI_AD[15:8] with EBI_nWRH assertion.



Note1: The EBI_MCLK is HCLK/2, MCLKDIV (EBI_CTLx[10:8]) = 1
Note2: x = 0, 1

Figure 6.21-8 Timing Control Waveform for Byte Write in 16-bit Data Mode

Insert Idle Cycle

When EBI accesses continuously, there may occur bus conflict if the device access time is much slow with system operating. The EBI controller supplies additional idle cycle to solve this problem. During idle cycle, all control signals of EBI are inactive. Figure 6.21-9 shows idle cycles.

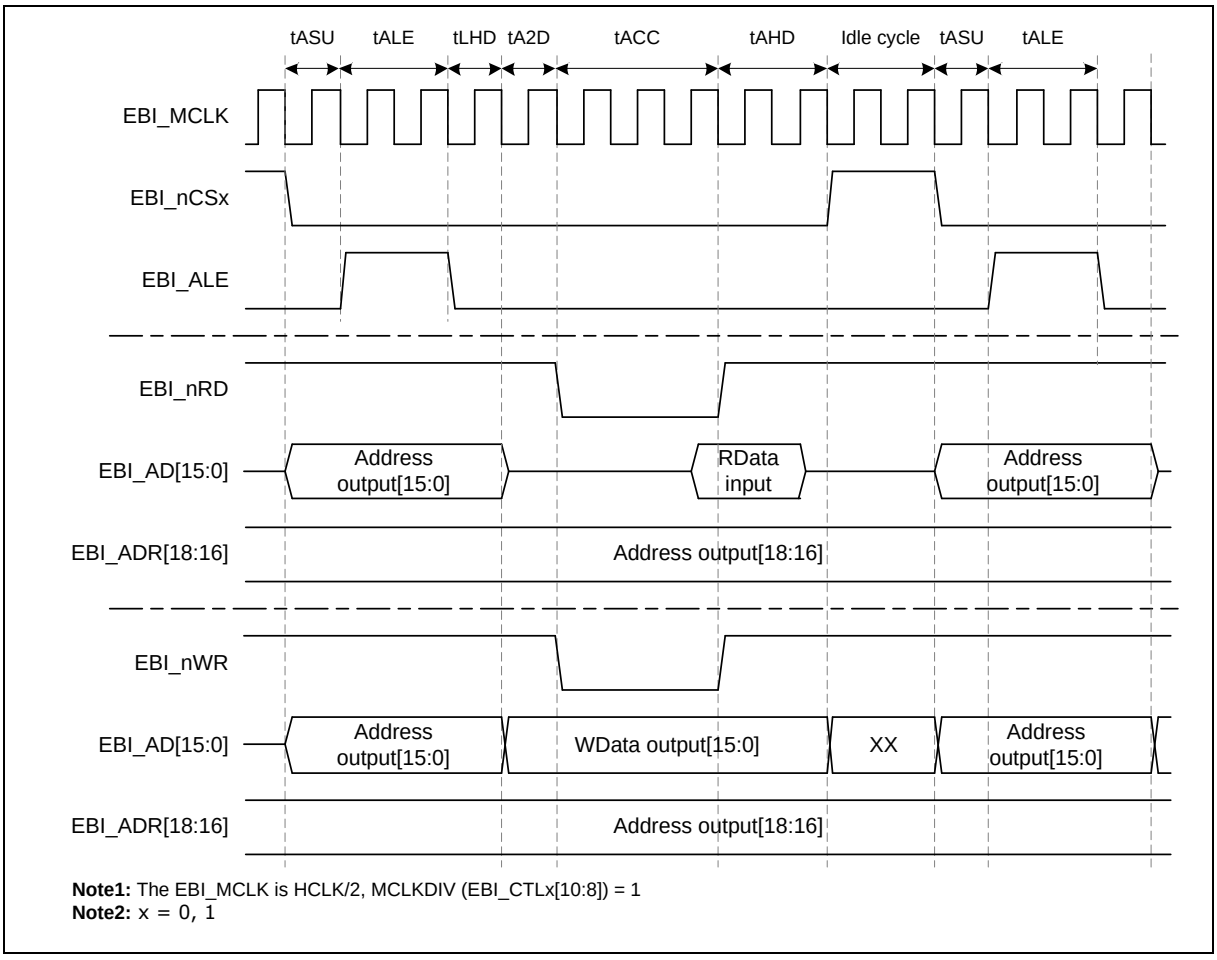


Figure 6.21-9 Timing Control Waveform for Insert Idle Cycle

There are two conditions that EBI can insert idle cycle by timing control:

1. After write access
2. After read access and before next read access (R2R idle cycle)

By setting W2X (EBI_TCTLx[15:12]), and R2R (EBI_TCTLx[27:24]), the time of idle cycle can be specified from 0~15 EBI_MCLK.

Chip Select Polarity Control

The EBI supports chip select polarity control for connecting to variable external device. When CSPOLINV (EBI_CTLx[2]) is set to 0, the chip select pins (EBI_nCSx) works as low active behavior. It means the external device can be access under EBI_nCSx at low state. When CSPOLINV (EBI_CTLx[2]) is set to 1, the chip select pin (EBI_nCS) works as high active behavior. It means the external device can be access under EBI_nCSx at high state.

Write Buffer

When user writes data to an external device through EBI bus, the EBI controller will start processing the write action immediately and the CPU is held until current EBI write action is finished. User can enable write buffer function to improve CPU and EBI access performance. When EBI write buffer function is enabled, the CPU can continuously execute other instruction during EBI controller process the write action to external device. There is one exception condition for this case. If CPU executes another data access through EBI bus when EBI process write action, the CPU will be held.

User can enable write buffer by setting WBUFEN (EBI_CTL0[24]).

Address Data Separate Mode

When EBI is set as separate mode, the tALE, tLHD, tA2D cycles are ignored. EBI_AD and EBI_ADR are dedicated for data and address bus separately.

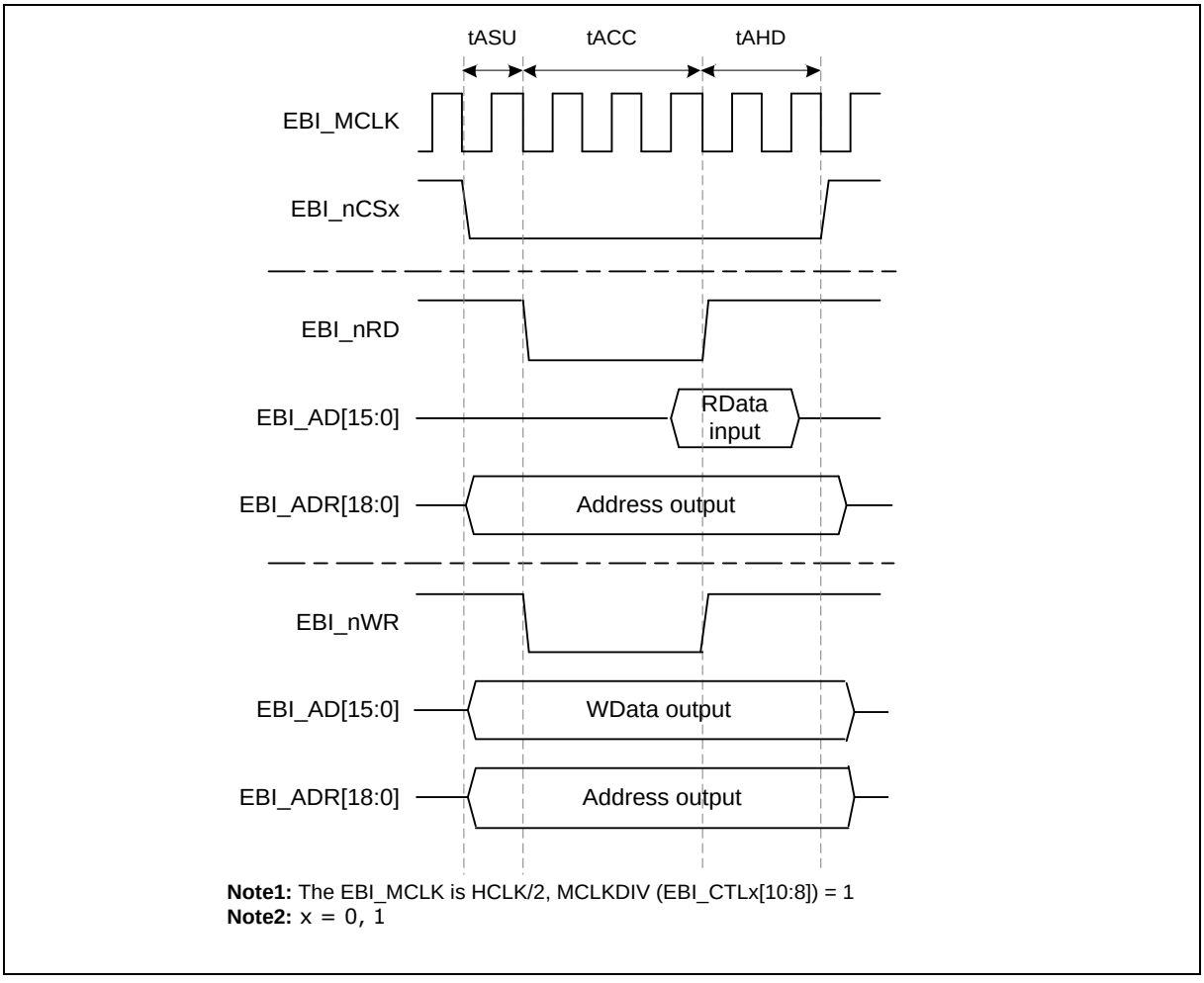


Figure 6.21-10 Timing Control Waveform for 16-bit Data Width for Separate Mode

Continuous Data Access Mode

The EBI supports continuous data access mode for the device which needs faster data access and do not need address control interface. User can enable this mode by setting CACCESS (EBI_CTLX[4]) for each bank. When EBI set as continuous data access mode, the tASU, tALE, tLHD cycles are ignored and EBI can access data continuously within one read or write command. There will be

dummy cycle between each access command. The timing waveform is shown as Figure 6.21-11.

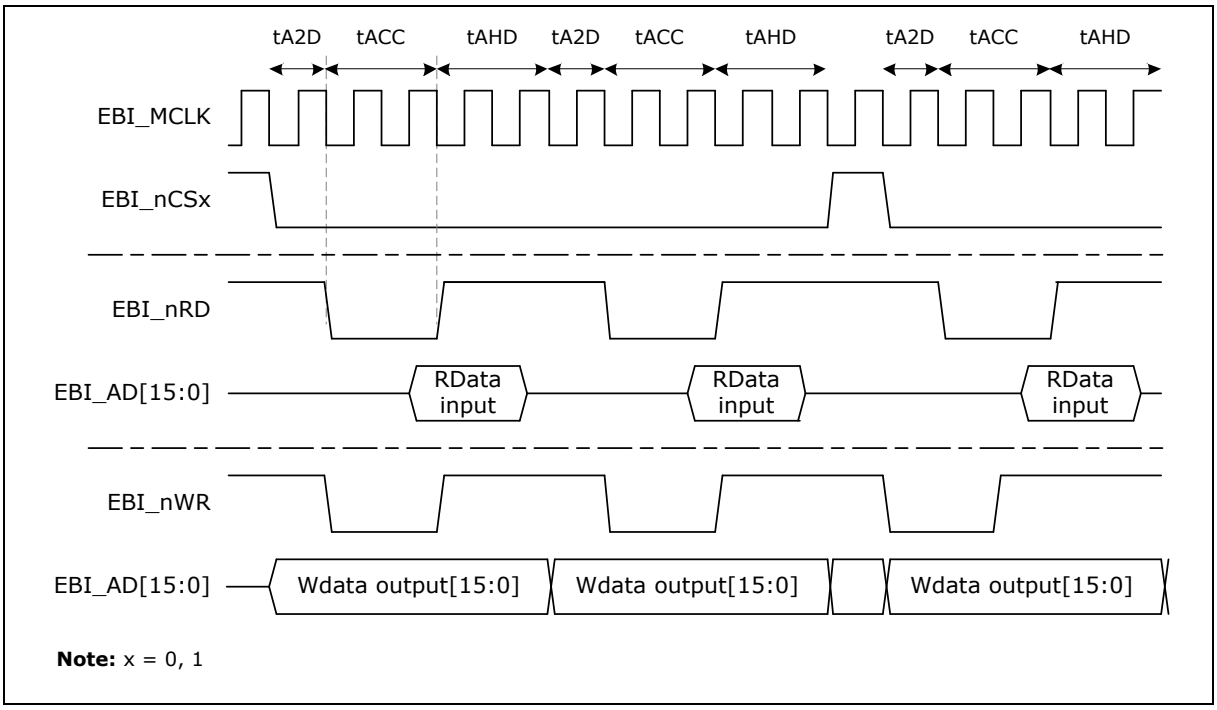


Figure 6.21-11 Timing Control Waveform for Continuous Data Access Mode

6.21.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
EBI Base Address: **EBI_BA = 0x4001_0000**				
EBI_CTL0	EBI_BA+0x00	R/W	External Bus Interface Bank0 Control Register	0x0000_0000
EBI_TCTL0	EBI_BA+0x04	R/W	External Bus Interface Bank0 Timing Control Register	0x0000_0000
EBI_CTL1	EBI_BA+0x10	R/W	External Bus Interface Bank1 Control Register	0x0000_0000
EBI_TCTL1	EBI_BA+0x14	R/W	External Bus Interface Bank1 Timing Control Register	0x0000_0000

6.21.7 Register Description

External Bus Interface Control Register (EBI_CTLx)

Register	Offset	R/W	Description	Reset Value
EBI_CTL0	EBI_BA+0x00	R/W	External Bus Interface Bank0 Control Register	0x0000_0000
EBI_CTL1	EBI_BA+0x10	R/W	External Bus Interface Bank1 Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reversed							WBUFEN
23	22	21	20	19	18	17	16
Reversed					TALE		
15	14	13	12	11	10	9	8
Reversed					MCLKDIV		
7	6	5	4	3	2	1	0
Reversed			CACCESS	ADSEPEN	CSPOLINV	DW16	EN

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**WBUFEN**	**EBI Write Buffer Enable Bit** 0 = EBI write buffer Disabled. 1 = EBI write buffer Enabled. **Note:** This bit is only available in EBI_CTL0 register.
[23:19]	**Reserved**	Reserved.
[18:16]	**TALE**	**Extend Time of ALE** The EBI_ALE high pulse period (tALE) to latch the address can be controlled by TALE. tALE = (TALE+1)*EBI_MCLK. **Note:** This field is only available in EBI_CTL0 register.
[15:11]	**Reserved**	Reserved.
[10:8]	**MCLKDIV**	**External Output Clock Divider** The frequency of EBI output clock (MCLK) is controlled by MCLKDIV as follow: 000 = HCLK/1. 001 = HCLK/2. 010 = HCLK/4. 011 = HCLK/8. 100 = HCLK/16. 101 = HCLK/32. 110 = HCLK/64. 111 = HCLK/128.
[7:5]	**Reserved**	Reserved.
[4]	**CACCESS**	**Continuous Data Access Mode**

		When Continuous access mode is enabled, the tASU, tALE and tLHD cycles are bypass for continuous data transfer request. 0 = Continuous data access mode Disabled. 1 = Continuous data access mode Enabled.
[3]	**ADSEPEN**	**EBI Address/Data Bus Separating Mode Enable Bit** 0 = Address/Data Bus Separating Mode Disabled. 1 = Address/Data Bus Separating Mode Enabled.
[2]	**CSPOLINV**	**Chip Select Pin Polar Inverse** This bit defines the active level of EBI chip select pin (EBI_nCS). 0 = Chip select pin (EBI_nCS) is active low. 1 = Chip select pin (EBI_nCS) is active high.
[1]	**DW16**	**EBI Data Width 16-bit Select** This bit defines if the EBI data width is 8-bit or 16-bit. 0 = EBI data width is 8-bit. 1 = EBI data width is 16-bit.
[0]	**EN**	**EBI Enable Bit** This bit is the functional enable bit for EBI. 0 = EBI function Disabled. 1 = EBI function Enabled.

External Bus Interface Timing Control Register (EBI_TCTLx)

Register	Offset	R/W	Description	Reset Value
EBI_TCTL0	EBI_BA+0x04	R/W	External Bus Interface Bank0 Timing Control Register	0x0000_0000
EBI_TCTL1	EBI_BA+0x14	R/W	External Bus Interface Bank1 Timing Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				R2R			
23	22	21	20	19	18	17	16
WAHDOFF	RAHDOFF	Reserved					
15	14	13	12	11	10	9	8
W2X				Reversed	TAHD		
7	6	5	4	3	2	1	0
TACC					Reserved		

Bits	Description	
[31:30]	**Reserved**	Reserved.
[27:24]	**R2R**	**Idle Cycle Between Read-to-read** This field defines the number of R2R idle cycle. R2R idle cycle = (R2R * EBI_MCLK). **Note:** When read action is finished and the next action is going to read, R2R idle cycle is inserted and EBI_nCS return to idle state.
[23]	**WAHDOFF**	**Access Hold Time Disable Control When Write** 0 = Data Access Hold Time (tAHD) during EBI writing Enabled. 1 = Data Access Hold Time (tAHD) during EBI writing Disabled.
[22]	**RAHDOFF**	**Access Hold Time Disable Control When Read** 0 = Data Access Hold Time (tAHD) during EBI reading Enabled. 1 = Data Access Hold Time (tAHD) during EBI reading Disabled.
[21:16]	**Reserved**	Reserved.
[15:12]	**W2X**	**Idle Cycle After Write** This field defines the number of W2X idle cycle. W2X idle cycle = (W2X * EBI_MCLK). **Note:** When write action is finished, W2X idle cycle is inserted and EBI_nCS return to idle state.
[11]	**Reserved**	Reserved.
[10:8]	**TAHD**	**EBI Data Access Hold Time** TAHD defines data access hold time (tAHD). tAHD = (TAHD +1) * EBI_MCLK.
[7:3]	**TACC**	**EBI Data Access Time** TACC defines data access time (tACC).

		tACC = (TACC +1) * EBI_MCLK.
[2:0]	**Reserved**	Reserved.

6.22 USB 2.0 Full-Speed Device Controller (USBD)

6.22.1 Overview

There is one set of USB 2.0 full-speed device controller and transceiver in this device. It is compliant with USB 2.0 full-speed device specification and supports control/bulk/interrupt/ isochronous transfer types.

In this device controller, there are two main interfaces: the APB bus and USB bus which comes from the USB PHY transceiver. For the APB bus, the CPU can program control registers through it. There are 512 Bytes internal SRAM as data buffer in this controller. For IN or OUT transfer, it is necessary to write data to SRAM or read data from SRAM through the APB interface or SIE. User needs to set the effective starting address of SRAM for each endpoint buffer through buffer segmentation register (USBD_BUFSEGx).

There are 8 endpoints in this controller. Each of the endpoint can be configured as IN or OUT endpoint. All the operations including Control, Bulk, Interrupt and Isochronous transfer are implemented in this block. The block of “Endpoint Control” is also used to manage the data sequential synchronization, endpoint states, current start address, transaction status, and data buffer status for each endpoint.

There are four different interrupt events in this controller. They are the no-event-wake-up, device plug-in or plug-out event, USB events, like IN ACK, OUT ACK etc, and BUS events, like suspend and resume, etc. Any event will cause an interrupt, and users just need to check the related event flags in interrupt event status register (USBD_INTSTS) to acknowledge what kind of interrupt occurring, and then check the related USB Endpoint Status Register USBD_EPSTS0 to acknowledge what kind of event occurring in this endpoint.

A software-disconnect function is also supported for this USB controller. It is used to simulate the disconnection of this device from the host. If user enables SE0 bit (USBD_SE0), the USB controller will force the output of USB_D+ and USB_D- to level low and its function is disabled. After disable the SE0 bit, host will enumerate the USB device again.

For more information on the Universal Serial Bus, please refer to Universal Serial Bus Specification Revision *2.0.*

6.22.2 Features

- Compliant with USB 2.0 Full-Speed specification
- Provides 1 interrupt vector with 5 different interrupt events (SOF, NEVWK, VBUSDET, USB and BUS)
- Supports Control/Bulk/Interrupt/Isochronous transfer type
- Supports suspend function when no bus activity existing for 3 ms
- Supports 8 endpoints for configurable Control/Bulk/Interrupt/Isochronous transfer types and maximum 512 byte buffer size
- Provides remote wake-up capability

Section	-	M031xB/C/D/E	M031xG/I
	M032xC/D	M032xE	M032xG/I
6.22.7 Register Description USB Configuration Register (USB_CFGx) DSQSYNC OUT Token Transaction	•	-	-

6.22.3 Block Diagram

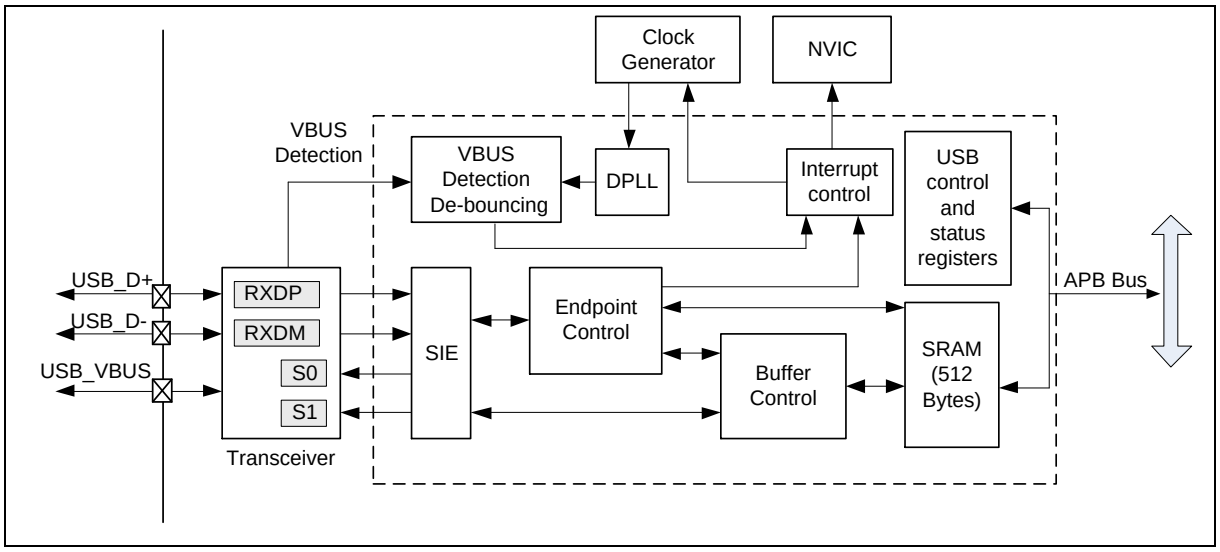


Figure 6.22-1 USB Block Diagram

6.22.4 Basic Configuration

The USBD clock source is derived from PLL. User has to set the PLL related configurations before USB device controller is enabled. Set the USBDCKEN (CLK_APBCLK0[27]) bit to enable USBD clock and 4-bit pre-scaler USBDIV (CLK_CLKDIV0[7:4]) to generate the proper USBD clock rate.

6.22.4.1 USB 2.0 Full-Speed Device Basic Configuration

- Clock source Configuration
  - Setting PLL controller (CLK_PLLCTL) or enable HIRC48M.
  - Select the clock divider number of USBD peripheral clock on USBDIV(CLK_CLKDIV0[7:4])
  - Enable USBD peripheral clock in USBDCKEN (CLK_APBCLK0[27]).
- Reset Configuration
  - Reset USBD controller in USBDRST (SYS_IPRST1[27]).

6.22.5 Functional Description

6.22.5.1 Serial Interface Engine (SIE)

The SIE is the front-end of the device controller and handles most of the USB packet protocol. The SIE typically comprehends signaling up to the transaction level. The functions that it handles could include:

- Packet recognition and transaction sequencing
- SOP, EOP, RESET, RESUME signal detection/generation
- Clock/Data separation
- NRZI Data encoding/decoding and bit-stuffing
- CRC generation and checking (for Token and Data)
- Packet ID (PID) generation and checking/decoding
- Serial-Parallel/Parallel-Serial conversion

6.22.5.2 Endpoint Control

This controller supports 8 endpoints. Each of the endpoint can be configured as Control, Bulk, Interrupt, or Isochronous transfer type. All the operations including Control, Bulk, Interrupt and Isochronous transfer are implemented in this block. It is also used to manage the data sequential synchronization, endpoint state control, current endpoint start address, current transaction status, and data buffer status in each endpoint.

6.22.5.3 Digital Phase Lock Loop (DPLL)

The bit rate of USB data is 12 MHz. The DPLL uses the 48 MHz which comes from the clock controller to lock the input data RXDP and RXDM. The 12 MHz bit rate clock is also converted from DPLL.

6.22.5.4 VBUS Detection De-bouncing

A USB device may be plugged-in or plugged-out from the USB host. To monitor the state of a USB device when it is detached from the USB host, the device controller provides hardware de-bouncing for USB VBUS detection interrupt to avoid bounce problems on USB plug-in or unplug. VBUS detection interrupt appears about 10 ms later than USB plug-in or plug-out. User can acknowledge USB plug-in/plug-out by reading USBD_VBUSDET register. The VBUSDET flag represents the current state on the bus without de-bouncing. If VBUSDET is 1, it means the USB cable is plugged-in. If user polls the flag to check USB state, software de-bouncing must be added if needed.

6.22.5.5 Interrupt control

This USB provides 1 interrupt vector with 4 interrupt events (NEVWK, VBUSDET, USB and BUS). The NEVWK event occurs after waking up the system from Power-down mode (The power mode function is defined in system power-down control register, CLK_PWRCTL). The VBUSDET event is used for USB plug-in or unplug. The USB event notifies users of some USB requests, such as IN ACK, OUT ACK., and the BUS event notifies users of some bus events, such as suspend and, resume. The related bits must be set in the interrupt enable register (USBD_INTEN) of USB Device Controller to enable USB interrupts.

NEVWK interrupt is only presented when no the other USB interrupt events happened more than 20ms after the chip is waked up from Power-down mode. After the chip enters Power-down mode, any change on USB_VBUS, USB_D+ and USB_D- can wake up this chip if USB wake-up function is enabled. If this change is not intentionally, no interrupt but NEVWK interrupt will occur. After waking up by USB, this interrupt will occur when no the other USB interrupt events are presented for more than 20ms. Figure 6.22-2 shows the control flow of wake-up interrupt.

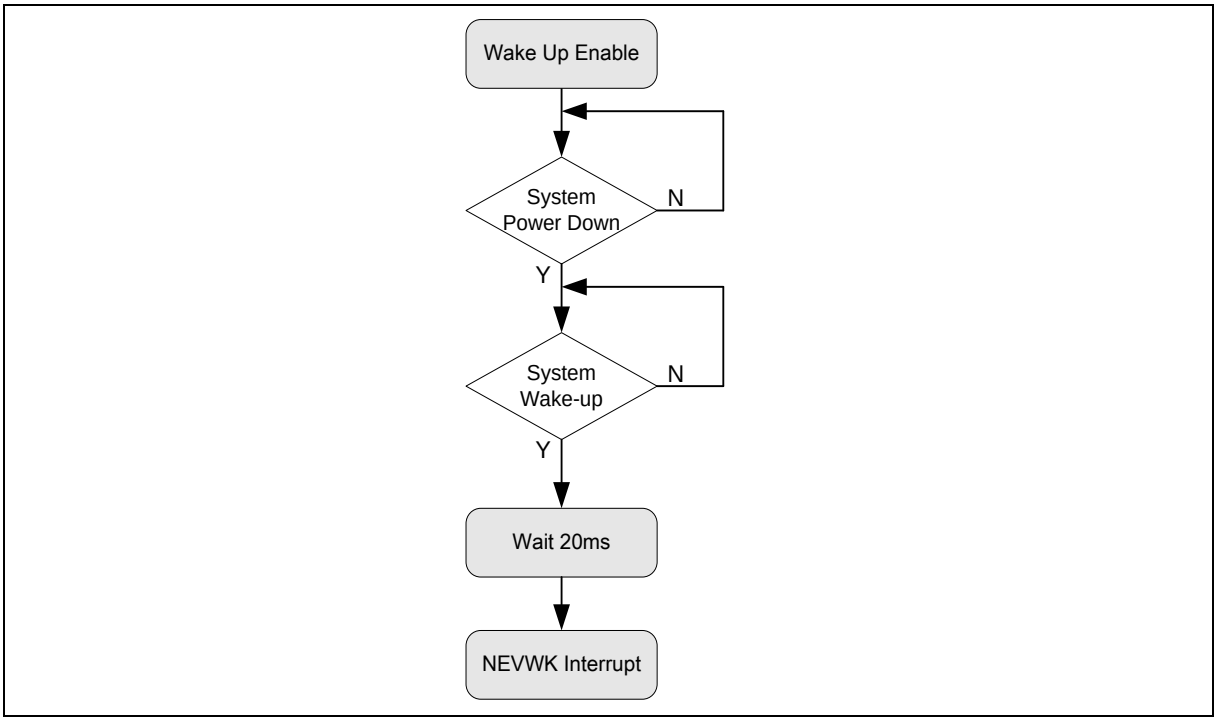


Figure 6.22-2 NEVWK Interrupt Operation Flow

The USB interrupt is used to notify users of any USB event on the bus, and user can read EPSTS (USBD_EPSTS0) and EPEVT7~0 (USBD_INTSTS[31:0]) to take necessary responses.

Same as USB interrupt, BUS interrupt notifies users of some bus events, like USB reset, suspend, time-out, and resume. A user can read USBD_ATTR to acknowledge bus events.

6.22.5.6 Power Saving

User can write 0 to USBD_ATTR[4] to disable PHY under special circumstances, like suspend, to conserve power.

6.22.5.7 Buffer Control

There is 512 byte of SRAM in the controller and the 8 endpoints share this buffer. User shall configure each endpoint's effective starting address in the buffer segmentation register before the USB function active. The "Buffer Control" block is used to control each endpoint's effective starting address and its SRAM size is defined in the USBD_MXPLDx register.

Figure 6.22-3 depicts the starting address for each endpoint according the content of USBD_BUFSEGx and USBD_MXPLDx registers. If the USBD_BUFSEG0 is programmed as 0x08h and USBD_MXPLD0 is set as 0x40h, the SRAM size of endpoint 0 is start from USBD_BA+0x108h and end in USBD_BA+0x148h.

Note: The USBD SRAM base is USBD_BA+0x100h.

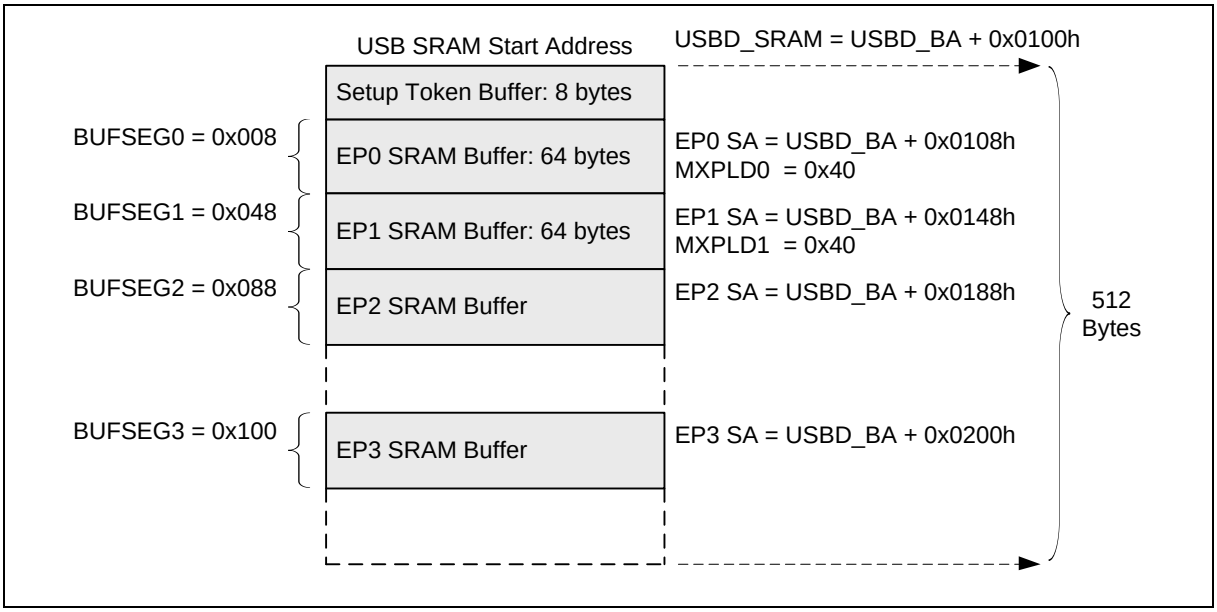


Figure 6.22-3 Endpoint SRAM Structure

6.22.5.8 Handling Transactions with USB Device Peripheral

User can use interrupt or polling USBD_INTSTS to monitor the USB transactions. When transactions occur, USBD_INTSTS will be set by hardware and send an interrupt request to CPU (if related interrupt enabled), or user can polling USBD_INTSTS to get these events without interrupt. The following is the control flow with interrupt enabled.

When USB host has requested data from a device controller, user needs to prepare related data in the specified endpoint buffer in advance. After buffering the required data, user needs to write the actual data length in the specified USBD_MXPLDx register. Once this register is written, the internal signal “In_Rdy” will be asserted and the buffering data will be transmitted immediately after receiving associated IN token from Host. Note that after transferring the specified data, the signal “In_Rdy” will de-assert automatically by hardware.

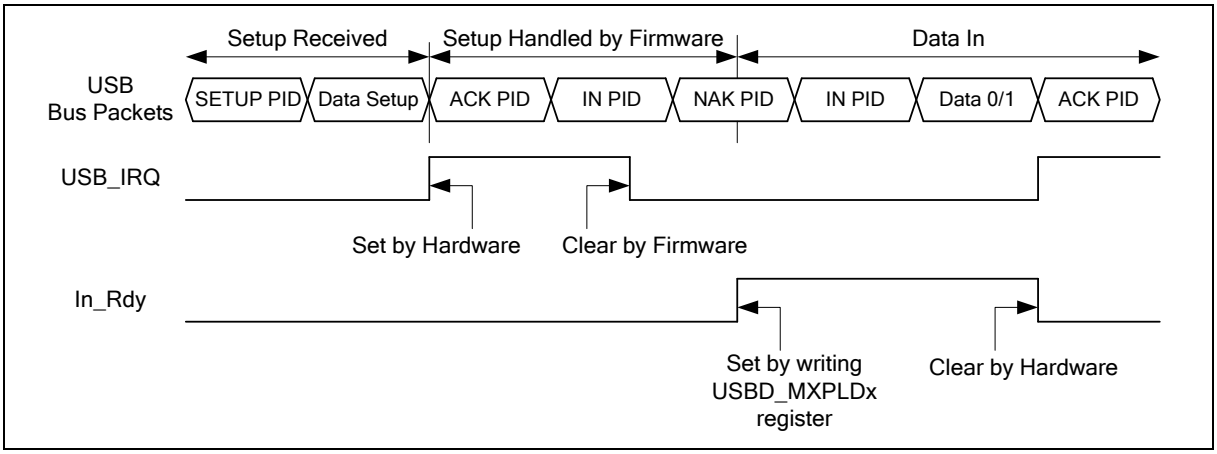


Figure 6.22-4 Setup Transaction Followed by Data IN Transaction

Alternatively, when USB host wants to transmit data to the OUT endpoint in the device controller, hardware will buffer these data to the specified endpoint buffer. After this transaction is completed, hardware will record the data length in specified USBD_MXPLDx register and de-assert the internal

signal “Out_Rdy”. This will avoid hardware accepting next transaction until user moves out the current data in the related endpoint buffer. Once users have processed this transaction, the specified USBD_MXPLDx register needs to be written by firmware to assert the signal “Out_Rdy” again to accept the next transaction.

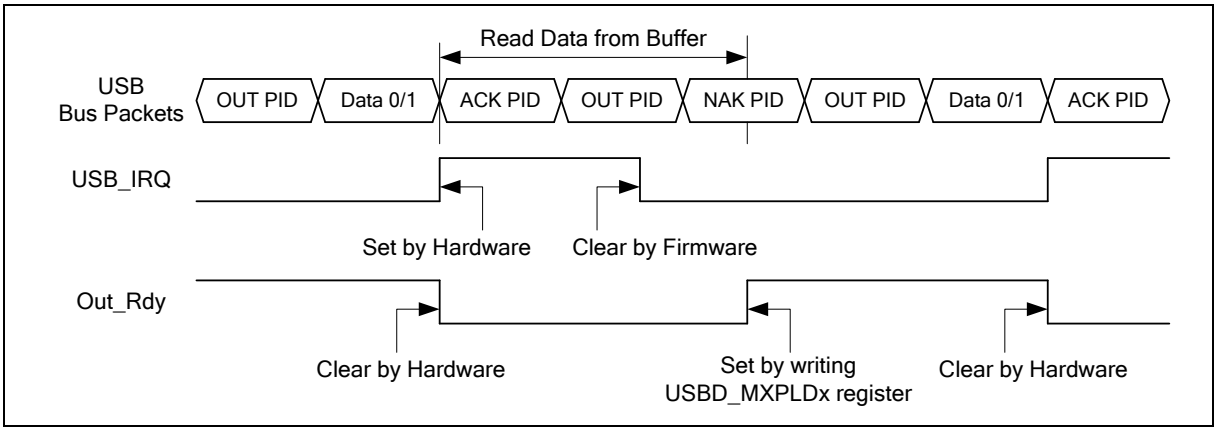


Figure 6.22-5 Data Out Transfer

6.22.5.9 Link Power Management (LPM)

Power Management(LPM) which is similar to the suspend/resume function, but has transitional latencies of tens of microseconds between power states (instead of three to greater than 20 millisecond latencies of the USB2.0 suspend/resume)

A new, fast mechanism can transition the bus on a root port from an enable state (called L0) to a new sleep state (called L1). See Table 6.22-1 for detailed definition of L0 and L1 state. The register USBD_ATTR & USBD_LPMATTR can let user know current power state for LPM mechanism.

LPM State	Description
L0(On)	In this state, the port is enabled for propagation of transaction signaling traffic. A port in L0 is either actively transmitting or receiving data (L0-Active) or able to do so but not currently transmitting or receiving information (L0-Idle). While in this state Start-of-Frame (SOF) packets are issued by the host at a rate corresponding to the speed of the client device
L1(Sleep)	L1 is similar to L2 (below) but supports finer granularity in use. When in L1, the line state is identical to L2. Entry to L1 is started by a request to a hub or host port to transition to L1. A LPM transaction is sent to the downstream device. The requested transition can only occur if the device response with an ACK handshake. Exit from L1 is via remote wake, resume signaling, reset signaling or disconnect. L1 does not impose any specific power draw requirements (from VBUS) on the attached device as L2 does. Either the host or device can initiate resume signaling when in L1. Although the signaling levels of resume are the same as L2, the duration of the signaling and transitional latencies associated with the L1 to L0 transition are much shorter
L2(Suspend)	This is the formalized name for USB 2.0 Suspend, Entry to L2 is nominally triggered by a command to a hub or host port to transition to suspend. The device discovers the suspend condition via observing 3ms of inactivity. The resultant line state is either Low or Full-speed idle. L2 also imposes power draw requirements (from VBUS) on the attached device. Exit from this state is via remote wake, resume signaling, reset signaling or disconnect.
L3(Off)	In this state, the port is not capable of performing any data signaling. It corresponds to the powered-off, disconnected, and disabled states

Table 6.22-1 USB Link Power Manager (Lx) States

The state transaction process please refer to Figure 6.22-6, and for more information on the USB Link Power Manager(LPM), please refer to USB2.0 Link Power Mangement ECN.

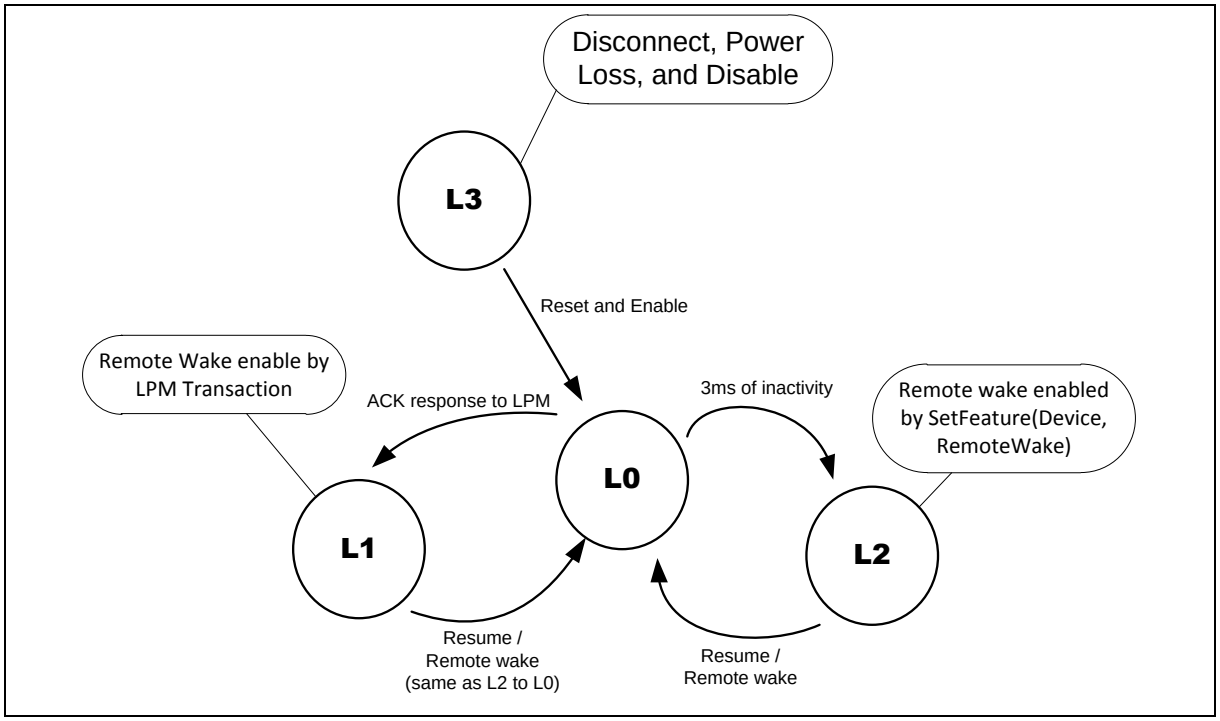


Figure 6.22-6 LPM State Transition Diagram

6.22.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
USBD Base Address: **USBD_BA = 0x400C_0000**				
USBD_INTEN	USBD_BA+0x000	R/W	USB Device Interrupt Enable Register	0x0000_0000
USBD_INTSTS	USBD_BA+0x004	R/W	USB Device Interrupt Event Status Register	0x0000_0000
USBD_FADDR	USBD_BA+0x008	R/W	USB Device Function Address Register	0x0000_0000
USBD_EPSTS	USBD_BA+0x00C	R	USB Device Endpoint Status Register	0x0000_0000
USBD_ATTR	USBD_BA+0x010	R/W	USB Device Bus Status and Attribution Register	0x0000_0040
USBD_VBUSDET	USBD_BA+0x014	R	USB Device VBUS Detection Register	0x0000_0000
USBD_STBUFSEG	USBD_BA+0x018	R/W	SETUP Token Buffer Segmentation Register	0x0000_0000
USBD_EPSTS0	USBD_BA+0x020	R	USB Device Endpoint Status Register 0	0x0000_0000
USBD_LPMATTR	USBD_BA+0x088	R	USB LPM Attribution Register	0x0000_0000
USBD_FN	USBD_BA+0x08C	R	USB Frame Number Register	0x0000_0XXX
USBD_SE0	USBD_BA+0x090	R/W	USB Device Drive SE0 Control Register	0x0000_0001
USBD_BUFSEG0	USBD_BA+0x500	R/W	Endpoint 0 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD0	USBD_BA+0x504	R/W	Endpoint 0 Maximal Payload Register	0x0000_0000
USBD_CFG0	USBD_BA+0x508	R/W	Endpoint 0 Configuration Register	0x0000_0000
USBD_CFGP0	USBD_BA+0x50C	R/W	Endpoint 0 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG1	USBD_BA+0x510	R/W	Endpoint 1 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD1	USBD_BA+0x514	R/W	Endpoint 1 Maximal Payload Register	0x0000_0000
USBD_CFG1	USBD_BA+0x518	R/W	Endpoint 1 Configuration Register	0x0000_0000
USBD_CFGP1	USBD_BA+0x51C	R/W	Endpoint 1 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG2	USBD_BA+0x520	R/W	Endpoint 2 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD2	USBD_BA+0x524	R/W	Endpoint 2 Maximal Payload Register	0x0000_0000
USBD_CFG2	USBD_BA+0x528	R/W	Endpoint 2 Configuration Register	0x0000_0000
USBD_CFGP2	USBD_BA+0x52C	R/W	Endpoint 2 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG3	USBD_BA+0x530	R/W	Endpoint 3 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD3	USBD_BA+0x534	R/W	Endpoint 3 Maximal Payload Register	0x0000_0000
USBD_CFG3	USBD_BA+0x538	R/W	Endpoint 3 Configuration Register	0x0000_0000
USBD_CFGP3	USBD_BA+0x53C	R/W	Endpoint 3 Set Stall and Clear In/Out Ready Control Register	0x0000_0000

USBD_BUFSEG4	USBD_BA+0x540	R/W	Endpoint 4 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD4	USBD_BA+0x544	R/W	Endpoint 4 Maximal Payload Register	0x0000_0000
USBD_CFG4	USBD_BA+0x548	R/W	Endpoint 4 Configuration Register	0x0000_0000
USBD_CFGP4	USBD_BA+0x54C	R/W	Endpoint 4 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG5	USBD_BA+0x550	R/W	Endpoint 5 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD5	USBD_BA+0x554	R/W	Endpoint 5 Maximal Payload Register	0x0000_0000
USBD_CFG5	USBD_BA+0x558	R/W	Endpoint 5 Configuration Register	0x0000_0000
USBD_CFGP5	USBD_BA+0x55C	R/W	Endpoint 5 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG6	USBD_BA+0x560	R/W	Endpoint 6 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD6	USBD_BA+0x564	R/W	Endpoint 6 Maximal Payload Register	0x0000_0000
USBD_CFG6	USBD_BA+0x568	R/W	Endpoint 6 Configuration Register	0x0000_0000
USBD_CFGP6	USBD_BA+0x56C	R/W	Endpoint 6 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_BUFSEG7	USBD_BA+0x570	R/W	Endpoint 7 Buffer Segmentation Register	0x0000_0000
USBD_MXPLD7	USBD_BA+0x574	R/W	Endpoint 7 Maximal Payload Register	0x0000_0000
USBD_CFG7	USBD_BA+0x578	R/W	Endpoint 7 Configuration Register	0x0000_0000
USBD_CFGP7	USBD_BA+0x57C	R/W	Endpoint 7 Set Stall and Clear In/Out Ready Control Register	0x0000_0000

Memory Type	**Address**	**Size**	**Description**
USBD_BA = 0x400C_0000			
USBD_SRAM	USBD_BA+0x100 ~ USBD_BA+0x2FF	512 Bytes	The SRAM is used for the entire endpoints buffer. Refer to section 6.22.5.7 for the endpoint SRAM structure and its description.

6.22.7 Register Description

USB Interrupt Enable Register (USBD_INTEN)

Register	Offset	R/W	Description	Reset Value
USBD_INTEN	USBD_BA+0x000	R/W	USB Device Interrupt Enable Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**INNAKEN**	**Active NAK Function and Its Status in IN Token** 0 = When device responds NAK after receiving IN token, IN NAK status will not be updated to USBD_EPSTS0 register, so that the USB interrupt event will not be asserted. 1 = IN NAK status will be updated to USBD_EPSTS0 register and the USB interrupt event will be asserted, when the device responds NAK after receiving IN token.
[14:9]	**Reserved**	Reserved.
[8]	**WKEN**	**Wake-up Function Enable Bit** 0 = USB wake-up function Disabled. 1 = USB wake-up function Enabled.
[7:5]	**Reserved**	Reserved.
[4]	**SOFIEN**	**Start of Frame Interrupt Enable Bit** 0 = SOF Interrupt Disabled. 1 = SOF Interrupt Enabled.
[3]	**NEVWKIEN**	**USB No-event-wake-up Interrupt Enable Bit** 0 = No-event-wake-up Interrupt Disabled. 1 = No-event-wake-up Interrupt Enabled.
[2]	**VBDETIEN**	**VBUS Detection Interrupt Enable Bit** 0 = VBUS detection Interrupt Disabled. 1 = VBUS detection Interrupt Enabled.
[1]	**USBIEN**	**USB Event Interrupt Enable Bit** 0 = USB event interrupt Disabled. 1 = USB event interrupt Enabled.
[0]	**BUSIEN**	**Bus Event Interrupt Enable Bit**

		0 = BUS event interrupt Disabled. 1 = BUS event interrupt Enabled.

USB Interrupt Event Status Register (USBD_INTSTS)

Register	Offset	R/W	Description	Reset Value
USBD_INTSTS	USBD_BA+0x004	R/W	USB Device Interrupt Event Status Register	0x0000_0000

31	30	29	28	27	26	25	24
SETUP	Reserved						
23	22	21	20	19	18	17	16
EPEVT7	EPEVT6	EPEVT5	EPEVT4	EPEVT3	EPEVT2	EPEVT1	EPEVT0
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved			SOFIF	NEVWKIF	VBDETIF	USBIF	BUSIF

Bits	Description	
[31]	**SETUP**	**Setup Event Status** 0 = No Setup event. 1 = Setup event occurred, cleared by write 1 to USBD_INTSTS[31].
[30:24]	**Reserved**	Reserved.
[23]	**EPEVT7**	**Endpoint 7's USB Event Status** 0 = No event occurred in endpoint 7. 1 = USB event occurred on Endpoint 7, check USBD_EPSTS0[31:28] to know which kind of USB event was occurred, cleared by write 1 to USBD_INTSTS[23] or USBD_INTSTS[1].
[22]	**EPEVT6**	**Endpoint 6's USB Event Status** 0 = No event occurred in endpoint 6. 1 = USB event occurred on Endpoint 6, check USBD_EPSTS0[27:24] to know which kind of USB event was occurred, cleared by write 1 to USBD_INTSTS[22] or USBD_INTSTS[1].
[21]	**EPEVT5**	**Endpoint 5's USB Event Status** 0 = No event occurred in endpoint 5. 1 = USB event occurred on Endpoint 5, check USBD_EPSTS0[23:20] to know which kind of USB event was occurred, cleared by write 1 to USBD_INTSTS[21] or USBD_INTSTS[1].
[20]	**EPEVT4**	**Endpoint 4's USB Event Status** 0 = No event occurred in endpoint 4. 1 = USB event occurred on Endpoint 4, check USBD_EPSTS0[19:16] to know which kind of USB event was occurred, cleared by write 1 to USBD_INTSTS[20] or USBD_INTSTS[1].
[19]	**EPEVT3**	**Endpoint 3's USB Event Status** 0 = No event occurred in endpoint 3. 1 = USB event occurred on Endpoint 3, check USBD_EPSTS0[15:12] to know which kind of USB event was occurred, cleared by write 1 to USBD_INTSTS[19] or USBD_INTSTS[1].

[18]	**EPEVT2**	**Endpoint 2's USB Event Status** 0 = No event occurred in endpoint 2. 1 = USB event occurred on Endpoint 2, check USBD_EPSTS0[11:8] to know which kind of USB event was occurred, cleared by write 1 to USBD_INTSTS[18] or USBD_INTSTS[1].
[17]	**EPEVT1**	**Endpoint 1's USB Event Status** 0 = No event occurred in endpoint 1. 1 = USB event occurred on Endpoint 1, check USBD_EPSTS0[7:4] to know which kind of USB event was occurred, cleared by write 1 to USBD_INTSTS[17] or USBD_INTSTS[1].
[16]	**EPEVT0**	**Endpoint 0's USB Event Status** 0 = No event occurred in endpoint 0. 1 = USB event occurred on Endpoint 0, check USBD_EPSTS0[3:0] to know which kind of USB event was occurred, cleared by write 1 to USBD_INTSTS[16] or USBD_INTSTS[1].
[15:5]	**Reserved**	Reserved.
[4]	**SOFIF**	**Start of Frame Interrupt Status** 0 = SOF event does not occur. 1 = SOF event occurred, cleared by write 1 to USBD_INTSTS[4].
[3]	**NEVWKIF**	**No-event-wake-up Interrupt Status** 0 = NEVWK event does not occur. 1 = No-event-wake-up event occurred, cleared by write 1 to USBD_INTSTS[3].
[2]	**VBDETIF**	**VBUS Detection Interrupt Status** 0 = There is not attached/detached event in the USB. 1 = There is attached/detached event in the USB bus and it is cleared by write 1 to USBD_INTSTS[2].
[1]	**USBIF**	**USB Event Interrupt Status** The USB event includes the SETUP Token, IN Token, OUT ACK, ISO IN, or ISO OUT events in the bus. 0 = No USB event occurred. 1 = USB event occurred, check EPSTS0~7[2:0] to know which kind of USB event was occurred, cleared by write 1 to USBD_INTSTS[1] or EPSTS0~7 and SETUP (USBD_INTSTS[31]).
[0]	**BUSIF**	**BUS Interrupt Status** The BUS event means that there is one of the suspense or the resume function in the bus. 0 = No BUS event occurred. 1 = Bus event occurred; check USBD_ATTR[3:0] to know which kind of bus event was occurred, cleared by write 1 to USBD_INTSTS[0].

USB Device Function Address Register (USBD_FADDR)

A 7-bit value is used as the address of a device on the USB BUS.

Register	Offset	R/W	Description	Reset Value
USBD_FADDR	USBD_BA+0x008	R/W	USB Device Function Address Register	0x0000_0000

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

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6:0]	**FADDR**	USB Device Function Address

USB Endpoint Status Register (USBD_EPSTS)

Register	Offset	R/W	Description	Reset Value
USBD_EPSTS	USBD_BA+0x00C	R	USB Device Endpoint Status Register	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**OV**	**Overrun** It indicates that the received data is over the maximum payload number or not. if received data is over the maximum payload number, the extra data will be ignored. 0 = No overrun. 1 = Out Data is more than the Max Payload in MXPLD register or the Setup Data is more than 8 Bytes.
[6:0]	**Reserved**	Reserved.

USB Bus Status and Attribution Register (USBD_ATTR)

Register	Offset	R/W	Description	Reset Value
USBD_ATTR	USBD_BA+0x010	R/W	USB Device Bus Status and Attribution Register	0x0000_0040

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**L1RESUME**	**L1SUSPEND**	**LPMACK**	**BYTEM**	**PWRDN**	**DPPUEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
USBEN	**Reserved**	**RWAKEUP**	**PHYEN**	**TOUT**	**RESUME**	**SUSPEND**	**USBRST**

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13]	**L1RESUME**	**LPM L1 Resume (Read Only)** 0 = Bus no LPM L1 state resume. 1 = LPM L1 state Resume from LPM L1 state suspend.
[12]	**L1SUSPEND**	**LPM L1 Suspend (Read Only)** 0 = Bus no L1 state suspend. 1 = This bit is set by the hardware when LPM command to enter the L1 state is successfully received and acknowledged.
[11]	**LPMACK**	**LPM Token Acknowledge Enable Bit** The NYET/ACK will be returned only on a successful LPM transaction if no errors in both the EXT token and the LPM token and a valid bLinkState = 0001 (L1) is received, else ERROR and STALL will be returned automatically, respectively. 0= the valid LPM Token will be NYET. 1= the valid LPM Token will be ACK.
[10]	**BYTEM**	**CPU Access USB SRAM Size Mode Selection** 0 = Word mode: The size of the transfer from CPU to USB SRAM can be Word only. 1 = Byte mode: The size of the transfer from CPU to USB SRAM can be Byte only.
[9]	**PWRDN**	**Power-down PHY Transceiver, Low Active** 0 = Power-down related circuit of PHY transceiver. 1 = Turn-on related circuit of PHY transceiver.
[8]	**DPPUEN**	**Pull-up Resistor on USB_DP Enable Bit** 0 = Pull-up resistor in USB_D+ bus Disabled. 1 = Pull-up resistor in USB_D+ bus Active.
[7]	**USBEN**	**USB Controller Enable Bit** 0 = USB Controller Disabled. 1 = USB Controller Enabled.

[6]	**Reserved**	Reserved.
[5]	**RWAKEUP**	**Remote Wake-up** 0 = Release the USB bus from K state. 1 = Force USB bus to K (USB_D+ low, USB_D-: high) state, used for remote wake-up.
[4]	**PHYEN**	**PHY Transceiver Function Enable Bit** 0 = PHY transceiver function Disabled. 1 = PHY transceiver function Enabled.
[3]	**TOUT**	**Time-out Status (Read Only)** When USB Device controller after received setup token or out token, USB controller stay J state to wait data package. If the waiting time exceeds 18-bit length timing, TOUT flag will be generated. 0 = No time-out. 1 = No Bus response more than 18 bits time.
[2]	**RESUME**	**Resume Status (Read Only)** 0 = No bus resume. 1 = Resume from suspend.
[1]	**SUSPEND**	**Suspend Status (Read Only)** 0 = Bus no suspend. 1 = Bus idle more than 3ms, either cable is plugged off or host is sleeping.
[0]	**USBRST**	**USB Reset Status (Read Only)** 0 = Bus no reset. 1 = Bus reset when SE0 (single-ended 0) more than 2.5us.

USB Device VBUS Detection Register (USBD_VBUSDET)

Register	Offset	R/W	Description	Reset Value
USBD_VBUSDET	USBD_BA+0x014	R	USB Device VBUS Detection Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**VBUSDET**	**Device VBUS Detection** 0 = Controller is not attached to the USB host. 1 = Controller is attached to the USB host.

USB SETUP Token Buffer Segmentation Register (USBD_STBUFSEG)

Register	Offset	R/W	Description	Reset Value
USBD_STBUFSEG	USBD_BA+0x018	R/W	SETUP Token Buffer Segmentation Register	0x0000_0000

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

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8:3]	**STBUFSEG**	**SETUP Token Buffer Segmentation** It is used to indicate the offset address for the SETUP token with the USB Device SRAM starting address The effective starting address is USBD_SRAM address + {STBUFSEG, 3'b000} Where the USBD_SRAM address = USBD_BA+0x100h. **Note:** It is used for SETUP token only.
[2:0]	**Reserved**	Reserved.

USB Endpoint Status Register 0 (USBD_EPSTS0)

Register	Offset	R/W	Description	Reset Value
USBD_EPSTS0	USBD_BA+0x020	R	USB Device Endpoint Status Register 0	0x0000_0000

31	30	29	28	27	26	25	24
EPSTS7				EPSTS6			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
EPSTS5				EPSTS4			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
EPSTS3				EPSTS2			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EPSTS1				EPSTS0			

Bits	Description	
[31:28]	**EPSTS7**	**Endpoint 7 Status** These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[27:24]	**EPSTS6**	**Endpoint 6 Status** These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[23:20]	**EPSTS5**	**Endpoint 5 Status** These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.

[19:16]	**EPSTS4**	**Endpoint 4 Status** **These Bits Are Used To Indicate The Current Status Of This Endpoint** **0000 = In ACK.** **0001 = In NAK.** **0010 = Out Packet Data0 ACK.** **0110 = Out Packet Data1 ACK.** **0111 = Isochronous Transfer End.**
[15:12]	EPSTS3	Endpoint 3 Status These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[11:8]	EPSTS2	Endpoint 2 Status These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[7:4]	EPSTS1	Endpoint 1 Status These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.
[3:0]	EPSTS0	Endpoint 0 Status These bits are used to indicate the current status of this endpoint 0000 = In ACK. 0001 = In NAK. 0010 = Out Packet Data0 ACK. 0110 = Out Packet Data1 ACK. 0111 = Isochronous transfer end.

USB LPM Attribution Register (USBD_LPMATTR)

Register	Offset	R/W	Description	Reset Value
USBD_LPMATTR	USBD_BA+0x088	R	USB LPM Attribution Register	0x0000_0000

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

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**LPMRWAKUP**	**LPM Remote Wakeup** This bit contains the bRemoteWake value received with last ACK LPM Token
[7:4]	**LPMBESL**	**LPM Best Effort Service Latency** These bits contain the BESL value received with last ACK LPM Token
[3:0]	**LPMLINKSTS**	**LPM Link State** These bits contain the bLinkState received with last ACK LPM Token

USB Frame Number Register (USBD_FN)

Register	Offset	R/W	Description	Reset Value
USBD_FN	USBD_BA+0x08C	R	USB Frame Number Register	0x0000_0XXX

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

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10:0]	**FN**	**Frame Number** These bits contain the 11-bits frame number in the last received SOF packet.

USB Drive SE0 Register (USBD_SE0)

Register	Offset	R/W	Description	Reset Value
USBD_SE0	USBD_BA+0x090	R/W	USB Device Drive SE0 Control Register	0x0000_0001

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**SE0**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**SE0**	**Drive Single Ended Zero in USB Bus** The Single Ended Zero (SE0) is when both lines (USB_D+ and USB_D-) are being pulled low. 0 = Normal operation. 1 = Force USB PHY transceiver to drive SE0.

USB Buffer Segmentation Register (USB_BUFSEGx)

Register	Offset	R/W	Description	Reset Value
USBD_BUFSEG0	USBD_BA+0x500	R/W	Endpoint 0 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG1	USBD_BA+0x510	R/W	Endpoint 1 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG2	USBD_BA+0x520	R/W	Endpoint 2 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG3	USBD_BA+0x530	R/W	Endpoint 3 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG4	USBD_BA+0x540	R/W	Endpoint 4 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG5	USBD_BA+0x550	R/W	Endpoint 5 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG6	USBD_BA+0x560	R/W	Endpoint 6 Buffer Segmentation Register	0x0000_0000
USBD_BUFSEG7	USBD_BA+0x570	R/W	Endpoint 7 Buffer Segmentation Register	0x0000_0000

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

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8:3]	**BUFSEG**	**Endpoint Buffer Segmentation** It is used to indicate the offset address for each endpoint with the USB SRAM starting address The effective starting address of the endpoint is USBD_SRAM address + { BUFSEG, 3'b000} Where the USBD_SRAM address = USBD_BA+0x100h. Refer to the section 6.22.5.7 for the endpoint SRAM structure and its description.
[2:0]	**Reserved**	Reserved.

USB Maximal Payload Register (USB_MXPLDx)

Register	Offset	R/W	Description	Reset Value
USBD_MXPLD0	USBD_BA+0x504	R/W	Endpoint 0 Maximal Payload Register	0x0000_0000
USBD_MXPLD1	USBD_BA+0x514	R/W	Endpoint 1 Maximal Payload Register	0x0000_0000
USBD_MXPLD2	USBD_BA+0x524	R/W	Endpoint 2 Maximal Payload Register	0x0000_0000
USBD_MXPLD3	USBD_BA+0x534	R/W	Endpoint 3 Maximal Payload Register	0x0000_0000
USBD_MXPLD4	USBD_BA+0x544	R/W	Endpoint 4 Maximal Payload Register	0x0000_0000
USBD_MXPLD5	USBD_BA+0x554	R/W	Endpoint 5 Maximal Payload Register	0x0000_0000
USBD_MXPLD6	USBD_BA+0x564	R/W	Endpoint 6 Maximal Payload Register	0x0000_0000
USBD_MXPLD7	USBD_BA+0x574	R/W	Endpoint 7 Maximal Payload Register	0x0000_0000

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

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8:0]	**MXPLD**	**Maximal Payload** Define the data length which is transmitted to host (IN token) or the actual data length which is received from the host (OUT token). It also used to indicate that the endpoint is ready to be transmitted in IN token or received in OUT token. (1) When the register is written by CPU, For IN token, the value of MXPLD is used to define the data length to be transmitted and indicate the data buffer is ready. For OUT token, it means that the controller is ready to receive data from the host and the value of MXPLD is the maximal data length comes from host. (2) When the register is read by CPU, For IN token, the value of MXPLD is indicated by the data length be transmitted to host For OUT token, the value of MXPLD is indicated the actual data length receiving from host. **Note:** Once MXPLD is written, the data packets will be transmitted/received immediately after IN/OUT token arrived.

USB Configuration Register (USB_CFGx)

Register	Offset	R/W	Description	Reset Value
USBD_CFG0	USBD_BA+0x508	R/W	Endpoint 0 Configuration Register	0x0000_0000
USBD_CFG1	USBD_BA+0x518	R/W	Endpoint 1 Configuration Register	0x0000_0000
USBD_CFG2	USBD_BA+0x528	R/W	Endpoint 2 Configuration Register	0x0000_0000
USBD_CFG3	USBD_BA+0x538	R/W	Endpoint 3 Configuration Register	0x0000_0000
USBD_CFG4	USBD_BA+0x548	R/W	Endpoint 4 Configuration Register	0x0000_0000
USBD_CFG5	USBD_BA+0x558	R/W	Endpoint 5 Configuration Register	0x0000_0000
USBD_CFG6	USBD_BA+0x568	R/W	Endpoint 6 Configuration Register	0x0000_0000
USBD_CFG7	USBD_BA+0x578	R/W	Endpoint 7 Configuration Register	0x0000_0000

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

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**CSTALL**	**Clear STALL Response** 0 = Disable the device to clear the STALL handshake in setup stage. 1 = Clear the device to response STALL handshake in setup stage.
[8]	**Reserved**	Reserved.
[7]	**DSQSYNC**	**Data Sequence Synchronization** 0 = DATA0 PID. 1 = DATA1 PID. **IN Token Transaction:** This bit is used to specify the DATA0 or DATA1 PID in the following IN token transaction. Hardware will toggle automatically in IN token base on it. **OUT Token Transaction:** This bit is used to specify the DATA0 or DATA1 PID in the following OUT token transaction. Hardware will toggle automatically OUT token base on this bit.
[6:5]	**STATE**	**Endpoint STATE** 00 = Endpoint is Disabled.

		01 = Out endpoint. 10 = IN endpoint. 11 = Undefined.
[4]	**ISOCH**	**Isochronous Endpoint** This bit is used to set the endpoint as Isochronous endpoint, no handshake. 0 = No Isochronous endpoint. 1 = Isochronous endpoint.
[3:0]	**EPNUM**	**Endpoint Number** These bits are used to define the endpoint number of the current endpoint

USB Extra Configuration Register (USB_CFGPx)

Register	Offset	R/W	Description	Reset Value
USBD_CFGP0	USBD_BA+0x50C	R/W	Endpoint 0 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP1	USBD_BA+0x51C	R/W	Endpoint 1 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP2	USBD_BA+0x52C	R/W	Endpoint 2 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP3	USBD_BA+0x53C	R/W	Endpoint 3 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP4	USBD_BA+0x54C	R/W	Endpoint 4 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP5	USBD_BA+0x55C	R/W	Endpoint 5 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP6	USBD_BA+0x56C	R/W	Endpoint 6 Set Stall and Clear In/Out Ready Control Register	0x0000_0000
USBD_CFGP7	USBD_BA+0x57C	R/W	Endpoint 7 Set Stall and Clear In/Out Ready Control Register	0x0000_0000

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

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**SSTALL**	**Set STALL** 0 = Disable the device to response STALL. 1 = Set the device to respond STALL automatically.
[0]	**CLRRDY**	**Clear Ready** When the USBD_MXPLDx register is set by user, it means that the endpoint is ready to transmit or receive data. If the user wants to disable this transaction before the transaction start, users can set this bit to 1 to disable it and it is auto clear to 0. For IN token, write '1' to clear the IN token had ready to transmit the data to USB. For OUT token, write '1' to clear the OUT token had ready to receive the data from USB. This bit is write 1 only and is always 0 when it is read back.

6.23 CRC Controller (CRC)

6.23.1 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.

6.23.2 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

6.23.3 Block Diagram

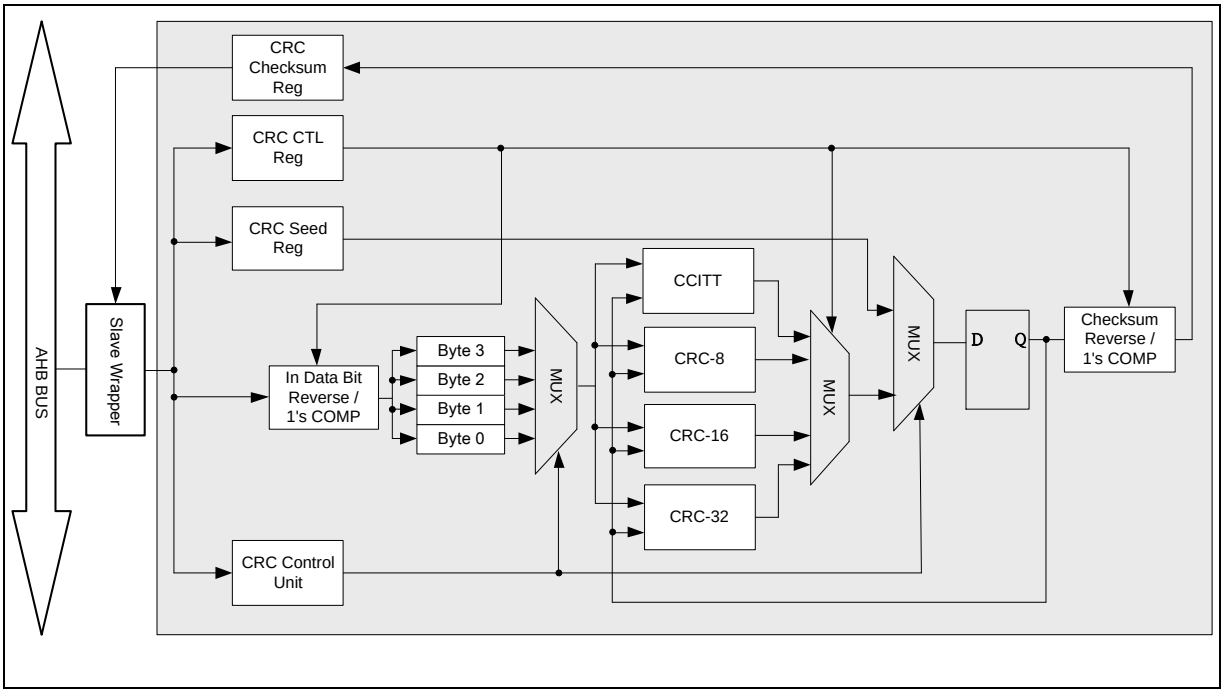


Figure 6.23-1 CRC Generator Block Diagram

6.23.4 Basic Configuration

- Clock Source Configuration
  - Enable CRC peripheral clock in CRCCKEN (CLK_AHBCLK[7]).
- Reset Configuration
  - Reset CRC controller in CRCRST (SYS_IPRST0[7]).

6.23.5 Functional Description

The 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).
2. Initial setting for CRC calculation.

- Configure 1's complement for CRC checksum by setting CHKSFMT (CRC_CTL[27] Checksum 1's Complement).
- Configure bit order reverse for CRC checksum by setting CHKSREV (CRC_CTL[25] Checksum Bit Order Reverse). The functional block is also shown in Figure 6.23-2Figure 6.23-2 CHECKSUM Bit Order Reverse Functional Block
- Configure 1's complement for CRC write data by setting DATFMT (CRC_CTL[26] Write Data 1's Complement).
- Configure bit order reverse for CRC write data per byte by setting DATREV (CRC_CTL[24] Write Data Bit Order Reverse). The functional block is also shown in Figure 6.23-3.

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.

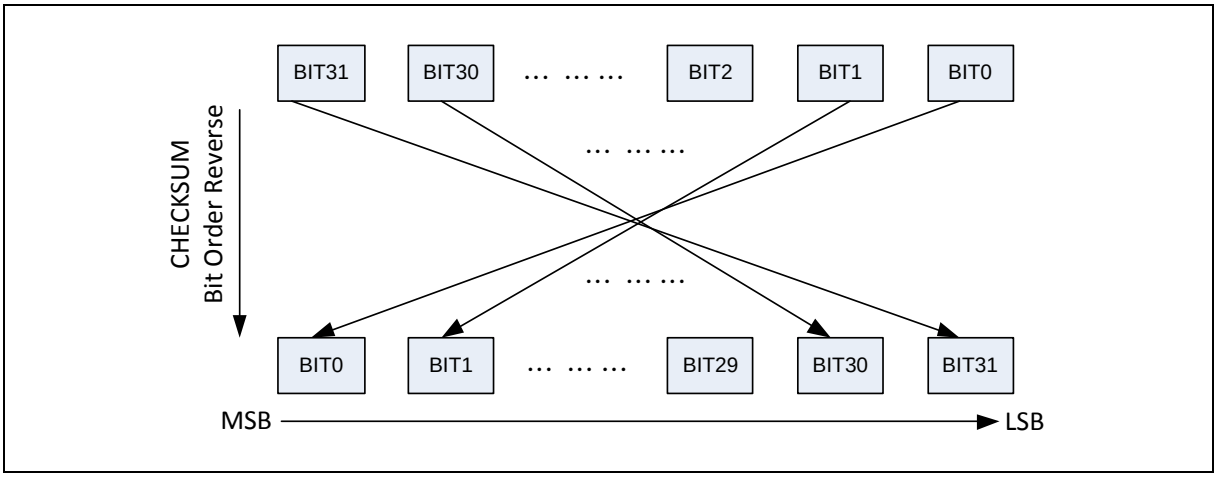


Figure 6.23-2 CHECKSUM Bit Order Reverse Functional Block

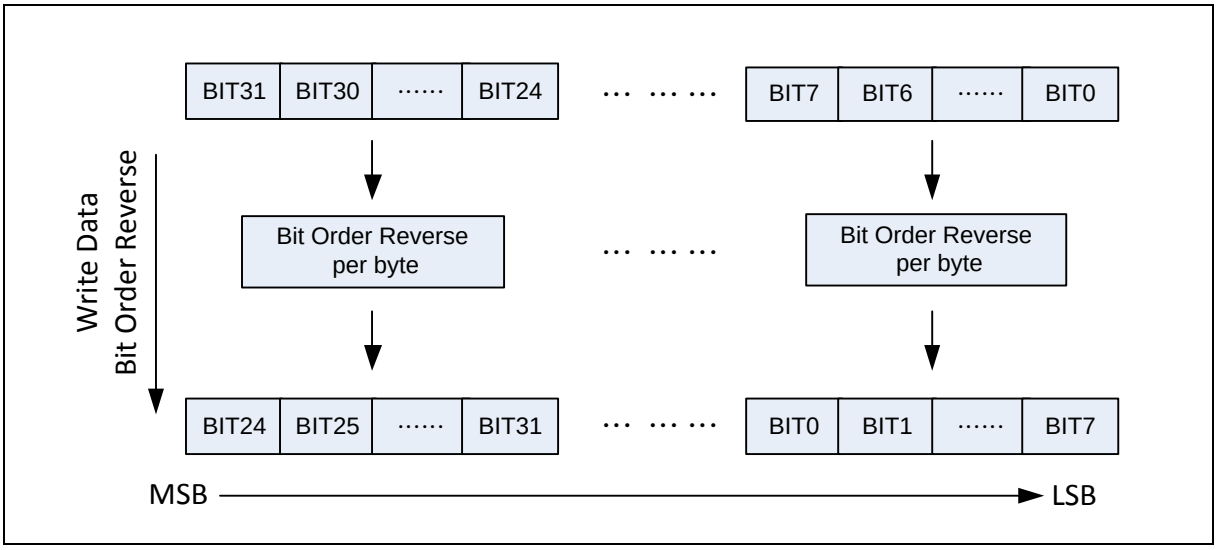


Figure 6.23-3 Write Data Bit Order Reverse Functional Block

6.23.6 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

6.23.7 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.

6.24 Hardware Divider (HDIV)

6.24.1 Overview

The hardware divider (HDIV) is useful to the high performance application. The hardware divider is a signed, integer divider with both quotient and remainder outputs.

6.24.2 Features

- Signed (two's complement) integer calculation
- 32-bit dividend with 16-bit divisor calculation capacity
- 32-bit quotient and 32-bit remainder outputs (16-bit remainder with sign extends to 32-bit)
- Divided by zero warning flag
- Write divisor to trigger calculation

6.24.3 Basic Configuration

Before using the hardware divider, the clock of hardware divider must be enabled. To enable hardware divider, HDIV_EN on AHBCLK[4] needs to be set to 1.

6.24.4 Functional Description

To use hardware divider, it needs to set dividend first. Then setting divisor and the hardware divider will trigger calculation automatically after divisor written. The calculation results including the quotient and remainder could be obtained by reading DIVQUO and DIVREM register. User can read quotient and remainder after one cycle of writing the divisor.

The DIV0 flag of DIVSTS will be set if divisor is 0.

The dividend is 32-bit signed integer and divisor is 16-bit signed integer. The quotient is 32-bit signed integer and the remainder is 16-bit signed integer.

Figure 6.24-1 shows the operation flow of hardware divider. To calculate X / Y, CPU needs to write X to DIVIDEND register, and then write Y to DIVISOR. CPU can read DIVQUO and DIVREM registers to get calculation results after DIVISOR is written.

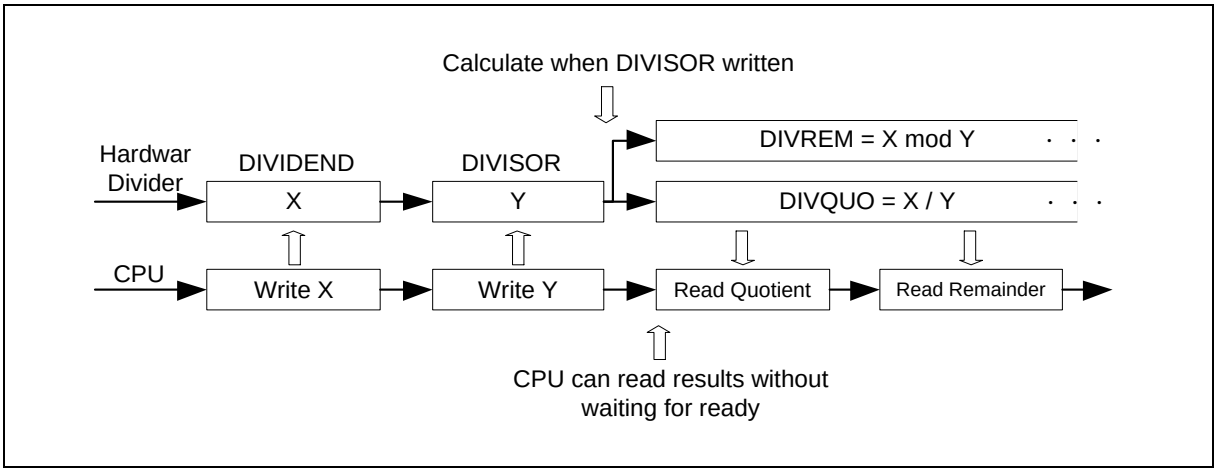


Figure 6.24-1 Hardware Divider Operation Flow

6.24.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
HDIV Base Address: **HDIV_BA = 0x4001_4000**				
DIVIDEND	HDIV_BA+0x00	R/W	Dividend Source Register	0x0000_0000
DIVISOR	HDIV_BA+0x04	R/W	Divisor Source Resister	0x0000_FFFF
DIVQUO	HDIV_BA+0x08	R/W	Quotient Result Resister	0x0000_0000
DIVREM	HDIV_BA+0x0C	R/W	Remainder Result Register	0x0000_0000
DIVSTS	HDIV_BA+0x10	R	Divider Status Register	0x0000_0001

6.24.6 Register Description

Dividend Source Register (DIVIDEND)

Register	Offset	R/W	Description	Reset Value
DIVIDEND	HDIV_BA+0x00	R/W	Dividend Source Register	0x0000_0000

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

Bits	Description	
[31:0]	**DIVIDEND**	**Dividend Source** This register is given the dividend of divider before calculation starts.

Divisor Source Register (DIVISOR)

Register	Offset	R/W	Description	Reset Value
DIVISOR	HDIV_BA+0x04	R/W	Divisor Source Resister	0x0000_FFFF

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**DIVISOR**	**Divisor Source** This register is provided with the divisor of divider before calculation starts. **Note:** When this register is written, hardware divider will start calculation.

Quotient Result Register (DIVQUO)

Register	Offset	R/W	Description	Reset Value
DIVQUO	HDIV_BA+0x08	R/W	Quotient Result Resister	0x0000_0000

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

Bits	Description	
[31:0]	**QUOTIENT**	**Quotient Result** This register holds the quotient result of divider after calculation is complete.

Remainder Result Register (DIVREM)

Register	Offset	R/W	Description	Reset Value
DIVREM	HDIV_BA+0x0C	R/W	Remainder Result Register	0x0000_0000

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

Bits	Description	
[31:16]	**REMAINDER[31:16]**	**Sign Extension of REMAINDER[15:0]** The remainder of hardware divider is 16-bit sign integer (REMAINDER[15:0]) with sign extension (REMAINDER[31:16]) to 32-bit integer.
[15:0]	**REMAINDER[15:0]**	**Remainder Result** This register holds the remainder result of divider after calculation is complete.

Divider Status Register (DIVSTS)

Register	Offset	R/W	Description	Reset Value
DIVSTS	HDIV_BA+0x10	R	Divider Status 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						**DIV0**	**Reserved**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**DIV0**	**Divisor Zero Warning (Read Only)** 0 = The divisor is not 0. 1 = The divisor is 0. **Note:** The DIV0 flag is used to indicate divide-by-zero situation and updated whenever DIVISOR is written.
[0]	**Reserved**	Reserved.

6.25 Analog-to-Digital Converter (ADC)

6.25.1 Overview

The ADC contains one 12-bit successive approximation analog-to-digital converter (SAR A/D converter) with 16 input channels. The A/D converter supports four operation modes: Single, Burst, Single-cycle Scan and Continuous Scan mode. The A/D converter can be started by software, external pin (STADC), timer0~3 overflow pulse trigger, PWM trigger or BPWM trigger.

6.25.2 Features

- Operating voltage: 1.8V~3.6V.
- Analog input voltage: 0 ~ AV_{DD}.
- Supports external reference voltage from V_{REF} pin.
- 12-bit resolution and 10-bit accuracy is guaranteed.
- Up to 16 single-end analog input channels or 8 differential analog input channels.
- Maximum ADC peripheral clock frequency is 34 MHz.
- Up to 2 MSPS sampling rate.
- Scan on enabled channels
- Threshold voltage detection
- Four operation modes:
  - Single mode: A/D conversion is performed one time on a specified channel.
  - Burst mode: A/D converter samples and converts the specified single channel and sequentially stores the result in FIFO.
  - Single-cycle Scan mode: A/D conversion is performed only one cycle on all specified channels with the sequence from the smallest numbered channel to the largest numbered channel.
  - Continuous Scan mode: A/D converter continuously performs Single-cycle Scan mode until software stops A/D conversion.
- An A/D conversion can be started by:
  - Software Write 1 to ADST bit
  - External pin (STADC)
  - Timer 0~3 overflow pulse trigger
  - BPWM trigger
  - PWM trigger
- Each conversion result is held in data register of each channel with valid and overrun indicators.
- Conversion result can be compared with specified value and user can select whether to generate an interrupt when conversion result matches the compare register setting.
- Supports extend sample time function (0~255 ADC clock).
- One internal channel from band-gap voltage (V_{BG}).
- One internal channel from internal pull-up/down circuit.
- Supports PDMA transfer mode.
- Supports Calibration mode.

- Supports Floating Detect Function

Note1: ADC sampling rate = (ADC peripheral clock frequency) / (total ADC conversion cycle)

Note2: If the internal channel for band-gap voltage is active, the maximum sampling rate will be 300k SPS.

Note3: The ADC Clock frequency must be slower than or equal to PCLK.

	M032xC/D	M031xB/C/D/E M032x/E	M031xG/I M032xG/I
6.25.5.11 PWM trigger	-	●	●
6.25.5.12 BPWM trigger	●	-	●
6.25.5.17 Floating Detect Function	●	-	●

Table 6.25-1 ADC Features Comparison Table

6.25.3 Block Diagram

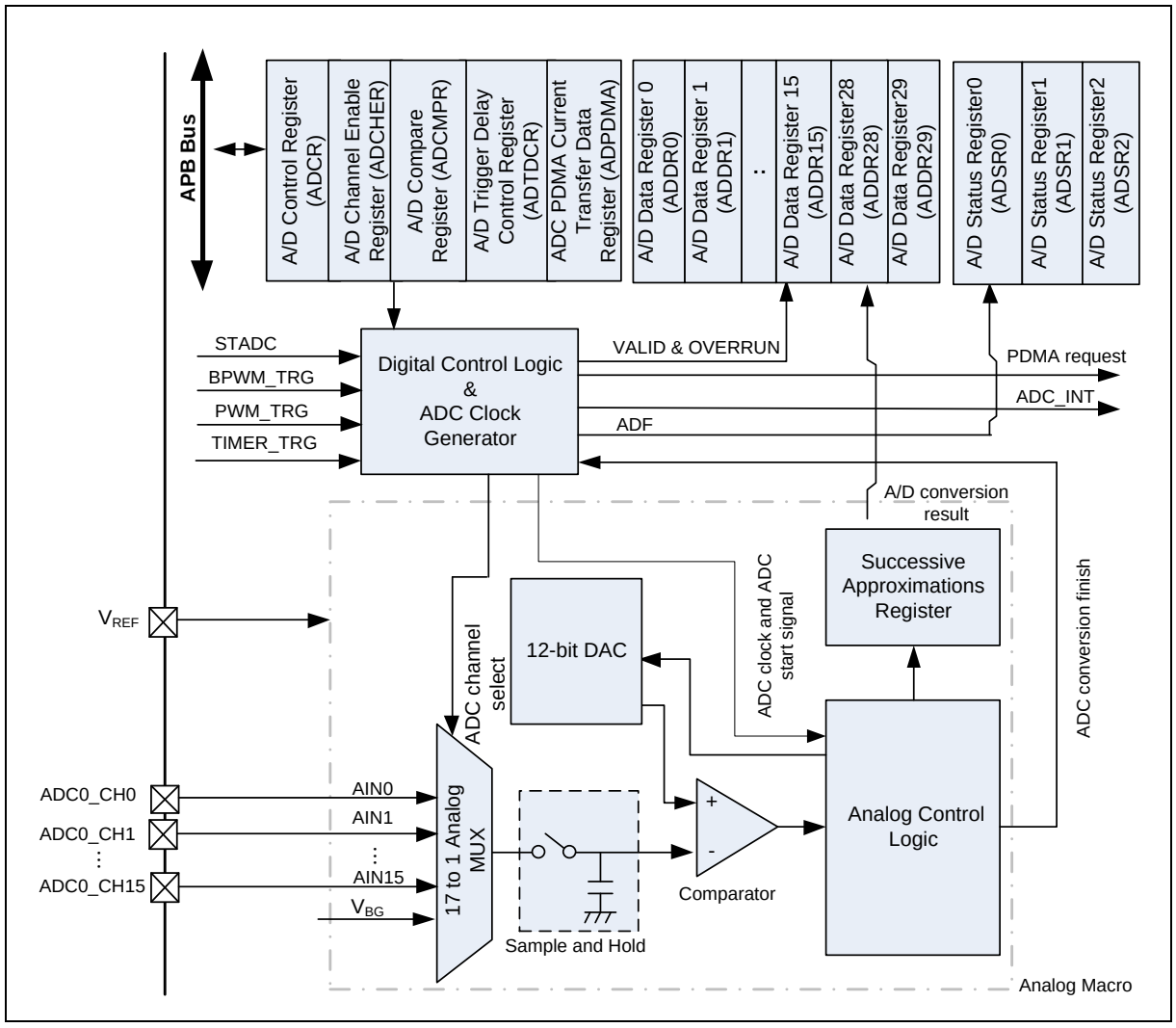


Figure 6.25-1 AD Controller Block Diagram

6.25.4 Basic Configuration

- Clock Source Configuration
  - Select the source of ADC peripheral clock on ADCSEL(CLKSEL2[21:20]).
  - Select the clock divider number of ADC peripheral clock on ADCDIV (CLKDIV0[23:16]).
  - Enable ADC peripheral clock in ADCCKEN (CLK_APBCLK0[28]).
- Reset Configuration

6.25.5 Functional Description

The A/D converter operates by successive approximation with 12-bit resolution. The ADC has four operation modes: Single, Burst, Single-cycle Scan mode and Continuous Scan mode. When user wants to change the operation mode or analog input channel, in order to prevent incorrect operation, software must clear ADST(ADCR[11]) bit to 0 in advance.

6.25.5.1 ADC peripheral Clock Generator

The maximum sampling rate is up to 2 MSPS. The ADC has four clock sources selected by ADCSEL (CLKSEL2[21:20]), the ADC peripheral clock frequency is divided by an 8-bit prescaler with the following formula:

ADC peripheral clock frequency = (ADC peripheral clock source frequency) / (ADCDIV+1); where the 8-bit ADCDIV is located in register CLKDIV0[23:16].

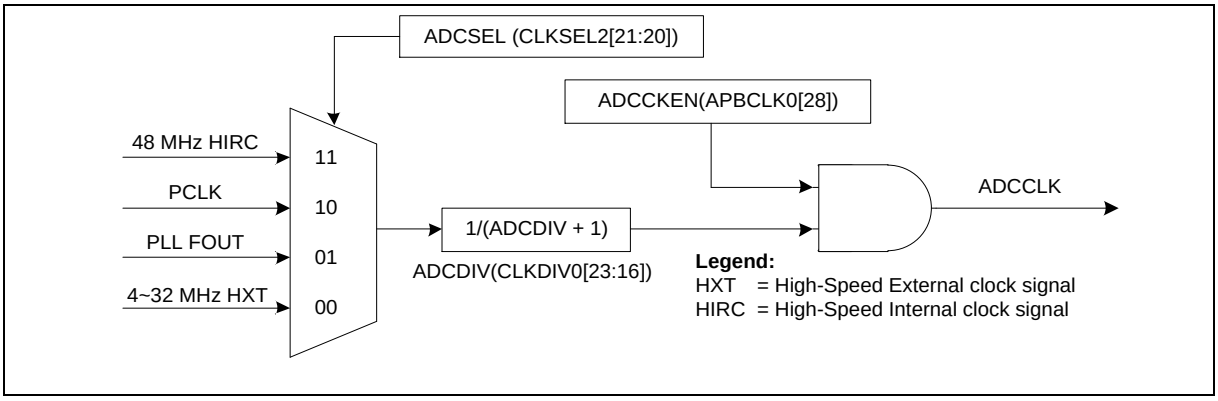


Figure 6.25-2 ADC Peripheral Clock Control

6.25.5.2 Single Mode

In Single mode, A/D conversion is performed only once on the specified single channel. The operations are as follows:

1. A/D conversion will be started when the ADST bit of ADCR register is set to 1 by software or external trigger input.
2. When A/D conversion is finished, the result is stored in the ADC data register corresponding to the channel.
3. After A/D conversion is finished, the ADF bit of ADSR0 register will be set to 1. If the ADIE bit of ADCR register is set to 1, the ADC interrupt will be asserted.
4. The ADST bit remains 1 during A/D conversion. When A/D conversion ends, the ADST bit is automatically cleared to 0 and the A/D converter enters idle state.

Note1: If software enables more than one channel in Single mode, only the channel with the smallest number will be selected and the other enabled channels will be ignored.

Note2: If ADST bit is cleared to 0 before ADC conversion done, ADC cannot finish the current conversion, the BUSY bit will be cleared to 0 immediately and A/D converter enters idle state directly.

An example timing diagram for Single mode is shown in Figure 6.25-3.

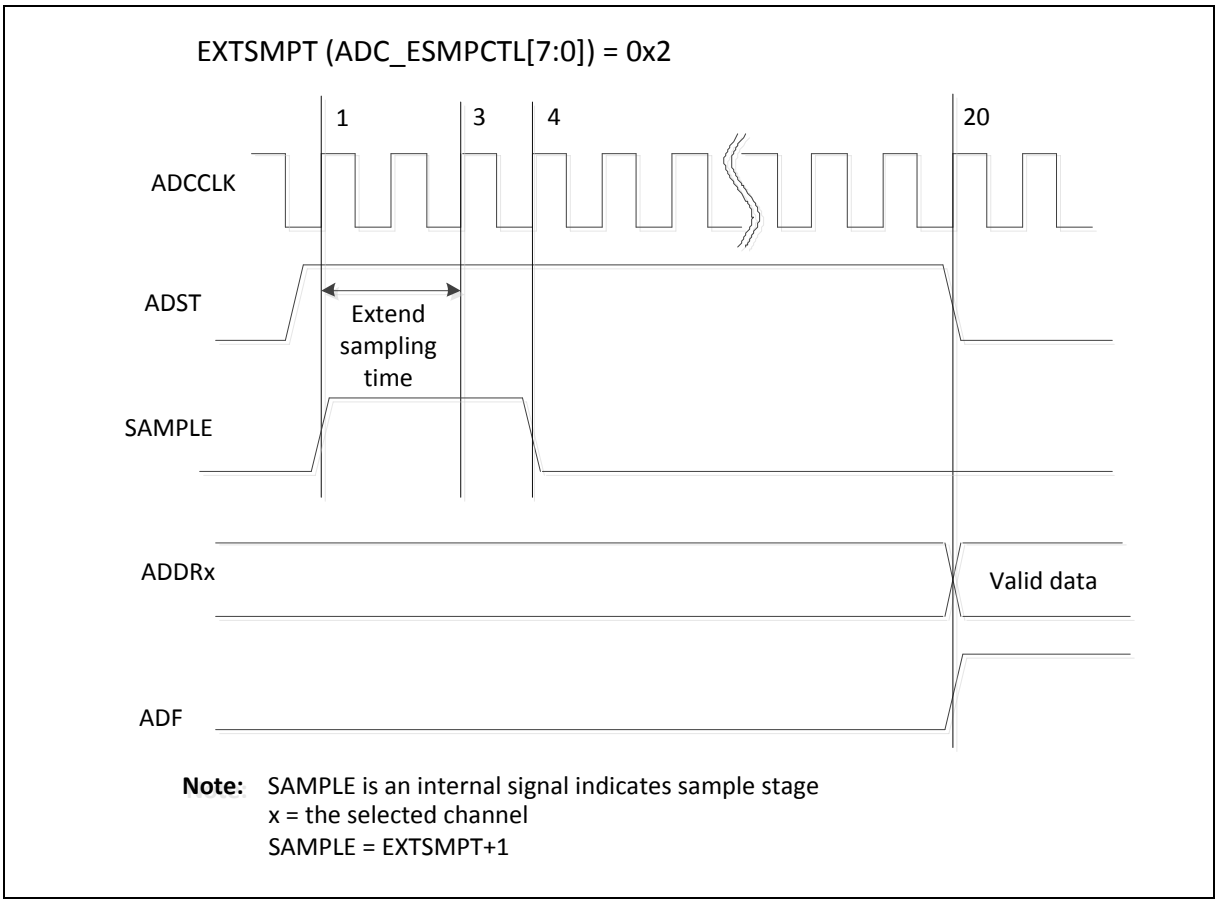


Figure 6.25-3 Single Mode Conversion Timing Diagram

6.25.5.3 Burst Mode

In Burst mode, A/D converter samples and converts the specified single channel and sequentially stores the result into FIFO (up to 8 samples). The operations are as follows:

1. When the ADST bit in ADCR register is set to 1 by software or external trigger input, A/D conversion is started on the enabled channel with the smallest number.
2. When A/D conversion for the specified channel is completed, the result is sequentially transferred to FIFO and can be accessed only from the ADC data register 0.
3. When more than or equal to 4 samples in FIFO, the ADF bit in ADSR0 register is set to 1. If the ADIE bit of ADCR register is set to 1 at this time, an ADC interrupt is requested after finishing the A/D conversion.
4. Steps 2 to 3 are repeated as long as the ADST bit remains 1. When the ADST bit is cleared to 0, ADC cannot finish the current conversion and A/D converter enters idle state directly

An example timing diagram for Burst mode is shown in Figure 6.25-4.

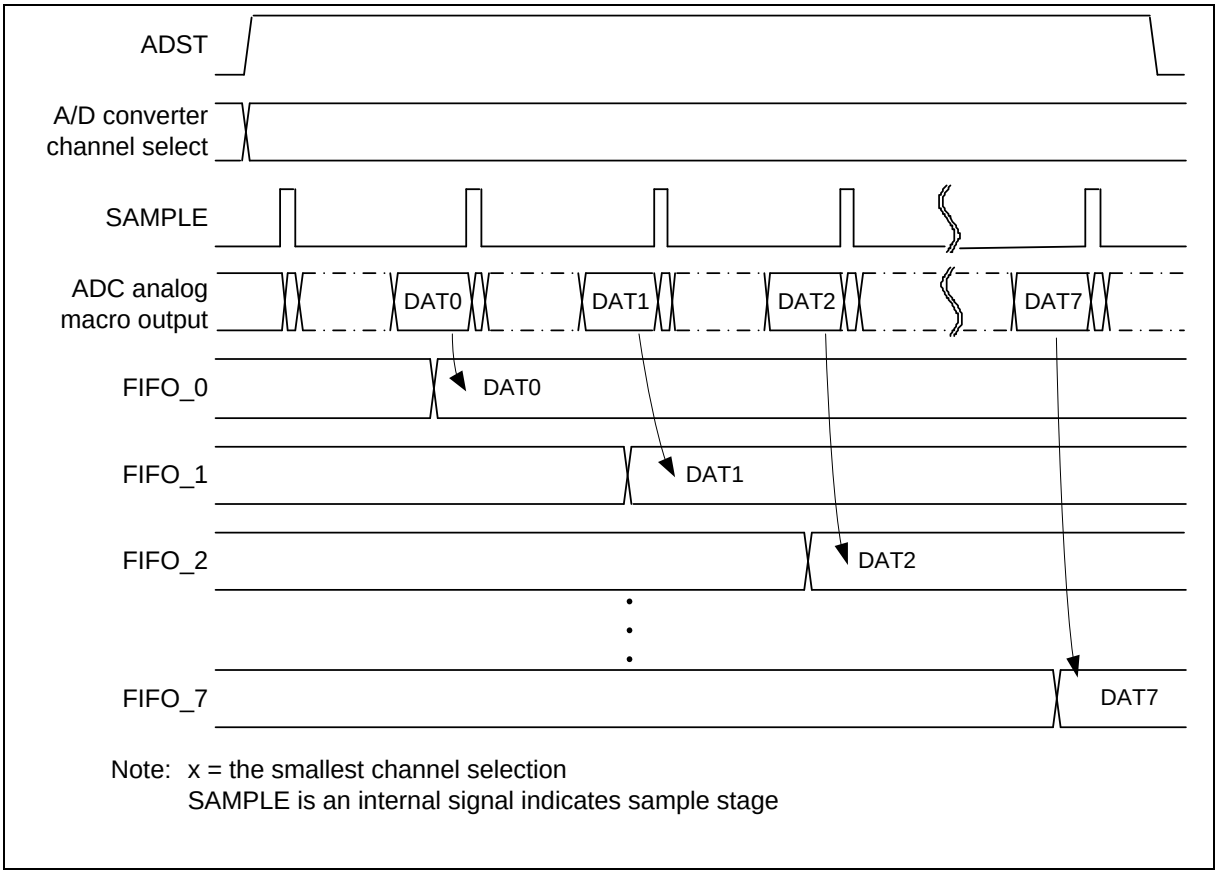


Figure 6.25-4 Burst Mode Conversion Timing Diagram

Note1: If software enables more than one channel in Burst mode, only the channel with the smallest number is converted and other enabled channels will be ignored.

Note2: User can obtain the conversion results by reading the ADC data register 0 (ADDR0) repeatedly until VALIDF (ADSR0[8]) turns to 0. For example, if there are 4 conversion results at FIFO, it needs to read ADC data register 0 (ADDR0) 4 times to get all of result.

6.25.5.4 Single-Cycle Scan Mode

In Single-cycle Scan mode, A/D converter samples and converts all of the specified channels once in the sequence from the smallest number enabled channel to the largest number enabled channel. Operations are as follows:

1. When the ADST bit in ADCR register is set to 1 by software or external trigger input, A/D conversion is started on the enabled channel with the smallest number.
2. When A/D conversion for each enabled channel is completed, the result is sequentially transferred to the ADC data register corresponding to each channel.
3. When the conversions of all the enabled channels are completed, the ADF bit in ADSR0 register is set to 1. If the ADC interrupt function is enabled, the ADC interrupt occurs.
4. After ADC finishes one cycle conversion, the ADST bit is automatically cleared to 0 and the A/D converter enters idle state. If ADST bit is cleared to 0 before all enabled ADC channels conversion done, ADC cannot finish the current conversion and A/D converter enters idle state directly.

An example timing diagram for Single-cycle Scan mode on enabled channels (0, 2, 3 and 7) is shown in Figure 6.25-5.

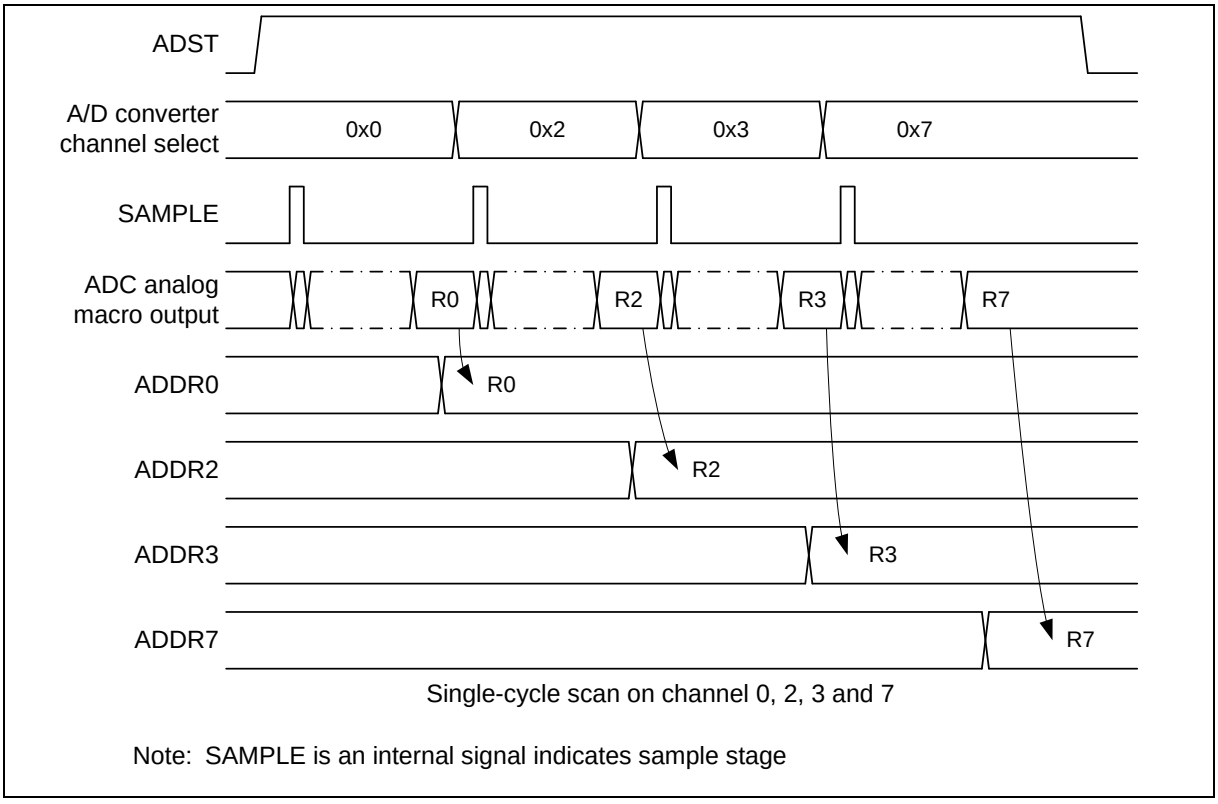


Figure 6.25-5 Single-Cycle Scan Mode on Enabled Channels Timing Diagram

6.25.5.5 A/D Extend Sampling Time

When A/D operates at high ADC 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 also set extend sampling time by writing EXTSMPT (ADC_ESMPCTL[7:0]) for each sample module. The range of extend sampling time is from 0 ~255 ADC clock.

6.25.5.6 Continuous Scan Mode

In Continuous Scan mode, A/D conversion is performed sequentially on the specified channels that enabled by CHEN bits in ADCHER register (maximum 16 channels for ADC). The operations are as follows:

1. When the ADST bit in ADCR register is set to 1 by software or external trigger input, A/D conversion is started on the enabled channel with the smallest number.
2. When A/D conversion for each enabled channel is completed, the result of each enabled channel is stored in the ADC data register corresponding to each enabled channel.
3. When A/D converter completes the conversions of all enabled channels sequentially, the ADF bit in ADSR0 register will be set to 1. If the ADC interrupt function is enabled, the ADC interrupt occurs. The conversion of the enabled channel with the smallest number will start again if software does not clear the ADST bit.
4. As long as the ADST bit remains at 1, the step 2 ~ 3 will be repeated. When ADST is cleared to 0, ADC cannot finish the current conversion and A/D converter enters idle state directly.

An example timing diagram for Continuous Scan mode on enabled channels (0, 2, 3 and 7) is shown in Figure 6.25-6.

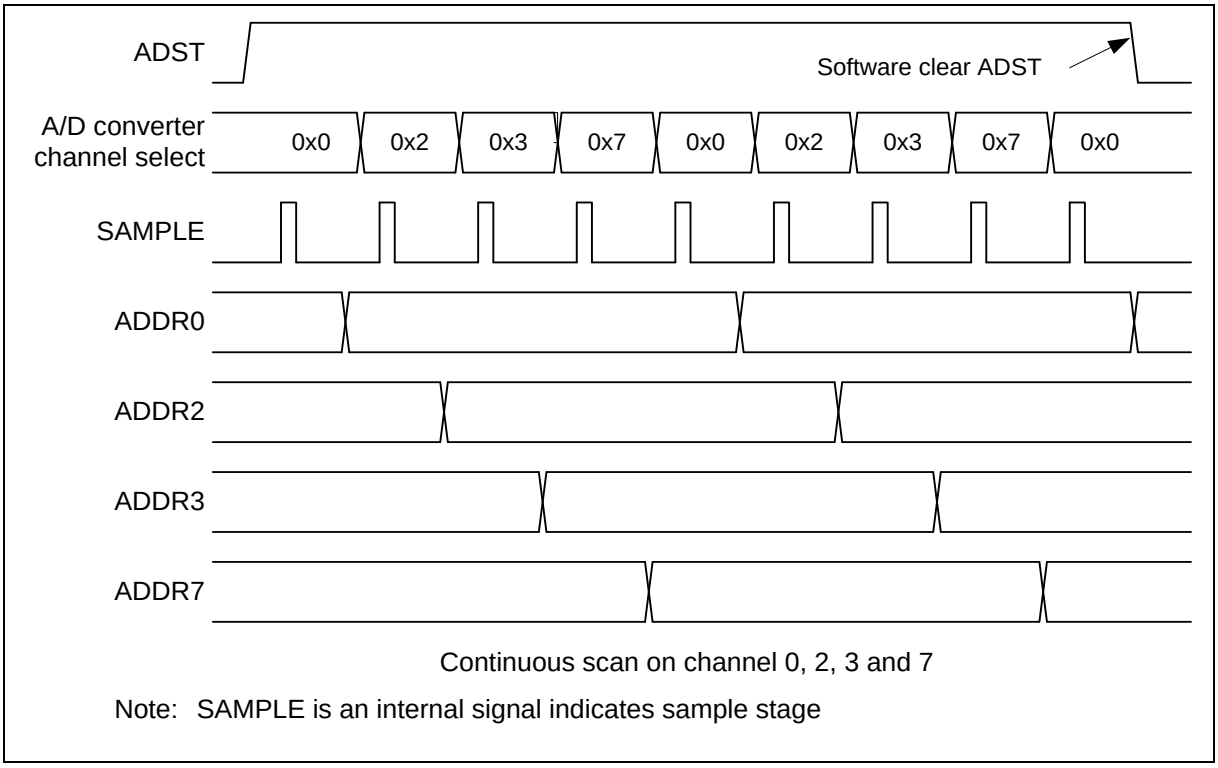


Figure 6.25-6 Continuous Scan Mode on Enabled Channels Timing Diagram

6.25.5.7 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. Setting CALEN(ADC_ADCALR[0]) bit to 1.
2. When the ADST bit in ADCR register is set to 1 by software, calibration is starting.
3. After calibration finish, the flag CALIF(ADC_ADCALSTSR[0]) will be set to 1.
4. After calibration finish, the mode will become Normal Mode automatically.

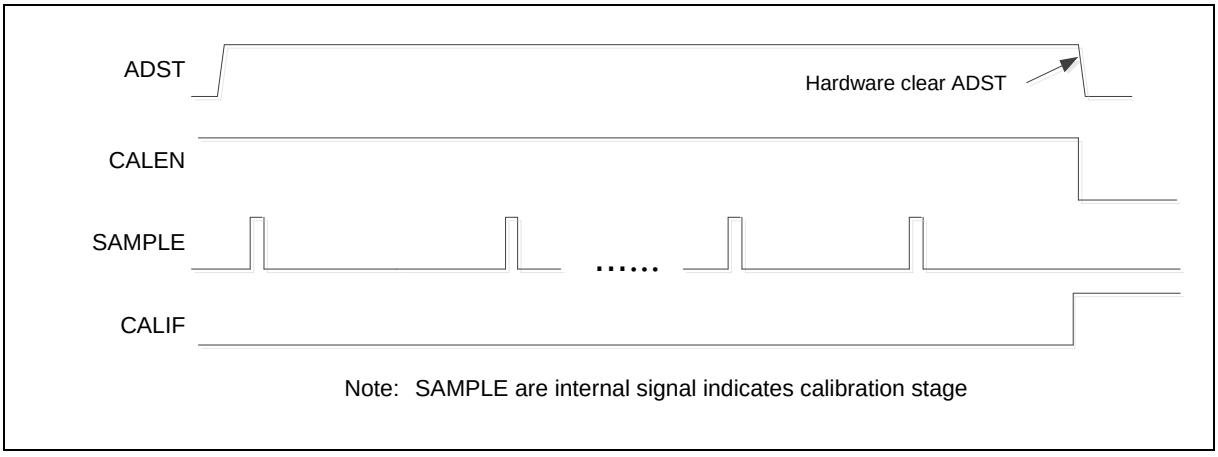


Figure 6.25-7 Calibration Mode With 16 time average

6.25.5.8 *External Trigger Input*

In Single-cycle Scan mode, A/D conversion can be triggered by external pin request. When the TRGEN bit of ADCR register is set to 1 to enable ADC external trigger function, setting the TRGS(ADCR[5:4]) bits to 2'b00 is to select external trigger input from the STADC pin. Software can set TRGCOND(ADCR[7:6]) to select trigger condition is falling/rising edge or low/high level. If level trigger condition is selected, the STADC pin must be kept at specified state at least 8 PCLKs. The ADST bit will be set to 1 at the 9th PCLK and start to convert. Conversion will keep going if external trigger input is kept at active state in level trigger mode. It is stopped only when external condition trigger condition disappears. If edge trigger condition is selected, the high and low state must be kept at least 4 PCLKs. Pulse that is shorter than this specification will be ignored.

Note: User enables the external trigger function or enables ADC must be at least 4 PCLKs after enabling ADC peripheral clock.

6.25.5.9 *Internal Reference Voltage*

The band-gap voltage reference (V_{BG}) is an internal fixed reference regardless of the power supply variation. The V_{BG} output is internally connected to the ADC channel multiplexer and analog comparator (ACMP) negative input side. Therefore, the voltage of AV_{DD} can be calculated by the A/D conversion result of V_{BG}.

V_{BG} can be used to calculate the AV_{DD} voltage when AV_{DD} is directly connected to the power supply. Compared to the voltage divider circuit, using V_{BG} can reduce the external components and power consumption.

The V_{BG} of each chip might be slightly different due to the process variation. Therefore, the calculated AV_{DD} might have some deviation. The M031/M032 series had built-in V_{BG} A/D conversion value (when AV_{DD} = 3072 mV) when chips are shipped from the factory. With the built-in V_{BG} A/D conversion value and the current V_{BG} A/D conversion value, users can calculate AV_{DD} in a more accurately way. It is more accurate in comparison with using only the V_{BG} A/D conversion value.

For applications with battery power detection, users can follow the steps below to calculate a more accurate battery voltage.

1. Use the ISP Command - Read Unique ID command (FMC_ISPCMD = 0x04) to read the built-in V_{BG} ADC conversion result ($ADC_{VBG\_Built-In}$, FMC_ISPDAT[11:0]).
2. Use V_{BG} as the ADC input channel (CHEN [29]), and then start A/D conversion to get the current V_{BG} A / D conversion value (ADC_{VBG}).
3. Use $ADC_{VBG\_Built-In}$, ADC_{VBG} and the following formula to get AV_{DD}.

   AV_{DD} = 3072 mV * $ADC_{VBG\_Built-In}$ / ADC_{VBG}

For example, if ADC_{VBG} is 0x61C (1564) and $ADC_{VBG\_Built-In}$ is 0x68D (1677), AV_{DD} can be calculated by the formula: 3072 mV * 1677 / 1564 = 3293 mV.

6.25.5.10 Timer Trigger

There are 4 Timer trigger sources for ADC. When the TRGEN bit of ADCR register is set to high to enable ADC external hardware trigger function, setting the TRGS (ADCR[5:4]) bits to 2'b01 is to select external hardware trigger input source from Timer trigger. The detail trigger conditions of Timer are described at TIMER0_CTL ~ TIMER3_CTL register.

6.25.5.11 PWM Trigger

In all operation mode, A/D conversion can be triggered by PWM request. When the TRGEN bit of ADCR register is set to high to enable ADC external hardware trigger function, setting the TRGS(ADCR[5:4]) bits to 2'b11 is to select external hardware trigger input source from PWM trigger.

6.25.5.12 BPWM Trigger

In all operation mode, A/D conversion can be triggered by BPWM request. When the TRGEN bit of ADCR register is set to high to enable ADC external hardware trigger function, setting the TRGS(ADCR[5:4]) bits to 2'b10 is to select external hardware trigger input source from BPWM trigger.

6.25.5.13 Conversion Result Monitor by Compare Mode Function

The M031/M032 series ADC controller provides two compare registers, ADCMPRx(x=0,1), to monitor maximum two specified channels. Software can select which channel to be monitored by setting CMPCH (ADCMPRx[7:3]). CMPCOND (ADCMPRx[2]) bit is used to determine the compare condition. If CMPCOND bit is cleared to 0, the internal match counter will increase one when the conversion result is less than the value specified in CMPD (ADCMPRx[27:16]); if CMPCOND bit is set to 1, the internal match counter will increase one when the conversion result is greater than or equal to the value specified in CMPD (ADCMPRx[27:16]). When the conversion of the channel specified by CMPCH(ADCMPRx[7:3]) is completed, the comparing action will be triggered one time automatically. When the compare result meets the setting, compare match counter will increase 1, otherwise, the compare match counter will be cleared to 0. When the match counter reaches the setting of (CMPMATCNT+1) then CMPF0/1 (ADSR0[1]/[2]) bit will be set to 1, if CMPIE (ADCMPRx[1]) bit is set then an ADC interrupt request is generated. Software can use it to monitor the external analog input pin voltage transition in scan mode without imposing a load on software. The detailed logic diagram is shown in Figure 6.25-8.

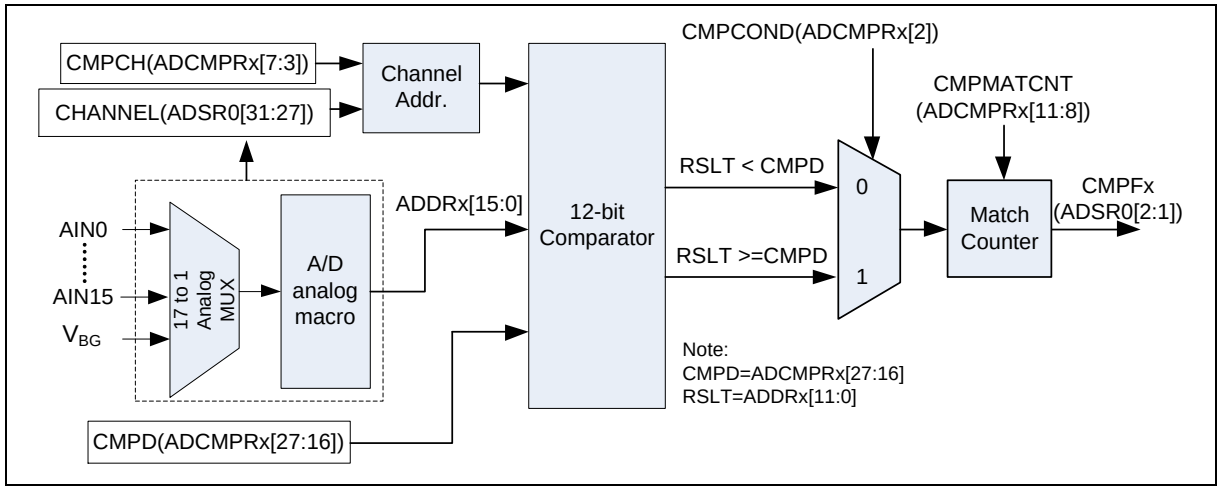


Figure 6.25-8 A/D Conversion Result Monitor Logic Diagram

6.25.5.14 Compare Window Mode

The ADC controller supports a compare window mode. User can set CMPWEN (ADCMPR0[15]) to enable this function. If user enables this function, CMPF0 (ADSR0[1]) will be set only when compared conditions of two conversion result monitor logic are matched and CMPF1 (ADSR0[2]) will always be zero. The range of compare window is between CMPD (ADCMPR0[27:16]) and CMPD (ADCMPR1[27:16]).

6.25.5.15 PDMA Transfer Mode

When A/D conversion is finished, the conversion result will be loaded into ADDRx(x=0~15, 29) register and VALID(ADDRx[17]) bit will be set to 1. If the PTEN(ADCR[9]) bit is set, ADC controller will generate a request to PDMA. User can use PDMA to transfer the conversion results to a user-specified memory space without CPU's intervention. The source address of PDMA operation is fixed

at ADPDMA, no matter what channels was selected. When PDMA is transferring the conversion result, ADC will continue converting the next selected channel if the operation mode of ADC is burst mode, single scan mode or continuous scan mode. User can monitor current PDMA transfer data through reading ADPDMA register. If ADC completes the conversion of a selected channel and the last conversion result of the same channel has not been transferred by PDMA, OVERRUN(ADSR2[31:0]) bit of the corresponding channel will be set and the last ADC conversion result will be overwritten by the new ADC conversion result. PDMA will transfer the latest data of selected channels to the user-specified destination address.

6.25.5.16 Interrupt Sources

There are four interrupt sources of ADC interrupt. When an ADC operation mode finishes its conversion, the A/D conversion end flag, ADF(ADSR0[0]), will be set to 1. The CMPF0(ADSR0[1]) and CMPF1(ADSR0[2]) are the compare flags of compare function. When the conversion result meets the settings of ADCMPR0/1 registers, the corresponding flag will be set to 1. The CALIF(ADCALSTSR[0]) is flag of calibration function. When calibration finish, this flag will be set to 1. When one of the flags, ADF, CMPF0, CMPF1 and CALIF, is set to 1 and the corresponding interrupt enable bit, ADIE of ADCR register, CMPIE of ADCMPR0/1 registers and CALIE of ADCALR register, is set to 1, the ADC interrupt will be asserted. Software can clear these flags to revoke the interrupt request.

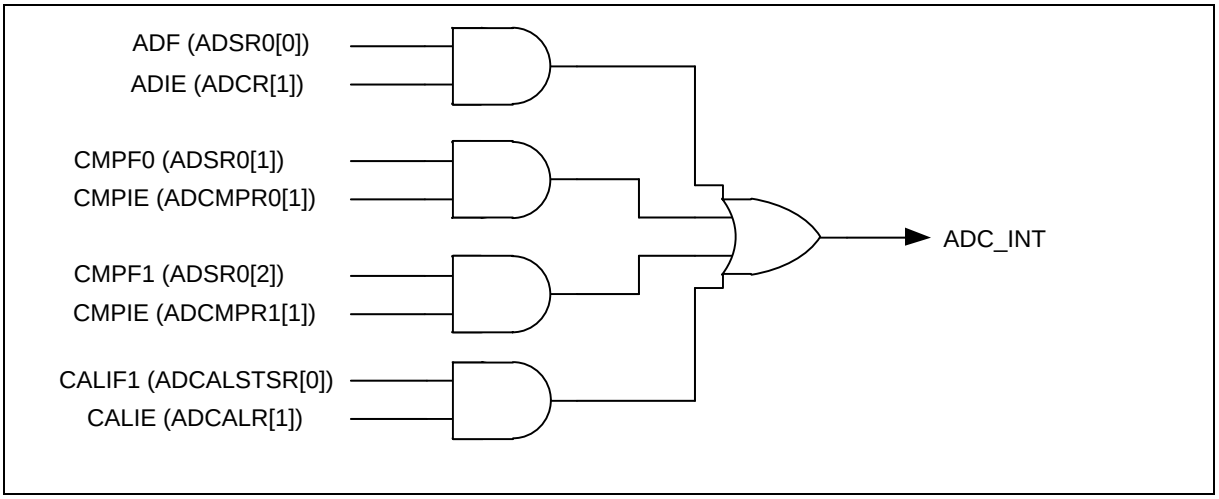


Figure 6.25-9 A/D Controller Interrupt

6.25.5.17 Floating Detect Function

The ADC internal channel can be used to detect pin disconnection situation. Setting FDETCHEN (ADC_CFDCTL[8]) to 1 can always turn on internal channel. Assume V_{REF} is 3.0V. For example:

1. Enable one external channel and set input voltage is 1.2V.
2. Set FDETCHEN and PRECHEN (ADC_CFDCTL[0]) to 1.
3. Wait for a time (by resistor value) and start conversion.
4. If the conversion result is close to V_{REF} result, this external channel disconnection is detected.

6.25.5.18 Differential Mode

The ADC 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 ADCs 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 ADCs prohibit the input common-mode voltage from varying more than approximately 10 percent beyond the mid-scale input (V_{REF} /2).

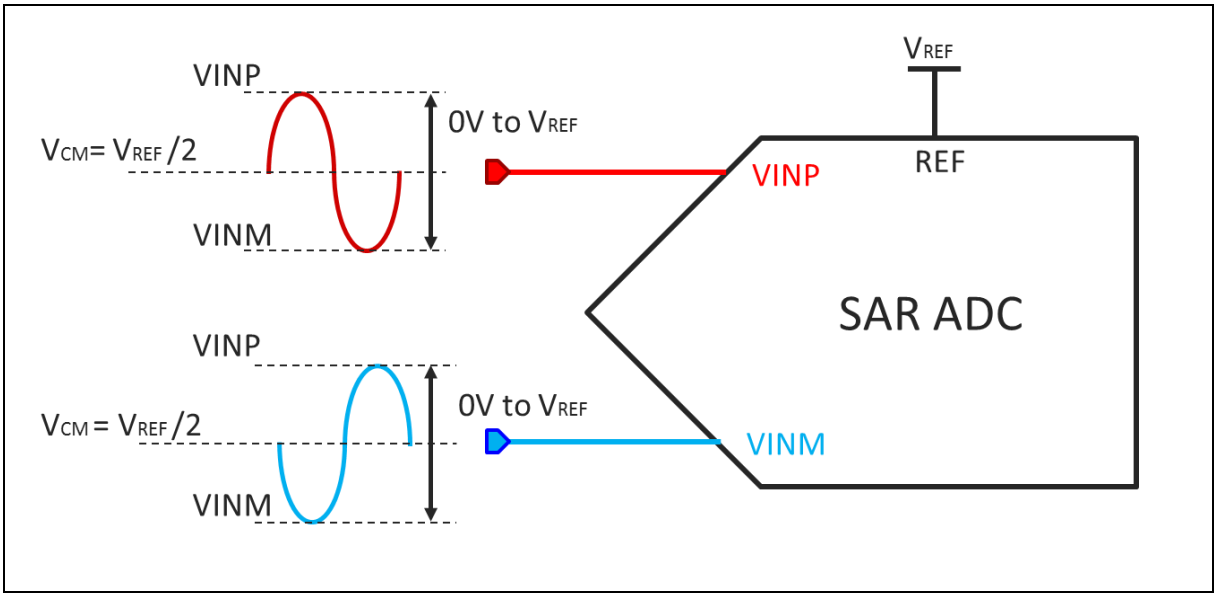


Figure 1.1-16 Input of Fully-Differential ADC

If user enables DIFFEN (ADC_ADCR[10]), the differential mode will enable. The pair of analog input channel is as Table 6.25-2.

Differential Analog Input Paired Channel	ADC Analog Input	
	V_{plus}	V_{minus}
0	ADC_CH0	ADC_CH1
1	ADC_CH2	ADC_CH3
2	ADC_CH4	ADC_CH5
3	ADC_CH6	ADC_CH7
4	ADC_CH8	ADC_CH9
5	ADC_CH10	ADC_CH11
6	ADC_CH12	ADC_CH13
7	ADC_CH14	ADC_CH15

Table 6.25-2 ADC Differential Model Channel Selection

In differential analog input mode, only the even number of the two corresponding channels needs to be enabled in CHEN (ADC_ADCHER[31:0]). The conversion result will be placed to the corresponding data register of the enabled channel. The conversion result will store with 2'complement format when DMOF (ADC_ADCR [31]) = 1.

6.25.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
ADC Base Address: **ADC_BA = 0x4004_3000**				
ADC_ADDR0	ADC_BA+0x00	R	ADC Data Register 0	0x0000_0000
ADC_ADDR1	ADC_BA+0x04	R	ADC Data Register 1	0x0000_0000
ADC_ADDR2	ADC_BA+0x08	R	ADC Data Register 2	0x0000_0000
ADC_ADDR3	ADC_BA+0x0C	R	ADC Data Register 3	0x0000_0000
ADC_ADDR4	ADC_BA+0x10	R	ADC Data Register 4	0x0000_0000
ADC_ADDR5	ADC_BA+0x14	R	ADC Data Register 5	0x0000_0000
ADC_ADDR6	ADC_BA+0x18	R	ADC Data Register 6	0x0000_0000
ADC_ADDR7	ADC_BA+0x1C	R	ADC Data Register 7	0x0000_0000
ADC_ADDR8	ADC_BA+0x20	R	ADC Data Register 8	0x0000_0000
ADC_ADDR9	ADC_BA+0x24	R	ADC Data Register 9	0x0000_0000
ADC_ADDR10	ADC_BA+0x28	R	ADC Data Register 10	0x0000_0000
ADC_ADDR11	ADC_BA+0x2C	R	ADC Data Register 11	0x0000_0000
ADC_ADDR12	ADC_BA+0x30	R	ADC Data Register 12	0x0000_0000
ADC_ADDR13	ADC_BA+0x34	R	ADC Data Register 13	0x0000_0000
ADC_ADDR14	ADC_BA+0x38	R	ADC Data Register 14	0x0000_0000
ADC_ADDR15	ADC_BA+0x3C	R	ADC Data Register 15	0x0000_0000
ADC_ADDR29	ADC_BA+0x74	R	ADC Data Register 29	0x0000_0000
ADC_ADCR	ADC_BA+0x80	R/W	ADC Control Register	0x0000_0000
ADC_ADCHER	ADC_BA+0x84	R/W	ADC Channel Enable Register	0x0000_0000
ADC_ADCMPR0	ADC_BA+0x88	R/W	ADC Compare Register 0	0x0000_0000
ADC_ADCMPR1	ADC_BA+0x8C	R/W	ADC Compare Register 1	0x0000_0000
ADC_ADSR0	ADC_BA+0x90	R/W	ADC Status Register0	0x0000_0000
ADC_ADSR1	ADC_BA+0x94	R	ADC Status Register1	0x0000_0000
ADC_ADSR2	ADC_BA+0x98	R	ADC Status Register2	0x0000_0000
ADC_ESMPCTL	ADC_BA+0xA0	R/W	ADC Extend Sample Time Control Register	0x0000_0000
ADC_CFDCTL	ADC_BA+0xA4	R/W	ADC Channel Floating Detect Control Register	0x0000_0000
ADC_ADPDMA	ADC_BA+0x100	R	ADC PDMA Current Transfer Data Register	0x0000_0000

ADC_ADCALR	ADC_BA+0x180	R/W	ADC Calibration Mode Register	0x0000_005C
ADC_ADCALSTSR	ADC_BA+0x184	R/W	ADC Calibration Status Register	0x0000_0000

6.25.7 Register Description

ADC Data Registers (ADC_ADDRx x = 0~15, 29)

Register	Offset	R/W	Description	Reset Value
ADC_ADDR0	ADC_BA+0x00	R	ADC Data Register 0	0x0000_0000
ADC_ADDR1	ADC_BA+0x04	R	ADC Data Register 1	0x0000_0000
ADC_ADDR2	ADC_BA+0x08	R	ADC Data Register 2	0x0000_0000
ADC_ADDR3	ADC_BA+0x0C	R	ADC Data Register 3	0x0000_0000
ADC_ADDR4	ADC_BA+0x10	R	ADC Data Register 4	0x0000_0000
ADC_ADDR5	ADC_BA+0x14	R	ADC Data Register 5	0x0000_0000
ADC_ADDR6	ADC_BA+0x18	R	ADC Data Register 6	0x0000_0000
ADC_ADDR7	ADC_BA+0x1C	R	ADC Data Register 7	0x0000_0000
ADC_ADDR8	ADC_BA+0x20	R	ADC Data Register 8	0x0000_0000
ADC_ADDR9	ADC_BA+0x24	R	ADC Data Register 9	0x0000_0000
ADC_ADDR10	ADC_BA+0x28	R	ADC Data Register 10	0x0000_0000
ADC_ADDR11	ADC_BA+0x2C	R	ADC Data Register 11	0x0000_0000
ADC_ADDR12	ADC_BA+0x30	R	ADC Data Register 12	0x0000_0000
ADC_ADDR13	ADC_BA+0x34	R	ADC Data Register 13	0x0000_0000
ADC_ADDR14	ADC_BA+0x38	R	ADC Data Register 14	0x0000_0000
ADC_ADDR15	ADC_BA+0x3C	R	ADC Data Register 15	0x0000_0000
ADC_ADDR29	ADC_BA+0x74	R	ADC Data Register 29	0x0000_0000

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

Bits	Description	
[31:18]	**Reserved**	Reserved.

[17]	**VALID**	**Valid Flag (Read Only)** This bit will be set to 1 when the conversion of the corresponding channel is completed. This bit will be cleared to 0 by hardware after ADDR register is read. 0 = Data in RSLT bits is not valid. 1 = Data in RSLT bits is valid.
[16]	**OVERRUN**	**Overrun Flag (Read Only)** If converted data in RSLT bits has not been read before new conversion result is loaded to this register, OVERRUN bit is set to 1. It is cleared by hardware after ADDR register is read. 0 = Data in RSLT bits is not overwritten. 1 = Data in RSLT bits is overwritten.
[15:0]	**RSLT**	**A/D Conversion Result (Read Only)** This field contains conversion result of ADC.

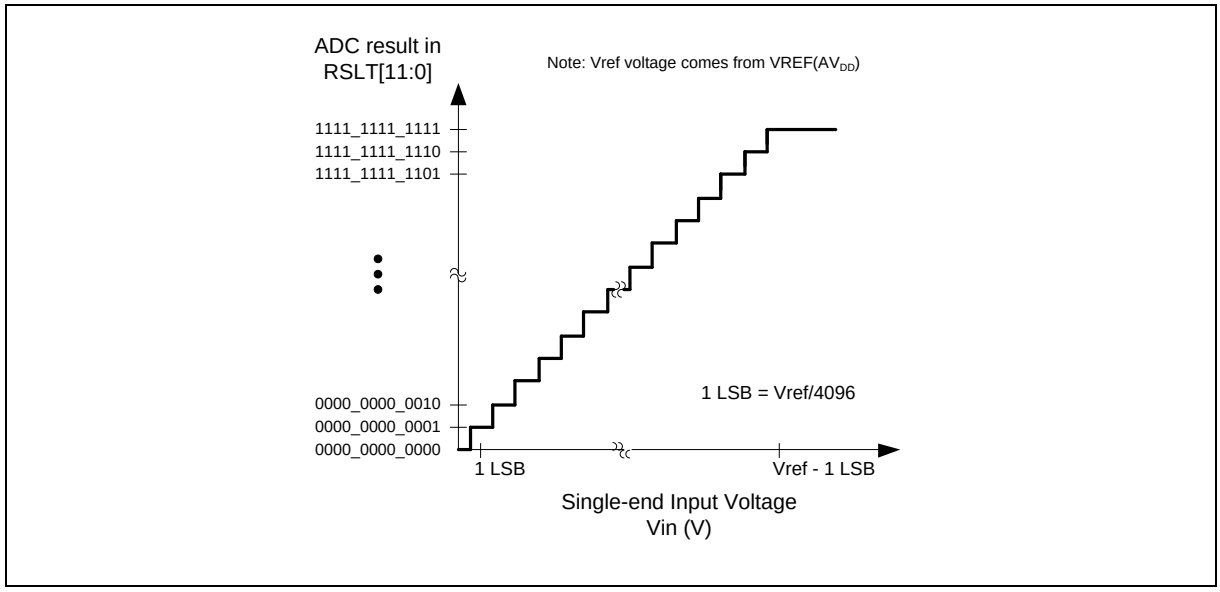


Figure 6.25-10 Conversion Result Mapping Diagram of ADC Single-end Input

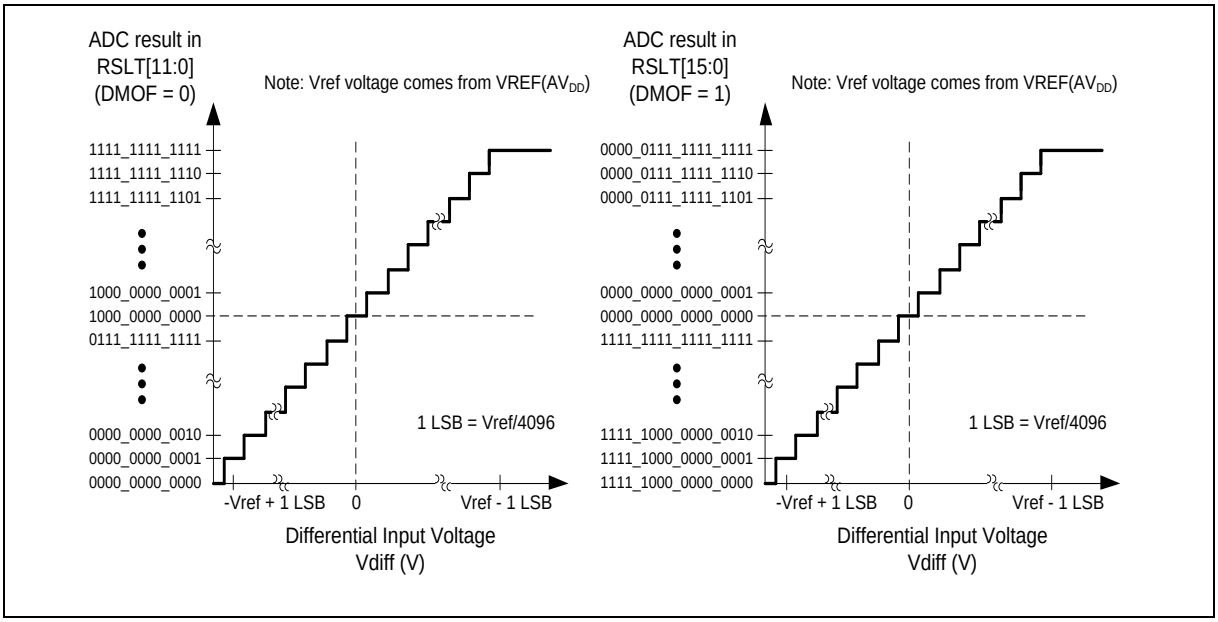


Figure 6.25-11 Conversion Result Mapping Diagram of ADC Differential Input

ADC Control Register (ADC_ADCR)

Register	Offset	R/W	Description	Reset Value
ADC_ADCR	ADC_BA+0x80	R/W	ADC Control Register	0x0000_0000

31	30	29	28	27	26	25	24
DMOF	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved			RESET	ADST	DIFFEN	PTEN	TRGEN
7	6	5	4	3	2	1	0
TRGCOND		TRGS		ADMD		ADIE	ADEN

Bits	Description	
[31]	**DMOF**	**Differential Input Mode Output Format** If user enables differential input mode, the conversion result can be expressed with binary straight format (unsigned format) or 2's complement format (signed format). 0 = A/D Conversion result will be filled in RSLT at ADDRx registers with unsigned format (straight binary format). 1 = A/D Conversion result will be filled in RSLT at ADDRx registers with 2's complement format.
[30:13]	**Reserved**	Reserved.
[12]	**RESET**	**ADC RESET (Write Protect)** If user writes this bit, the ADC analog macro will reset. Calibration data in macro will be deleted, but registers in ADC controller will keep. **Note:** This bit is cleared by hardware.
[11]	**ADST**	**A/D Conversion Start or Calibration Start** ADST bit can be set to 1 from four sources: software, external pin STADC, PWM trigger and Timer trigger. ADST bit will be cleared to 0 by hardware automatically at the ends of Single mode, Single-cycle Scan mode and Calibration mode. In Continuous Scan mode and Burst mode, A/D conversion is continuously performed until software writes 0 to this bit or chip is reset. 0 = Conversion stops and A/D converter enters idle state. 1 = Conversion starts or Calibration Start. **Note 1:** When ADST becomes from 1 to 0, ADC macro will reset to initial state. After macro reset to initial state, user should wait at most 2 ADC clock and set this bit to start next conversion. **Note 2:** Calibration Start only if CALEN (ADC_ADCALR[0]) = 1.
[10]	**DIFFEN**	**Differential Input Mode Control** Differential input voltage (V_{diff}) = V_{plus} - V_{minus}. The relation between V_{plus} and V_{minus} is V_{plus} + V_{minus} = V_{ref}. The V_{plus} of differential input paired channel x is from ADC0_CHy pin; V_{minus} is from ADC0_CHz pin, x=0,1..7, y=2*x, z=y+1. 0 = Single-end analog input mode. 1 = Differential analog input mode.

		Note: In Differential Input mode, only the even number of the two corresponding channels needs to be enabled in ADCHER register. The conversion result will be placed to the corresponding data register of the enabled channel.
[9]	**PTEN**	**PDMA Transfer Enable Bit** When A/D conversion is completed, the converted data is loaded into ADDR0~15, ADDR29. Software can enable this bit to generate a PDMA data transfer request. 0 = PDMA data transfer Disabled. 1 = PDMA data transfer in ADDR0~15, ADDR29 Enabled. **Note:** When PTEN=1, software must set ADIE=0 to disable interrupt.
[8]	**TRGEN**	**External Trigger Enable Bit** Enable or disable triggering of A/D conversion by external STADC pin, PWM trigger, BPWM trigger and Timer trigger. If external trigger is enabled, the ADST bit can be set to 1 by the selected hardware trigger source. 0 = External trigger Disabled. 1 = External trigger Enabled. **Note:** The ADC external trigger function is only supported in Single-cycle Scan mode.
[7:6]	**TRGCOND**	**External Trigger Condition** These two bits decide external pin STADC trigger event is level or edge. The signal must be kept at stable state at least 8 PCLKs for level trigger and at least 4 PCLKs for edge trigger. 00 = Low level. 01 = High level. 10 = Falling edge. 11 = Rising edge.
[5:4]	**TRGS**	**Hardware Trigger Source** 00 = A/D conversion is started by external STADC pin. 01 = Timer0 ~ Timer3 overflow pulse trigger. 10 = A/D conversion is started by BPWM trigger. 11 = A/D conversion is started by PWM trigger. **Note:** Software should clear TRGEN bit and ADST bit to 0 before changing TRGS bits.
[3:2]	**ADMD**	**A/D Converter Operation Mode Control** 00 = Single conversion. 01 = Burst conversion. 10 = Single-cycle Scan. 11 = Continuous Scan. **Note 1:** When changing the operation mode, software should clear ADST bit first. **Note 2:** In Burst mode, the A/D result data is always at ADC Data Register 0.
[1]	**ADIE**	**A/D Interrupt Enable Bit** A/D conversion end interrupt request is generated if ADIE bit is set to 1. 0 = A/D interrupt function Disabled. 1 = A/D interrupt function Enabled.
[0]	**ADEN**	**A/D Converter Enable Bit** 0 = A/D converter Disabled. 1 = A/D converter Enabled. **Note:** Before starting A/D conversion function, this bit should be set to 1. Clear it to 0 to disable A/D converter analog circuit to save power consumption.

ADC Channel Enable Register (ADC_ADCHER)

Register	Offset	R/W	Description	Reset Value
ADC_ADCHER	ADC_BA+0x84	R/W	ADC Channel Enable Register	0x0000_0000

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

Bits	Description	
[31:0]	**CHEN**	**Analog Input Channel Enable Control** Set ADCHER[15:0] bits to enable the corresponding analog input channel 15 ~ 0. If DIFFEN bit is set to 1, only the even number channel needs to be enabled. Besides, setting the ADCHER[29] bit will enable internal channel for band-gap voltage. Other bits are reserved. 0 = Channel Disabled. 1 = Channel Enabled. **Note:** If the internal channel for band-gap voltage (CHEN[29]) is active, the maximum sampling rate will be 300k SPS.

ADC Compare Register 0/1 (ADC_ADCMPR0/1)

Register	Offset	R/W	Description	Reset Value
ADC_ADCMPR0	ADC_BA+0x88	R/W	ADC Compare Register 0	0x0000_0000
ADC_ADCMPR1	ADC_BA+0x8C	R/W	ADC Compare Register 1	0x0000_0000

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

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:16]	**CMPD**	**Comparison Data** The 12-bit data is used to compare with conversion result of specified channel. **Note:** CMPD bits should be filled in unsigned format (straight binary format).
[15]	**CMPWEN**	**Compare Window Mode Enable Bit** 0 = Compare Window Mode Disabled. 1 = Compare Window Mode Enabled. **Note:** This bit is only presented in ADCMPR0 register.
[14:12]	**Reserved**	Reserved.
[11:8]	**CMPMATCNT**	**Compare Match Count** When the specified A/D channel analog conversion result matches the compare condition defined by CMPCOND bit, the internal match counter will increase 1. When the internal counter reaches the value to (CMPMATCNT +1), the CMPFx bit will be set.
[7:3]	**CMPCH**	**Compare Channel Selection** 00000 = Channel 0 conversion result is selected to be compared. 00001 = Channel 1 conversion result is selected to be compared. 00010 = Channel 2 conversion result is selected to be compared. 00011 = Channel 3 conversion result is selected to be compared. 00100 = Channel 4 conversion result is selected to be compared. 00101 = Channel 5 conversion result is selected to be compared. 00110 = Channel 6 conversion result is selected to be compared. 00111 = Channel 7 conversion result is selected to be compared. 01000 = Channel 8 conversion result is selected to be compared. 01001 = Channel 9 conversion result is selected to be compared. 01010 = Channel 10 conversion result is selected to be compared.

		01011 = Channel 11 conversion result is selected to be compared. 01100 = Channel 12 conversion result is selected to be compared. 01101 = Channel 13 conversion result is selected to be compared. 01110 = Channel 14 conversion result is selected to be compared. 01111 = Channel 15 conversion result is selected to be compared. 11100 = Floating detect channel conversion result is selected to be compared. 11101 = Band-gap voltage conversion result is selected to be compared. Others = Reserved.
[2]	**CMPCOND**	**Compare Condition** 0 = Set the compare condition as that when a 12-bit A/D conversion result is less than the 12-bit CMPD bits, the internal match counter will increase one. 1 = Set the compare condition as that when a 12-bit A/D conversion result is greater than or equal to the 12-bit CMPD bits, the internal match counter will increase one. **Note:** When the internal counter reaches to (CMPMATCNT +1), the CMPFx bit will be set.
[1]	**CMPIE**	**Compare Interrupt Enable Bit** If the compare function is enabled and the compare condition matches the setting of CMPCOND and CMPMATCNT, CMPFx bit will be asserted, in the meanwhile, if CMPIE bit is set to 1, a compare interrupt request is generated. 0 = Compare function interrupt Disabled. 1 = Compare function interrupt Enabled.
[0]	**CMPEN**	**Compare Enable Bit** Set this bit to 1 to enable ADC controller to compare CMPD (ADCMPRx[27:16]) with specified channel conversion result when converted data is loaded into ADDR register. 0 = Compare function Disabled. 1 = Compare function Enabled.

ADC Status Register0 (ADC_ADSR0)

Register	Offset	R/W	Description	Reset Value
ADC_ADSR0	ADC_BA+0x90	R/W	ADC Status Register0	0x0000_0000

31	30	29	28	27	26	25	24
CHANNEL					**Reserved**		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							**OVERRUNF**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**VALIDF**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BUSY	**Reserved**				**CMPF1**	**CMPF0**	**ADF**

Bits	Description	
[31:27]	**CHANNEL**	**Current Conversion Channel (Read Only)** When BUSY=1, this filed reflects current conversion channel. When BUSY=0, it shows the number of the next converted channel.
[26:17]	**Reserved**	Reserved.
[16]	**OVERRUNF**	**Overrun Flag (Read Only)** If any one of OVERRUN (ADDRx[16]) is set, this flag will be set to 1. **Note:** When ADC is in burst mode and the FIFO is overrun, this flag will be set to 1.
[15:9]	**Reserved**	Reserved.
[8]	**VALIDF**	**Data Valid Flag (Read Only)** If any one of VALID (ADDRx[17]) is set, this flag will be set to 1. **Note:** When ADC is in burst mode and any conversion result is valid, this flag will be set to 1.
[7]	**BUSY**	**BUSY/IDLE (Read Only)** This bit is a mirror of ADST bit in ADCR register. 0 = A/D converter is in idle state. 1 = A/D converter is busy at conversion.
[6:3]	**Reserved**	Reserved.
[2]	**CMPF1**	**Compare Flag 1** When the A/D conversion result of the selected channel meets setting condition in ADCMPR1 register, this bit is set to 1; it is cleared by writing 1 to it 0 = Conversion result in ADDR does not meet ADCMPR1 setting. 1 = Conversion result in ADDR meets ADCMPR1 setting.
[1]	**CMPF0**	**Compare Flag 0** When the A/D conversion result of the selected channel meets setting condition in ADCMPR0 register then this bit is set to 1. This bit is cleared by writing 1 to it. 0 = Conversion result in ADDR does not meet ADCMPR0 setting. 1 = Conversion result in ADDR meets ADCMPR0 setting.

[0]	**ADF**	**A/D Conversion End Flag** A status flag that indicates the end of A/D conversion. Software can write 1 to clear this bit. The ADF bit is set to 1 at the following three conditions: 1. When A/D conversion ends in Single mode. 2. When A/D conversion ends on all specified channels in Single-cycle Scan mode and Continuous Scan mode. 3. When more than or equal to 4 samples in FIFO in Burst mode.

ADC Status Register1 (ADC_ADSR1)

Register	Offset	R/W	Description	Reset Value
ADC_ADSR1	ADC_BA+0x94	R	ADC Status Register1	0x0000_0000

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

Bits	Description	
[31:0]	**VALID**	**Data Valid Flag (Read Only)** VALID[29, 15:0] are the mirror of the VALID bits in ADDR29[17], ADDR15[17]~ ADDR0[17]. The other bits are reserved. **Note:** When ADC is in burst mode and any conversion result is valid, VALID[29, 15:0] will be set to 1.

ADC Status Register2 (ADC_ADSR2)

Register	Offset	R/W	Description	Reset Value
ADC_ADSR2	ADC_BA+0x98	R	ADC Status Register2	0x0000_0000

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

Bits	Description	
[31:0]	**OVERRUN**	**Overrun Flag (Read Only)** OVERRUN[29, 15:0] are the mirror of the OVERRUN bit in ADDR29[16], ADDR15[16] ~ ADDR0[16]. The other bits are reserved. **Note:** When ADC is in burst mode and the FIFO is overrun, OVERRUN[29, 15:0] will be set to 1.

ADC Extend Sample Time Control Register (ADC_ESMPCTL)

Register	Offset	R/W	Description	Reset Value
ADC_ESMPCTL	ADC_BA+0xA0	R/W	ADC Extend Sample Time Control Register	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**EXTSMPT**	**ADC Sampling Time Extend** When ADC converting at high conversion rate, the sampling time of analog input voltage may not enough if input channel loading is heavy, user can extend ADC sampling time after trigger source is coming to get enough sampling time. The range of start delay time is from 0~255 ADC clock.

ADC Channel Floating Detect Control Register (ADC_CFDCTL)

Register	Offset	R/W	Description	Reset Value
ADC_CFDCTL	ADC_BA+0xA4	R/W	ADC Channel Floating Detect Control Register	0x0000_0000

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

Bits	Description	
[31:9]	**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 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 enable.
[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 enable.

ADC PDMA Current Transfer Data Register (ADC_ADPDMA)

Register	Offset	R/W	Description	Reset Value
ADC_ADPDMA	ADC_BA+0x100	R	ADC PDMA Current Transfer Data Register	0x0000_0000

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

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17:0]	**CURDAT**	**ADC PDMA Current Transfer Data Register (Read Only)** When PDMA transferring, read this register can monitor current PDMA transfer data. Current PDMA transfer data could be the content of ADDR0 ~ ADDR15, and ADDR29 registers.

ADC Calibration Mode Register (ADC_ADCALR)

Register	Offset	R/W	Description	Reset Value
ADC_ADCALR	ADC_BA+0x180	R/W	ADC Calibration Mode Register	0x0000_005C

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

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**CALIE**	**Calibration Interrupt Enable Bit** If calibration function is enabled and the calibration finish, CALIF bit will be asserted, in the meanwhile, if CALIE bit is set to 1, a calibration interrupt request is generated. 0 = Calibration function Interrupt Disabled. 1 = Calibration function Interrupt Enabled.
[0]	**CALEN**	**Calibration Function Enable Bit** 0 = Calibration function Disabled. 1.= Calibration function Enabled. **Note:** If chip is powered off, calibration function should be executed again.

ADC Calibration Status Register (ADC_ADCALSTSR)

Register	Offset	R/W	Description	Reset Value
ADC_ADCALSTSR	ADC_BA+0x184	R/W	ADC 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							**CALIF**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**CALIF**	**Calibration Finish Interrupt Flag** If calibration is finished, this flag will be set to 1. It is cleared by writing 1 to it.

6.26 Analog Comparator Controller (ACMP)

6.26.1 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.

6.26.2 Features

- Analog input voltage range: 0 ~ AV_{DD} (voltage of AV_{DD} pin)
- Supports hysteresis function
- 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
  - 3 negative sources:
    - ACMP0_N
    - Comparator Reference Voltage (CRV)
    - Internal band-gap voltage (VBG)
- ACMP1 supports
  - 4 multiplexed I/O pins at positive sources:
    - ACMP1_P0, ACMP1_P1, ACMP1_P2, or ACMP1_P3
  - 3 negative sources:
    - ACMP1_N
    - Comparator Reference Voltage (CRV)
    - Internal band-gap voltage (VBG)
- 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 PWM
- Supports window compare mode and window latch mode
- Supports calibration function

Section	Sub-Section	M031xB/C/D/E M032xB/C/D/E	M031xG/I M032xG/I
Function Description	6.26.5.7 Calibration function	-/-/-/●	●

Table 6.26-1 Calibration Function Features Comparison Table at Different Chip

6.26.3 Block Diagram

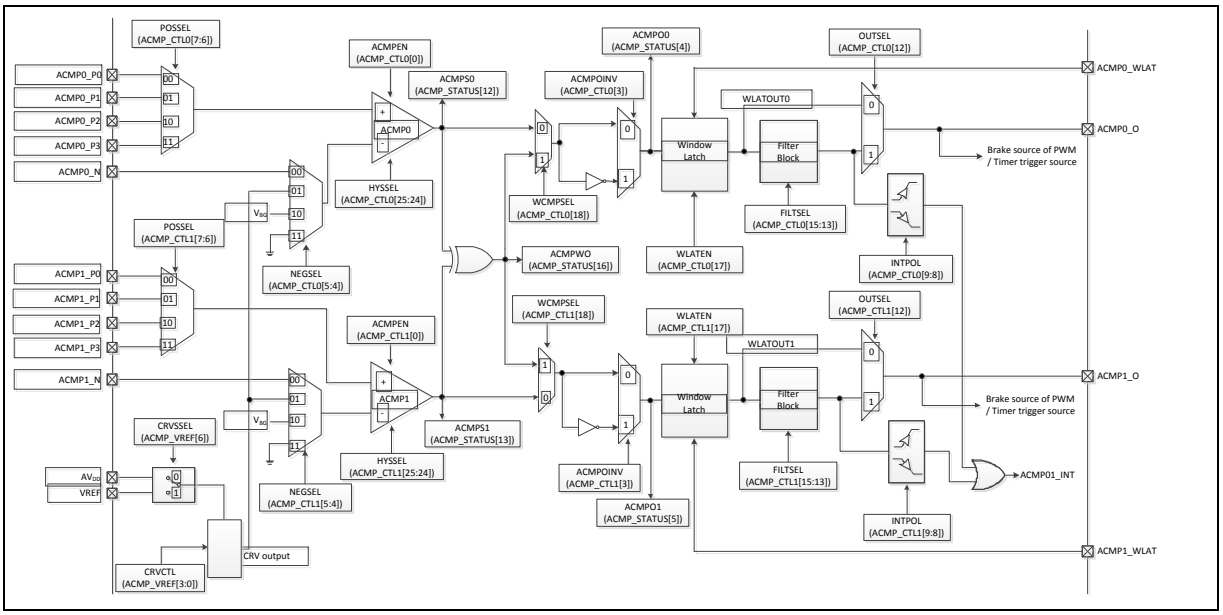


Figure 6.26-1 Analog Comparator Block Diagram

6.26.4 Basic Configuration

6.26.4.1 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]).

6.26.4.2 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]).

6.26.5 Functional Description

6.26.5.1 Hysteresis Function

The analog comparator provides the hysteresis function to make the comparator to have a stable output transition (referring to Figure 6.26-2). 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.

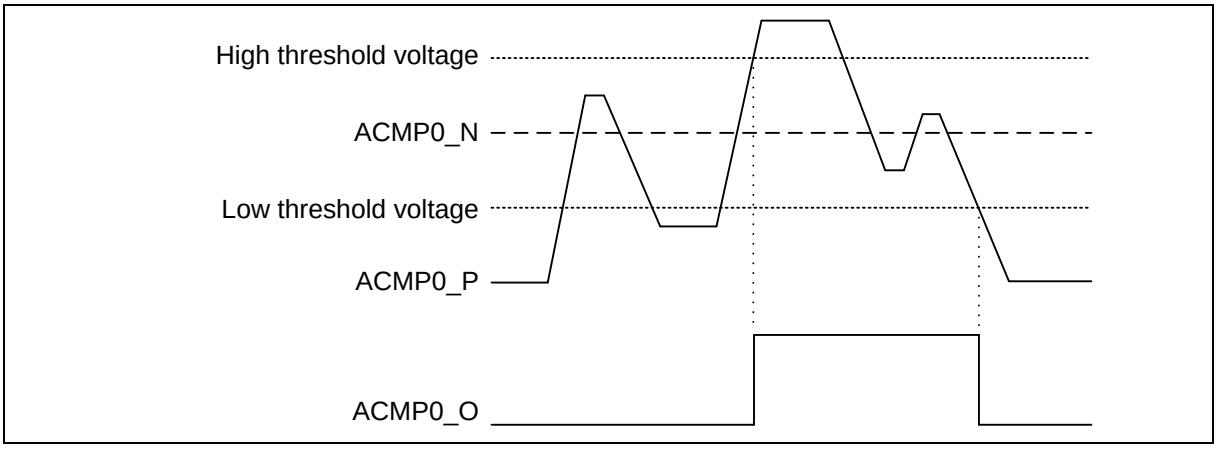


Figure 6.26-2 Comparator Hysteresis Function of ACMP0

6.26.5.2 Window Latch Mode

Figure 6.26-3 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.

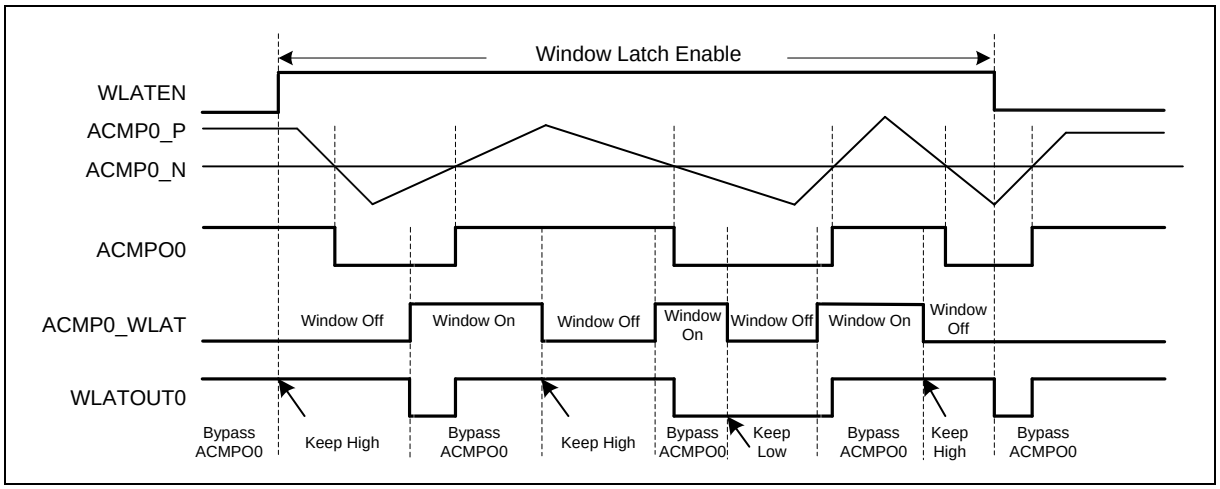


Figure 6.26-3 Window Latch Mode

6.26.5.3 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.

Figure 6.26-4 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 filted.

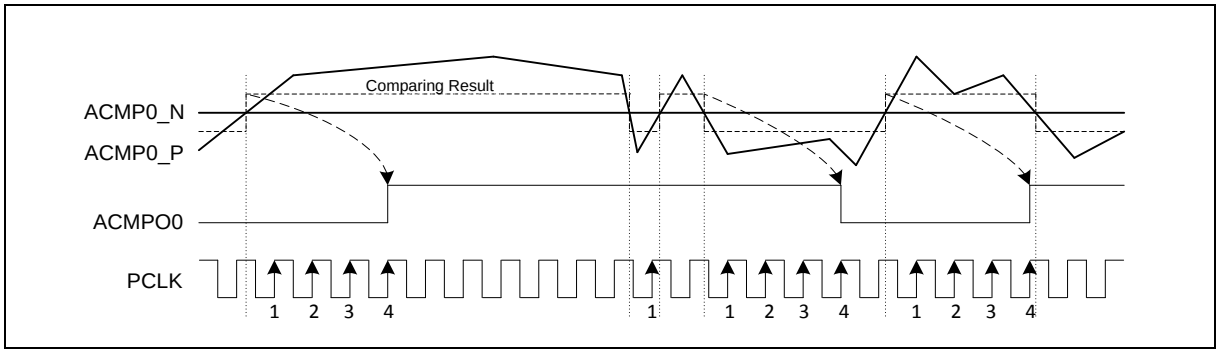


Figure 6.26-4 An example of filter function

6.26.5.4 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 6.26-5, 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.

If ACMP wakeup function is enabled and system wakeup from power down by ACMP with interrupt enabled (ACMPIE), the WKIF (ACMP_STATUS[8], ACMP_STATUS[9]) will be set that causes interrupt rising.

Figure 6.26-5 shows the interrups of ACMP is coming from ACMPIF or WKIF and enabled or disabled by ACMPIE.

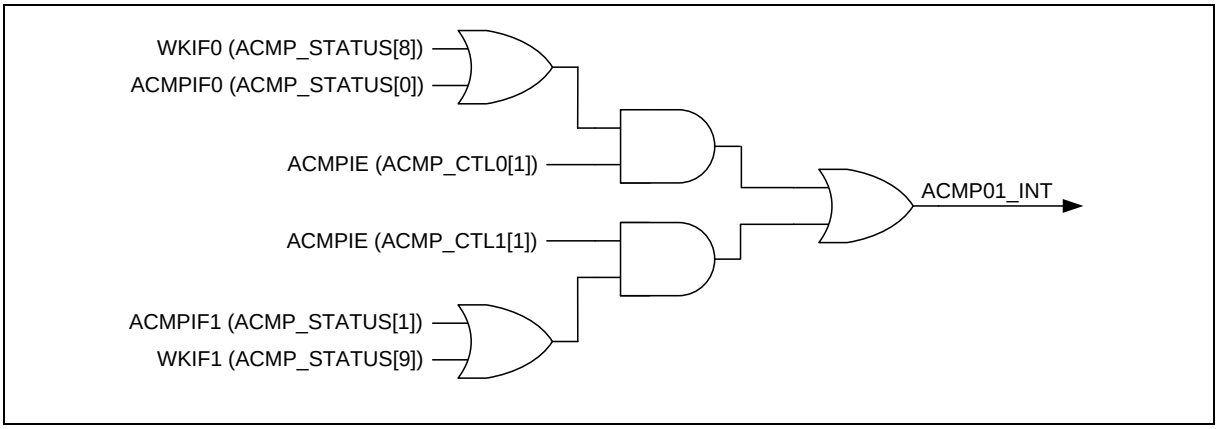


Figure 6.26-5 Comparator Controller Interrupt

6.26.5.5 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 CRVCTL (ACMP_VREF[3:0]). 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 6.26-6 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 V_{REF} pin.

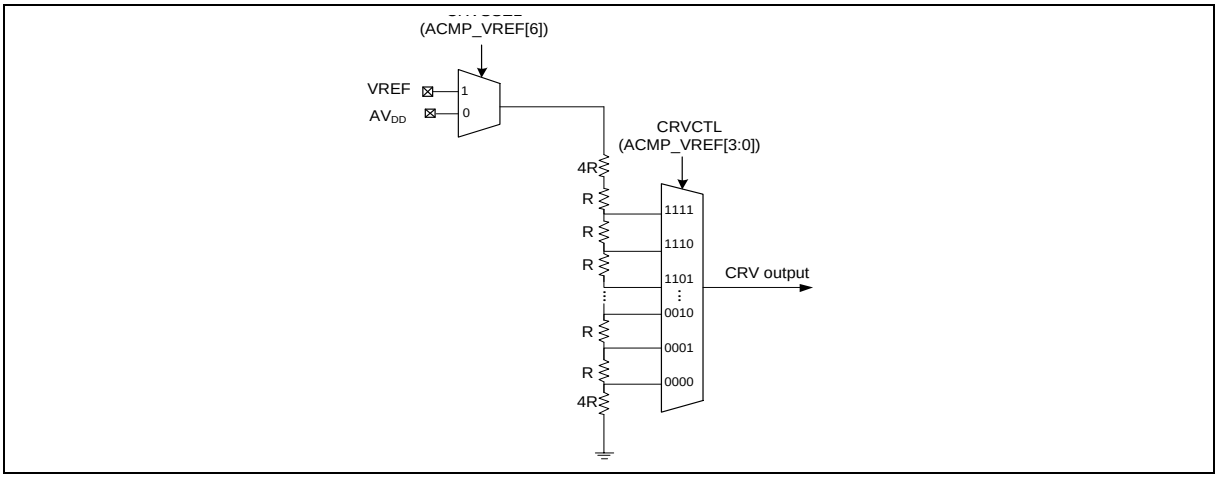


Figure 6.26-6 Comparator Reference Voltage Block Diagram

6.26.5.6 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 6.26-7 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 is in the window. Otherwise, the voltage source is outside the window.

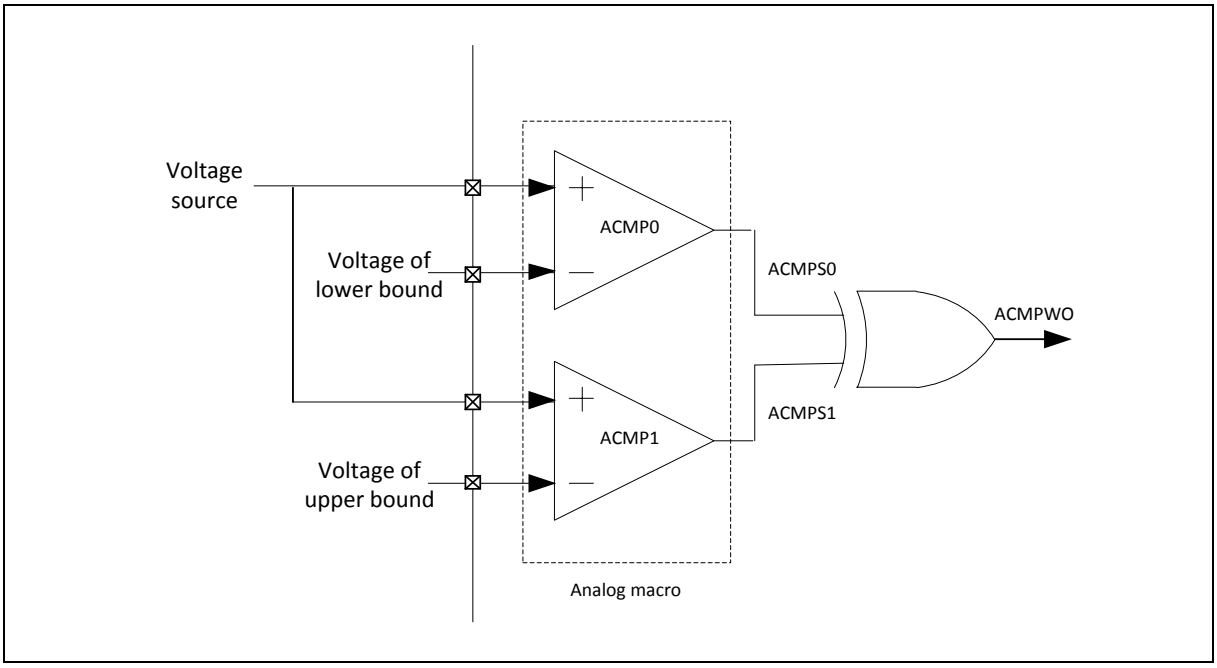


Figure 6.26-7 Example of Window Compare Mode

The comparator window output (ACMPWO) can be shown in ACMP_STATUS[16] and the truth table of window compare logic are shown in Table 6.26-2.

ACMPS0	ACMPS1	ACMPWO
0	0	0
0	1	1
1	1	0
1	0	1

Table 6.26-2 Truth Table of Window Compare Logic

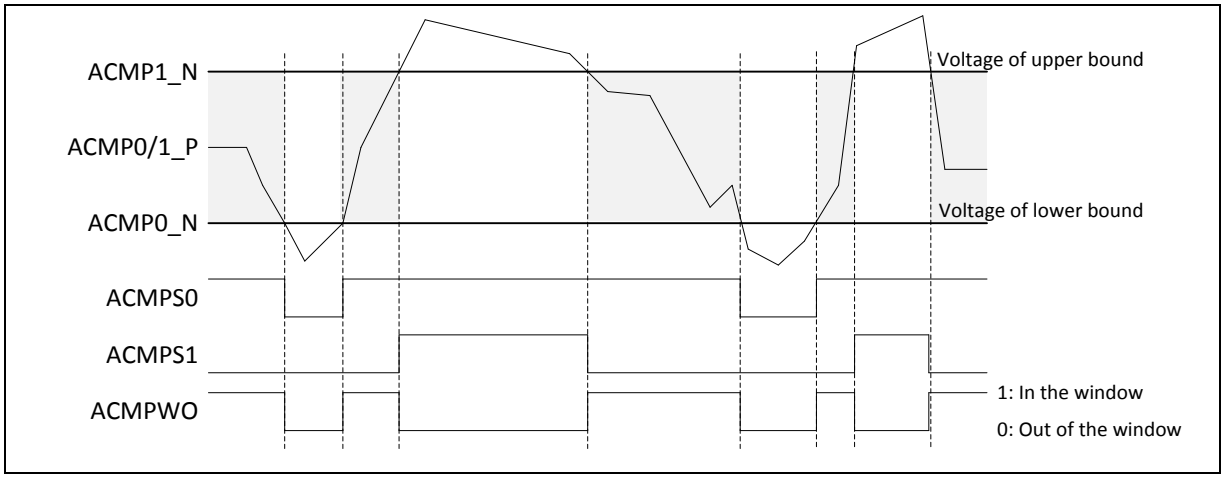


Figure 6.26-8 Example of Window Compare Mode

As shown in Figure 6.26-8, 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.

6.26.5.7 Calibration function

The two comparators have its own three trim bits which can be used to calibrate the offset voltage. 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 rail-to-rail 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]) to start calibration function getting newer offset trim values, or comparator will operating with elder offset trim values.

6.26.6 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_00f0
ACMP_CALSR	ACMP01_BA+0x14	R	Analog Comparator Calibration Status Register	0x0000_0000

6.26.7 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							
23	22	21	20	19	18	17	16
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	
[31:19]	**Reserved**	Reserved.
[18]	**WCMPSEL**	**Window Compare Mode Selection** 0 = Window Compare Mode Disabled. 1 = Window Compare Mode 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 (CRV). 10 = Band-gap voltage. 11 = Reserved.
[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							
23	22	21	20	19	18	17	16
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	
[31:19]	**Reserved**	Reserved.
[18]	**WCMPSEL**	**Window Compare Mode Selection** 0 = Window Compare Mode Disabled. 1 = Window Compare Mode 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.
[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.

Bits	Description	
[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 (CRV). 10 = Band-gap voltage. 11 = Reserved.
[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

Bits	Description	
		is disabled, i.e. 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							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**CRVSSEL**	**Reserved**		**CRVCTL**			

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**CRVSSEL**	**CRV Source Voltage Selection** 0 = AV_{DD} is selected as CRV source voltage. 1 = V_{REF} is selected as as CRV source voltage.
[5:4]	**Reserved**	Reserved.
[3:0]	**CRVCTL**	**Comparator Reference Voltage Setting** CRV = CRV source voltage * (1/6+CRVCTL/24).

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_00f0

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						CALRVS1	CALRVS0
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						CALTRG1	CALTRG0

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**CALRVS1**	**OPA1 Calibration Reference Voltage Selection** 0 = V_{REF} is $\frac{1}{2}$ AV_{DD}. 1 = V_{REF} from high vcm to low vcm. **Note:** CALRVS0 and CALRVS1 must be the same setting in calibration
[16]	**CALRVS0**	**OPA0 Calibration Reference Voltage Selection** 0 = V_{REF} is $\frac{1}{2}$ AV_{DD}. 1 = V_{REF} from high vcm to low vcm. **Note:** CALRVS0 and CALRVS1 must be the same setting in calibration
[15:2]	**Reserved**	Reserved.
[1]	**CALTRG1**	**OP Amplifier 1 Calibration Trigger Bit** 0 = Calibration is stopped. 1 = Calibration is triggered. **Note 1:** Before this bit is enabled, ACMPEN(ACMP_CTL1) 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 next calibration is triggered by software. **Note 3:** If user must trigger calibration twice or more times, the second trigger have to wait at least 300us after the previous calibration done
[0]	**CALTRG0**	**OP Amplifier 0 Calibration Trigger Bit** 0 = Calibration is stopped. 1 = Calibration is triggered. **Note 1:** Before this bit is enabled,ACMPEN(ACMP_CTL0) 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 next calibration is triggered by software **Note 3:** If user must trigger calibration twice or more times, the second trigger have to wait at least 300us after the previous calibration 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.

6.27 Peripherals Interconnection

6.27.1 Overview

Some peripherals have interconnections which allow autonomous communication or synchronous action between peripherals without needing to involve the CPU. Peripherals interact without CPU saves CPU resources, reduces power consumption, operates with no software latency and fast responds.

6.27.2 Peripherals Interconnect Matrix table

Source	Destination					
	ADC	HIRC TRIM	BPWM	PWM	Timer	UART/USCI
ACMP	-	-	-	3	6	-
BOD	-	-	-	3	-	-
Clock Fail	-	-	-	3	-	-
CPU Lockup	-	-	-	3	-	-
LIRC	-	-	-	-	6	-
HXT	-	-	-	-	-	
LXT	-	2		-	-	
BPWM	1		4	-	-	-
PWM	1	-	4	4	-	8
Timer	1	-	5	5	7	-
USBD	-	2	-	-	-	-

Table 6.27-1 Peripherals Interconnect Matrix table

6.27.3 Functional Description

6.27.3.1 From BPWM, PWM, TIMER to ADC

BPWM Trigger ADC Conversion

BPWM0/1 can be one of the ADC conversion trigger source.

Setting the ADC external hardware trigger input source from BPWM trigger described in section 6.25.5.12.

The detailed BPWM trigger conditions are described in section 6.11.5.16.

PWM Trigger ADC Conversion

PWM0/1 can be one of the ADC conversion trigger source.

Setting the ADC external hardware trigger input source from PWM trigger described in section 6.25.5.11.

The detailed PWM trigger conditions are described in section 6.12.5.23.

Timer Trigger ADC Conversion

Timer0 ~ Timer3 can be one of the ADC conversion trigger source. When timer counter value matches the timer compared value or when the TMx_EXT pin edge transition meets setting, timer will trigger the ADC to start the conversion.

Setting the ADC external hardware trigger input source from timer trigger described in section 6.25.5.10.

The detailed Timer trigger conditions are described in section 6.7.5.10.

6.27.3.2 From LXT and USBD to HIRC TRIM

Use LXT or USB Synchronous Mode to system auto-trim HIRC circuit

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

The detail of HIRC trim setting is described in section 6.3.9.

6.27.3.3 From ACMP, BOD, Clock Fail and CPU Lockup to PWM

ACMP_O Output and System Fail Signal as PWM Brake Source

PWM brake source can be ACMP0/1_O output signal or several different system fail conditions include clock fail, Brown-out detect, and Core lockup. When system fault, PWM brake signal generated, PWM output will be set to protect the power switch controlled by PWM.

The detailed setting of PWM brake function is described in section 6.12.5.19.

6.27.3.4 From BPWM/PWM to BPWM/PWM

BPWM Synchronous Start Function

Select synchronous source from BPWM0, BPWM1, PWM0 or PWM1, and select BPWM channels. The chosen BPWM channels will start counting at the same time once the synchronous start function is enabled and CNTSEN(BPWM_SSTRG[0]) is set.

The detailed setting of BPWM synchronous start function is described in section 6.11.5.11.

PWM Synchronous Start Function

Select synchronous source from PWM0 or PWM1, and select PWM channels. The chosen PWM channels will start counting at the same time once the synchronous start function is enabled and CNTSEN(PWM_SSTRG[0]) is set.

The detailed setting of PWM synchronous start function is described in section 6.12.5.21.

6.27.3.5 From TIMER to BPWM/PWM

Timer Generates Trigger Pulses as BPWM External Clock Source

Timer0 ~ Timer3 can generates trigger pules as BPWM0/BPWM1 external clock source.

When timer counter value matches the timer compared value or when the TMx_EXT pin edge transition meets setting, timer can generate a trigger pulse by setting described in section 6.7.5.10.

The setting of BPWM clock source is described in section 6.11.3.

Timer Generates Trigger Pulses as PWM External Clock Source

Timer0 ~ Timer3 can generates trigger pules as PWM0/PWM1 external clock source.

When timer counter value matches the timer compared value or when the TMx_EXT pin edge transition meets setting, timer can generate a trigger pulse by setting described in section 6.7.5.10.

The setting of PWM clock source is described in section 6.12.3

6.27.3.6 From ACMP and LIRC to Timer Capture Function

Measure the Time Interval of ACMP0/1 Output Signal or LIRC clock Speed

Sets the timer capture source from ACMP0/1 output signal or LIRC clock and measures the time interval of the signal by using timer capture function. Users can use the results of time interval to trim LIRC through software or to get the ACMP0/1 output pulse width.

The detailed setting of time capture function is described in section 6.7.5.8 and 6.7.5.9.

6.27.3.7 From Timer0/2 to Timer1/3

Inter-Timer Trigger Capture Mode

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. The Timer1/3 will be forced in capture mode and start/stop trigger-counting by Timer0/2 counter status.

The detail of inter-timer trigger capture mode is described in section 6.7.5.11.

6.27.3.8 From PWM to UART0/USCI0

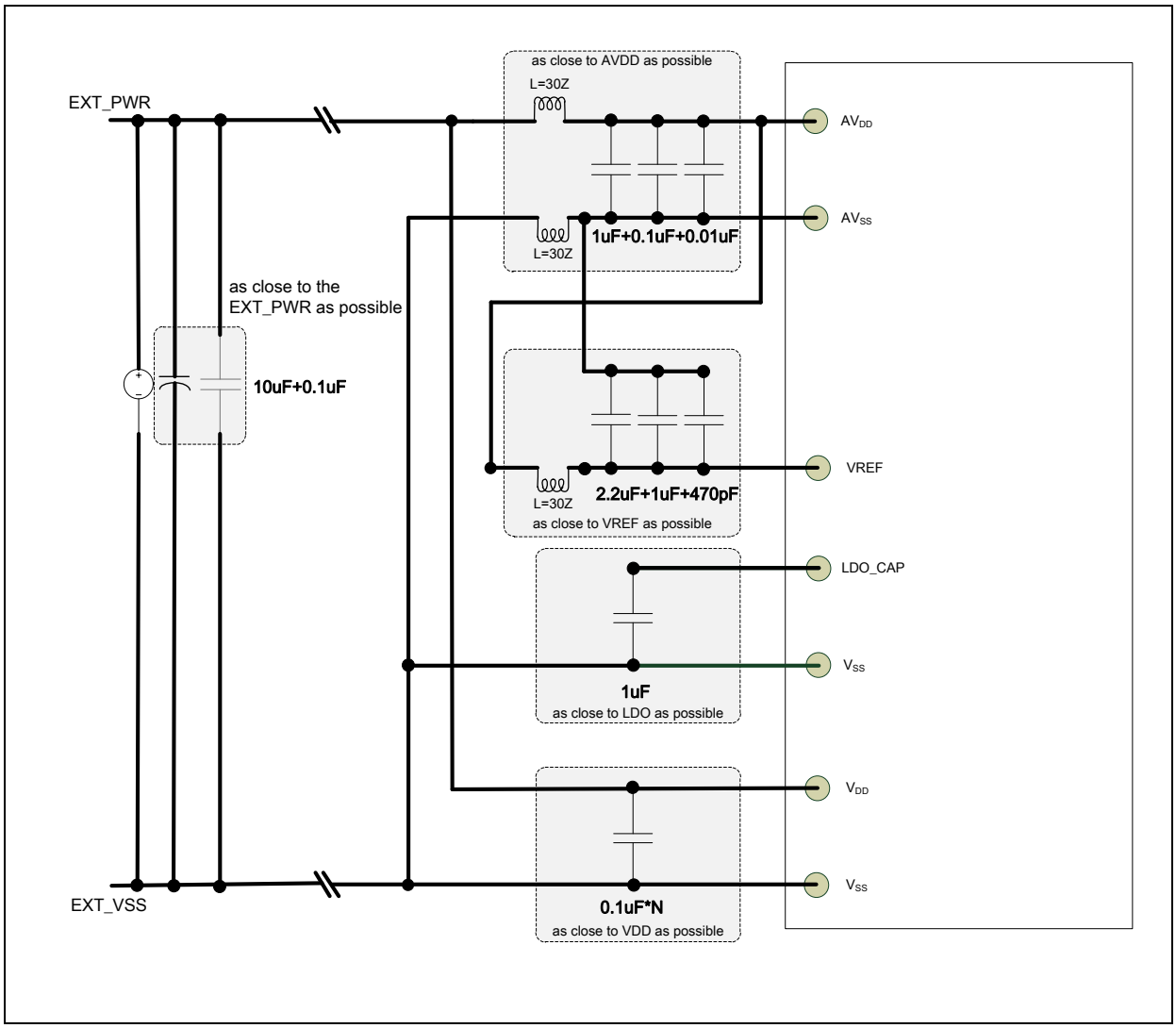
UART0_TXD/USCI0_DAT1 modulation with PWM

This chip supports UART0_TXD/USCI0_DAT1 to modulate with PWM channel. By modulate UART signal with PWM output signal, it is easy to implement IrDA protocol signal.

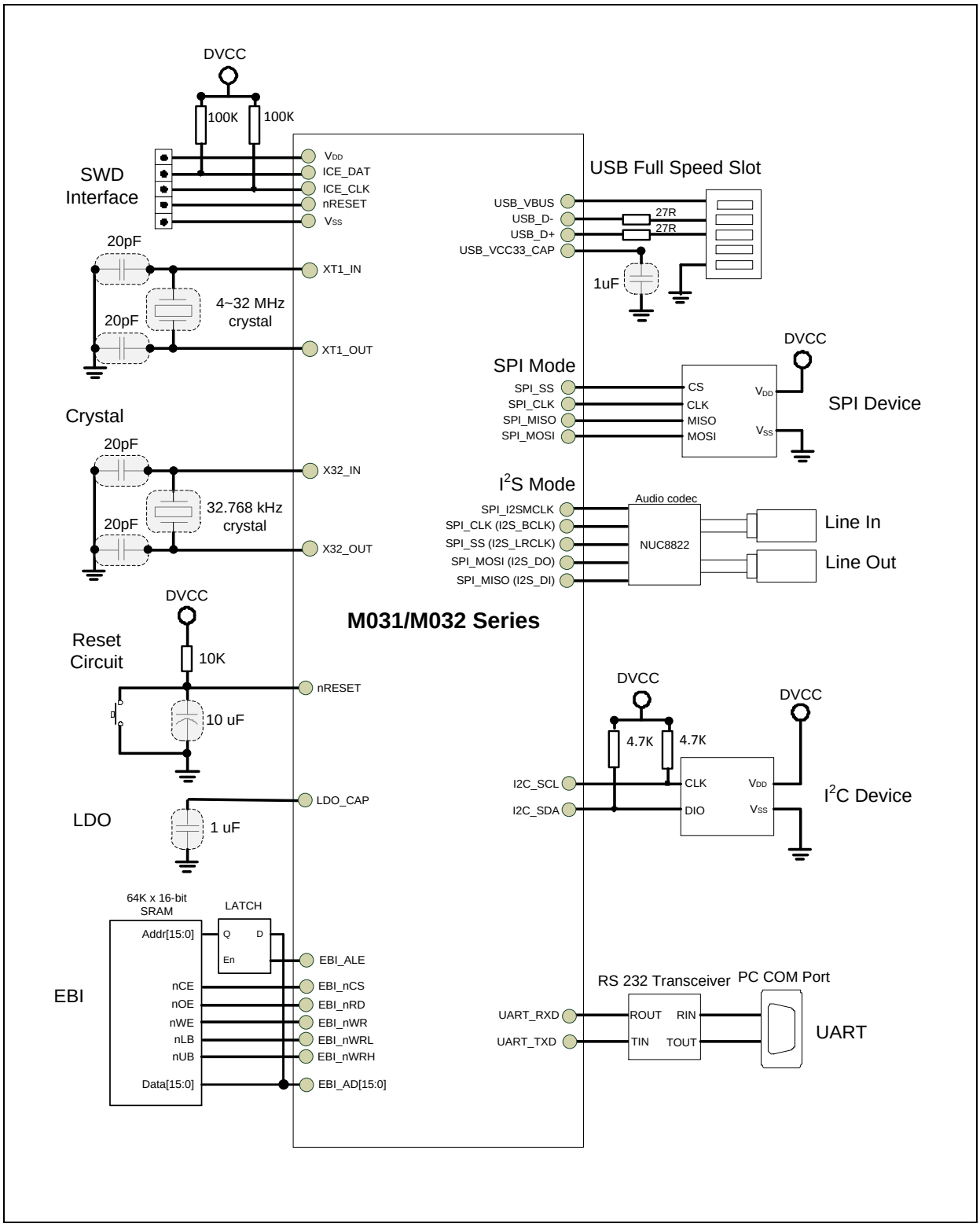
The detailed setting of modulation is described in section 6.3.11.

7 APPLICATION CIRCUIT

7.1 Power Supply Scheme



7.2 Peripheral Application Scheme



Note 1: It is recommended to use 100 kΩ pull-up resistor on both ICE_DAT and ICE_CLK pin.

Note 2: It is recommended to use 10 kΩ pull-up resistor and 10 uF capacitor on nRESET pin.

8 ELECTRICAL CHARACTERISTICS

Please refer to the relative Datasheet for detailed information about the M031/M032 electrical characteristics.

9 ABBREVIATIONS

9.1 Abbreviations

Acronym	Description
ACMP	Analog Comparator Controller
ADC	Analog-to-Digital Converter
AES	Advanced Encryption Standard
APB	Advanced Peripheral Bus
AHB	Advanced High-Performance Bus
BOD	Brown-out Detection
CAN	Controller Area Network
DAP	Debug Access Port
DES	Data Encryption Standard
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	4~32 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 9.1-1 List of Abbreviations

10 REVISION HISTORY

Date	Revision	Description
2018.12.24	1.00	Initial version.
2019.02.25	1.01	1. Modified ISP ROM size in section 3.2. 2. Modified Figure 6.7-1 in section 6.7.3. 3. Modified HIRC trim reference clock in section 6.3.9, section 6.27.2 and section 6.27.3. 4. Modified description of PDMA_TACTSTS register in section 6.6.7.
2019.05.15	1.02	1. Modified description of USBD_EPSTS0 register in section 6.22.7. 2. Modified contents of section 6.25.5.9 Internal Reference Voltage. 3. Added Built-in V_{BG} ADC conversion result in Table 6.4-4 ISP Command List.
2019.07.15	1.03	1. Updated Figure 6.3-6 to add a USB block and remove the temperature sensor block. 2. Added multi-function pin tables in section 4.1.
2019.11.04	2.00	Added new part numbers for M031xI / M032xI / M031xG / M032xG / M032xC / M032xD and updated the description of the new part numbers.
2020.04.29	2.01	1. Modified Multi-function Pin Diagram name and Multi-function Pin Table in section 4.1.4.1 and 4.1.4.2. 2. Changed the Pin Description tables to Pin Mapping tables and Pin Function Description table in section 4.2 and 4.3.. 3. Modified description of Flash Access Time Control Register in section 6.4.6. 4. Modified QSPI Clock Divider Register's frequency name from SPI to QSPI in section 6.15.9.. 5. Modified the value of SYS_RSTSTS after Power-On-Reset(POR) in Table 6.3-1. 6. Added notes about the hardware reference design for ICE_DAT, ICE_CLK and nRESET pins in section 4.3 and chapter 7.
2020.09.29	2.02	1. Added a new part number M031TE3AE and updated the description of the new part number in chapter 3 and 4. 2. Modified the format of the BPWM_WGCTL0 register table in section 6.11.7. 3. Modified the basic configuration of USCI-UART and USCI-I2C in section 6.18.4 and 6.20.4.

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.