ARM926EJ-S™ Based

32-bit Microprocessor

NUC980 Series
Technical Reference Manual

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

Nuvoton is providing this document only for reference purposes of NUC980 microprocessor based system design. Nuvoton assumes no responsibility for errors or omissions.

All data and specifications are subject to change without notice.

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

www.nuvoton.com

NUC980 SERIES TECHNICAL REFERENCE MANUAL

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 GENERAL DESCRIPTION

The NUC980 series 32-bit microprocessor is powered by the ARM926EJ-S™ processor core with 16 KB I-cache, 16 KB D-cache and MMU running up to 300 MHz. Its SDRAM interface supports SDR/DDR/DDR2/LPDDR type SDRAM running up to 150 MHz. The NUC980 series supports built-in 16 KB embedded SRAM and 16.5 KB IBR (Internal Boot ROM) for booting from USB, NAND, SD/eMMC and SPI Flash, and industrial operating temperature from -40°C to 85°C. In addition, the NUC980 series provides built-in DDR in LQFP package to ease PCB design and reduce the BOM cost.

The NUC980 series is equipped with a large number of high speed digital peripherals, such as two 10/100 Mbps Ethernet MAC supporting RMII, a USB 2.0 high speed host/device and a USB 2.0 high speed host controller, up to six USB 1.1 host lite interfaces, two CMOS sensor interfaces supporting CCIR601 and CCIR656 type sensor, two SD interfaces supporting SD/SDHC/SDIO card, a NAND Flash interface supporting SLC and MLC type NAND Flash, an I^{2}S interface supporting I^{2}S and PCM protocol. Also, the NUC980 series offers a built-in hardware cryptography accelerator supporting RSA, ECC, AES, SHA, HMAC and a random number generator (RNG).

The NUC980 series provides up to ten UART interfaces, two ISO-7816-3 interfaces, a Quad-SPI interface, two SPI interfaces, up to four I^{2}C interfaces, four CAN 2.0B interfaces, eight channels PWM output, 8-channel 12-bit SAR ADC, six 32-bit timers, WDT (Watchdog Timer), WWDT (Window Watchdog Timer), 32.768 kHz XTL and RTC (Real Time Clock). The NUC980 series also supports two 10-channel peripheral DMA (PDMA) for automatic data transfer between memories and peripherals.

Key Features

- 300 MHz Arm® ARM926EJ-S™ MPU with 16 KB I-cache, 16 KB D-cache
- Memory Manager Unit (MMU)
- Built-in 16 MB /64 MB/128 MB DDR Memory in LQFP package
- Supports booting from SPI ROM/SPI NAND Flash/NAND/eMMC/SD Card and USB device
- Supports up to 100MHz Quad-SPI
- Dual Ethernet MAC
- Four CAN 2.0B interfaces
- Six USB FS Lite hosts
- Two USB High speed hosts
- One USB High speed device
- Two CCIR656/601 Camera interfaces
- Supports PRNG, AES256, SHA, ECC, and RSA2048

Applications

- Smart Home gateway
- Fingerprint Machine.
- Power concentrator
- Data Collector
- Smart Home Appliance
- Serial server
- 2D/1D Barcode reader
- Barcode printer
- Power Distribution Unit
- Ethernet Industrial Control
- SNMP Card
- Ethernet RTU / DTU

2 FEATURES DESCRIPTION

Core And System	
Boot Loader	• Factory pre-loaded 16.5 KB mask ROM supporting four booting modes – Boot from USB – Boot from SD/eMMC – Boot from NAND Flash – Boot from SPI Flash (SPI-NOR/SPI-NAND)
ARM926EJ-S™	• ARM926EJ-S™ processor core running up to 300 MHz • Built-in 16 KB instruction cache and 16 KB data cache • Built-in Memory Management Unit (MMU) • Supports JTAG debug interface
Advanced Interrupt Controller	• Up to 64 interrupt sources including 4 external interrupts. • Configurable normal (IRQ) or fast interrupt mode (FIQ). • Configurable 8-level interrupt priority scheme.
Low Voltage Detect (LVD)	• Two-level LVD with low voltage detect interrupt. (2.8V/2.6V)
Low Voltage Reset (LVR)	• LVR with 2.4V threshold voltage level.
Memories	
SDRAM	• Supports SDR (Single-Data-Rate), DDR (Double-Data-Rate), DDR2 (Double-Data-Rate 2) and LPDDR (Low Power DDR) SDRAM • Clock speed up to 150 MHz • Supports 16-bit data width
SRAM	• Up to 16 KB on-chip SRAM • Byte-, half-word- and word-access • PDMA operation
Peripheral DMA (PDMA)	• Two sets of PDMA with ten 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 • Stride function for rectangle image data movement • 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	• 12 MHz High-speed eXternal crystal oscillator (HXT) for precise timing operation • 32.7688 kHz Low-speed eXternal crystal oscillator (LXT) for RTC function and

	low-power system operation
Internal Clock Source	• Two on-chip PLL up to 500 MHz on-chip PLL, sourced from HXT, allows CPU operation up to the maximim CPU frequency without the need for a high-frequency crystal
Real-Time Clock (RTC)	• Real-Time Clock with a separate power domain (VBAT33) • The RTC clock source includes Low-speed external crystal oscillator (LXT) • The RTC block includes 64 bytes backup registers • Able to wake up CPU • 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 • Selectable 12-hour or 24-hour mode • Automatic leap year recognition • Supports 1 Hz clock to be Timer capture source for calibration
Timers	
32-bit Timer	• Six 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 internal capture source from RTC 1 Hz clock for interval measurement resetting 24-bit up counter • Supports chip wake-up function, if a timer interrupt signal is generated
PWM (PWM)	• Eight 16-bit down-count counters with four 8-bit prescalar for eight PWM output channels. • Supports complementary mode for 4 complementary paired PWM output channels
Watchdog	• 18-bit free running up counter for WDT time-out interval • Supports multiple clock sources from HXT, HXT/512 (default selection), PCLK2/4096 or LXT with 8 selectable time-out period • Able to wake up system from Power-down or Idle mode • Time-out event to trigger interrupt or reset system • Supports four WDT reset delay periods, including 1026, 130, 18 or 3 WDT_CLK reset delay period • Configured to force WDT enabled on chip power-on or reset.
Window Watchdog	• Clock sourced from HXT, HXT/512 (default selection), PCLK2/4096 or LXT; the window set by 6-bit counter with 11-bit prescale • Suspended in Idle/Power-down mode
Analog Interfaces	
Analog-to-Digital Converter (ADC)	• One 12-bit, 9-ch 200 KSPS SAR ADC with up to 8 single-ended input channels; 10-bit accuracy is guaranteed.

	• One internal channels for band-gap VBG input. • Supports external V_{REF} pin.
Communication Interfaces	
Low-power UART	• 10 sets of UARTs with up to 17.45 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 LIN function on UART0 and UART1 • Supports RS-485 9-bit mode and direction control • Supports nCTS, incoming data, Received Data FIFO reached threshold and RS-485 Address Match (AAD mode) wake-up function in idle mode. • Supports hardware or software 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.
Smart Card Interface	• Two sets of ISO-7816-3 which are compliant with ISO-7816-3 T=0, T=1 • Supports full duplex UART function. • 4-byte FIFOs with programmable level trigger • Programmable guard time selection (11 ETU ~ 266 ETU) • One 24-bit and two 8 bit time-out counters for Answer to Request (ATR) and waiting times processing • Auto inverse convention function • Stop clock level and clock stop (clock keep) function • Transmitter and receiver error retry function • Supports hardware activation, deactivation and warm reset sequence process • Supports hardware auto deactivation sequence after card removal.
I^{2}C	• Four sets of I^{2}C devices with Master/Slave mode. • Supports Standard mode (100 kbps), Fast mode (400 kbps) and Fast mode plus (1 Mbps) • Supports 10 bits mode • Programmable clocks allowing for versatile rate control • Supports multiple address recognition (four slave address with mask option) • Supports SMBus and PMBus • Supports multi-address power-down wake-up function
Quad SPI	• One set of SPI Quad controller with Master/Slave mode, up to 96 MHz at 2.7V~3.6V stsyem voltage.

	• Supports Dual and Quad I/O Transfer mode • Supports one/two 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.
SPI	• Up to two sets of SPI controllers with Master/Slave mode. • SPI provides separate 4-level of 32-bit (or 8-level of 16-bit) transmit and receive FIFO buffers. • Able to communicate at up to 96 Mbit/s • Configurable bit length of a transfer word from 8 to 32-bit. • MSB first or LSB first transfer sequence. • Byte reorder function. • Supports Byte or Word Suspend mode. • Supports one data channel half-duplex transfer. • Supports receive-only mode. • PDMA operation.
I^{2}S	• One set of I^{2}S controller with I^{2}S protocol and PCM protocol. • Supports mono and stereo audio data with 8-, 16- and 24-bit word sizes. • Four 8-level 24-bit FIFO data buffers for left/right channel record and left/right playback. • Built-in DMA function • Supports 2 buffer address for left/right channel and 2 slots data transfer. **I^{2}S Mode** • Supports record and playback. • Supports master and slave mode. • Supports Philips standard and MSB-justified data format. **PCM Mode** • Supports record and playback. • Supports master mode. • Supports PCM standard data format.
Controller Area Network (CAN)	• Four CAN 2.0B interfaces • Each supports 32 Message Objects; each Message Object has its own identifier mask. • Programmable FIFO mode (concatenation of Message Object). • Disabled Automatic Re-transmission mode for Time Triggered CAN applications.

	• Supports power-down wake-up function.
Secure Digital Host Controller (SDHC)	• Two sets of Secure Digital Host Controllers, compliant with SD Memory Card Specification Version 2.0. • Supports 50 MHz to achieve 200 Mbps at 3.3V operation. • Supports dedicated DMA master with Scatter-Gather function to accelerate the data transfer between system memory and SD/SDHC/SDIO card.
NAND Flash Controller	• Supports SLC and MLC type NAND Flash device. • Supports 2KB, 4KB and 8KB page size NAND Flash device. • 8-bit data width. • Supports ECC8, ECC12 and ECC24 BCH algorithm with ECC code generation, error detection and error correction. • Supports dedicated DMA master with Scatter-Gather function to accelerate the data transfer between system memory and NAND Flash.
External Bus Interface (EBI)	• Supports up to three 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. • 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. • Supports LCD interface i80 mode. • PDMA operation.
GPIO	• Supports four I/O modes: Bi-direction, Push-Pull output, Open-Drain output and Input only with high impendence mode. • Selectable TTL/Schmitt trigger input. • Configured as interrupt source with edge/level trigger setting. • Supports independent pull-up/pull-down control. • Supports 5V-tolerance function except analog I/O. (Except PB.0 ~ 7; All USB High Speed PIN.)
Advanced Connectivity	
USB 1.1 Host Lite	• Compliant with USB Revision 2.0 Specification. • Compatible with OHCI (Open Host Controller Interface) Revision 1.0. • Supports full-speed (12Mbps) and low-speed (1.5Mbps) USB devices. • Supports Control, Bulk, Interrupt, Isochronous and Split transfers. • Supports an integrated Root Hub. • Up to six USB Host Lite ports. • Built-in DMA.
USB 2.0 High Speed with on-chip transceiver	**USB 2.0 High Speed Host/Device** • One set of on-chip USB 2.0 high speed dual role transceiver configurable as host, device or ID-dependent. • One set of on-chip USB 2.0 high speed transceiver with host only.

	USB 2.0 High Speed Host Controller • Compliant with USB Revision 2.0 Specification. • Compatible with EHCI (Enhanced Host Controller Interface) Revision 1.0. • Compatible with OHCI (Open Host Controller Interface) Revision 1.0. • Supports high-speed (480Mbps), full-speed (12Mbps) and low-speed (1.5Mbps) USB devices. • Integrated with a port routing logic to route full/low speed device to OHCI controller. • Supports an integrated Root Hub. • Built-in DMA. **USB 2.0 High Speed Device Controller** • Compliant with USB Revision 2.0 Specification. • Supports one dedicate control endpoint and 12 configurable endpoints; each can be Isochronous, Bulk or Interrupt and either IN or OUT direction. • 4096 bytes configurable RAM for endpoint buffer and up to 1024 bytes packet size. • Three different operation modes of an in-endpoint: Auto Validation mode, Manual Validation mode and Fly mode. • Suspend, resume and remote wake-up capability. • Built-in DMA.
Ethernet MAC	• IEEE Std. 802.3 CSMA/CD protocol. • Ethernet frame time stamping for IEEE Std. 1588 – 2002 protocol. • Two sets of Ethernet MAC. • Supports both half and full duplex for 10 Mbps or 100 Mbps operation. • RMII (Reduced Media Independent Interface) and serial management interface (MDC/MDIO). • Pause and remote pause function for flow control. • Long frame (more than 1518 bytes) and short frame (less than 64 bytes) reception. • CAM function for Ethernet MAC address recognition. • Supports Magic Packet recognition to wake system up from Power-down mode. • Built-in DMA.
CMOS Sensor Interface	• Two sets of CMOS sensor interfaces supporting CCIR601 and CCIR656 type sensor. • Resolution up to 3M pixels. • Supports YUV422 and RGB565 color format for data output by CMOS image sensor. • Supports YUV422, RGB565, RGB555 and Y-only color format with planar and packet data format for data storing to system memory. • Supports image cropping and cropping window up to 4096x2048. • Supports vertical and horizontal scaling-down with N/M scaling factor. • Supports Negative, Sepia and Posterization color effects
Cryptography Accelerator	

Rivest、Shamir and Adleman Cryptography (RSA)	• Hardware RSA accelerator. • Supports both encryption and decryption. • Supports up to 2048 bits.
Elliptic Curve Cryptography (ECC)	• Hardware ECC accelerator. • Supports 192-bit and 256-bit key length. • Supports both prime field GF(p) and binary field GF(2m). • Supports NIST P-192, P-224, P-256, P-384 and P-521 curve sizes. • Supports NIST B-163, B-233, B-283, B-409 and B-571 curve sizes. • Supports NIST K-163, K-233, K-283, K-409 and K-571 curve sizes. • Supports point multiplication, addition and doubling operations in GF(p) and GF(2m). • Supports modulus division, multiplication, addition and subtraction operations in GF(p).
Advanced Encryption Standard (AES)	• Hardware AES accelerator. • Supports 128-bit, 192-bit and 256-bit key length and key expander, and compliant with FIPS 197. • Supports ECB, CBC, CFB, OFB, CTR, CBC-CS1, CBC-CS2 and CBC-CS3 block cipher modes • Compliant with NIST SP800-38A and addendum.
Secure Hash Algorithm (SHA)	• Hardware SHA accelerator. • Supports SHA-160, SHA-224, SHA-256, SHA-384 and SHA-512. • Compliant with FIPS 180/180-2.
keyed-Hash Message Authentication Code (HMAC)	• Hardware HMAC accelerator. • Supports HMAC-SHA-160, HMAC-SHA-224, HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512. • Compliant with FIPS 180/180-2.
PRNG	• Supports 64-/128-/192-/256-bit random number generator.

3 PARTS INFORMATION

3.1 Package Type

Part No.	LQFP64-EP	LQFP128	LQFP216
NUC980	NUC980DRxxY, NUC980DRxxYC	NUC980DKxxY, NUC980DKxxYC	NUC980DFxxY, NUC980DFxxYC

Table 3.1-1 NUC980 Series Package Type

3.2 NUC980 Series Selection Guide

Part Number	NUC980											
	DF71YC	DF71Y	DF61YC	DK71YC	DK71Y	DK61YC	DK61Y	DK41Y	DR61YC	DR61Y	DR41YC	DR41Y
DDR Size(MB)	128	128	64	128	128	64	64	16	64	64	16	16
I/O	104	104	104	92	92	92	92	92	40	40	40	40
Boot Source	NAND QSPI NAND QSPI NOR SPI NOR eMMC SD Card	NAND QSPI NAND QSPI NOR SPI NOR eMMC SD Card	NAND QSPI NAND QSPI NOR SPI NOR eMMC SD Card	NAND QSPI NAND QSPI NOR SPI NOR eMMC SD Card	NAND QSPI NAND QSPI NOR SPI NOR eMMC SD Card	NAND QSPI NAND QSPI NOR SPI NOR eMMC SD Card	NAND QSPI NAND QSPI NOR SPI NOR eMMC SD Card	NAND QSPI NAND QSPI NOR SPI NOR eMMC SD Card	SPI NOR	SPI NOR	SPI NOR	SPI NOR
RTC	√	√	√	√	√	√	√	√	-	-	-	-
UART	10	10	10	10	10	10	10	10	8	8	8	8
ISO-7816	2	2	2	2	2	2	2	2	2	2	2	2
SPI	3	3	3	3	3	3	3	3	2	2	2	2
I^{2}S	1	1	1	1	1	1	1	1	1	1	1	1
I^{2}C	4	4	4	4	4	4	4	4	2	2	2	2
CAN	4	-	4	4	-	4	-	-	2	-	2	-
SDHC/SDIO/eMMC	2	2	2	2	2	2	2	2	1	1	1	1
Crypto	√	√	√	√	√	√	√	√	√	√	√	√
External Bus Interface	√	√	√	√	√	√	√	√	-	-	-	-
Camera Interface	2	2	2	2	2	2	2	2	1	1	1	1
16-bit PWM	8	8	8	8	8	8	8	8	5	5	5	5
10/100Mb Ethernet MAC	2	2	2	2	2	2	2	2	1	1	1	1
USB 1.1 FS Host Lite	6	6	6	6	-	6	-	-	-	-	-	-
USB 2.0 HS Host	1	1	1	1	1	1	1	1	-	-	-	-
USB 2.0 HS Host / Device	1	1	1	1	1	1	1	1	1	1	1	1
12-bit ADC	8	8	8	8	8	8	8	8	2	2	2	2
Package	LQFP216	LQFP216	LQFP216	LQFP 128	LQFP 128	LQFP 128	LQFP 128	LQFP 128	LQFP 64 - EP	LQFP 64 - EP	LQFP 64 - EP	LQFP 64 - EP

Table 3.2-1 NUC980 Series Part Selection Guide

3.3 NUC980 Series Naming Rule

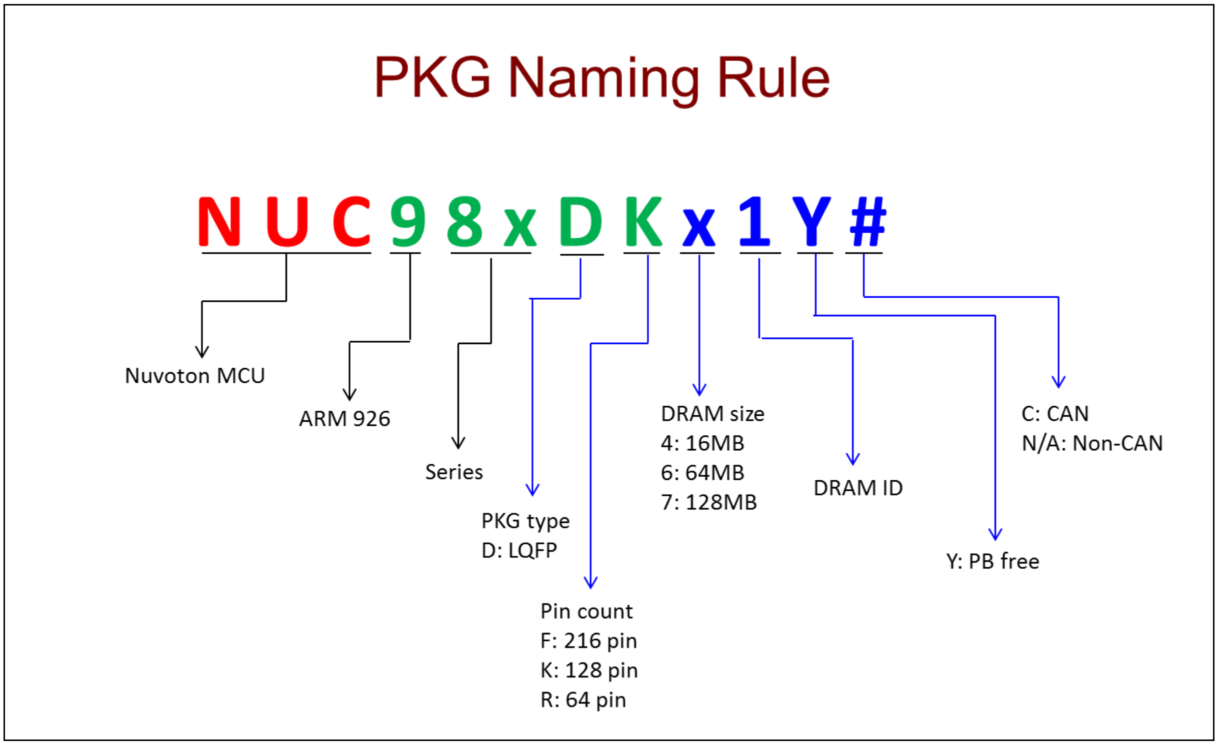


Figure 3.3-1 NUC980 Series Selection Code

4 PIN CONFIGURATION

4.1 Pin Configuration

4.1.1 NUC980DRxxY LQFP64-EP Pin Diagram

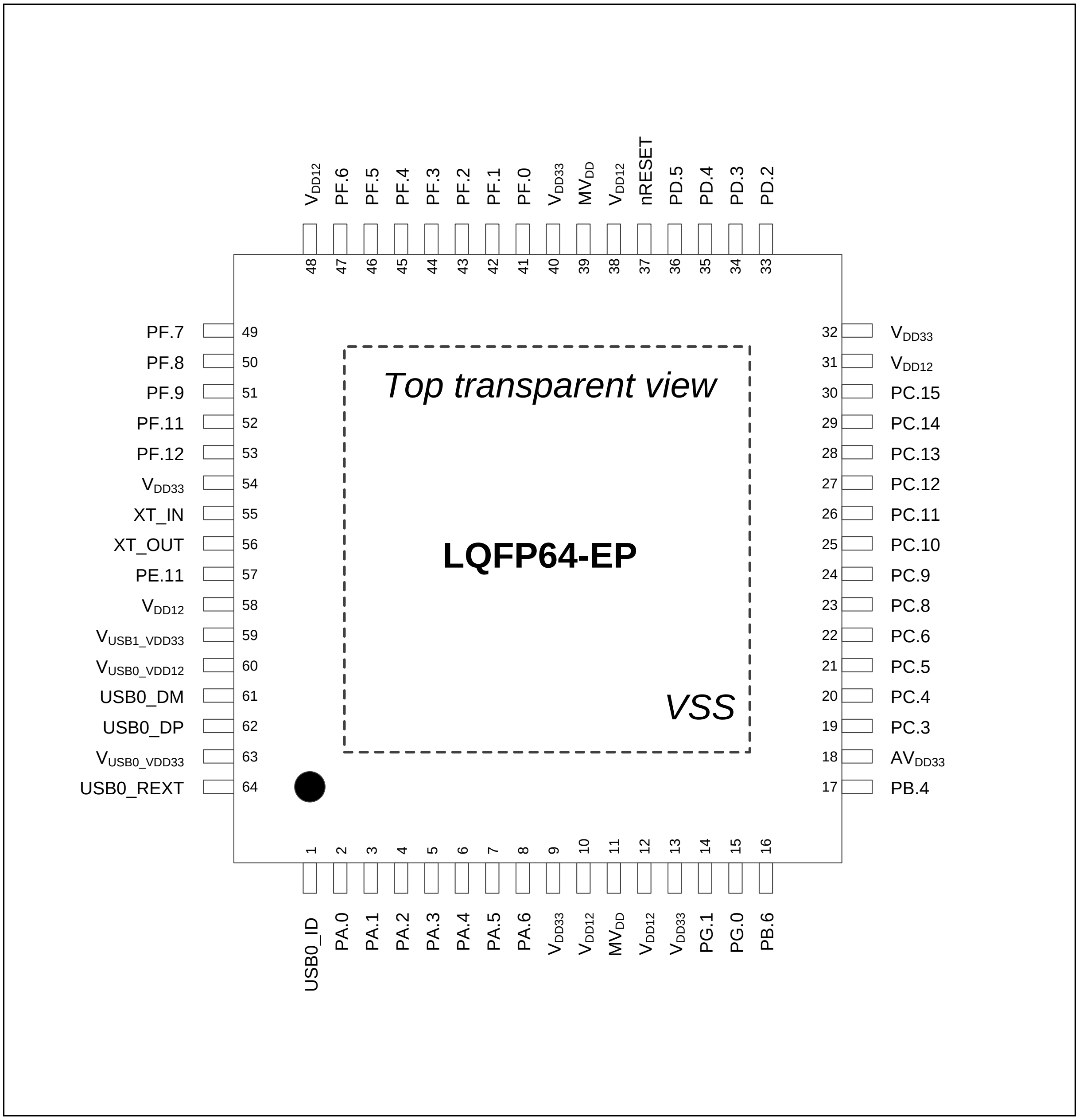


Figure 4.1-1 NUC980DRxxY LQFP 64-pin with EX-PAD Diagram

4.1.2 NUC980DKxxY LQFP128 Pin Diagram

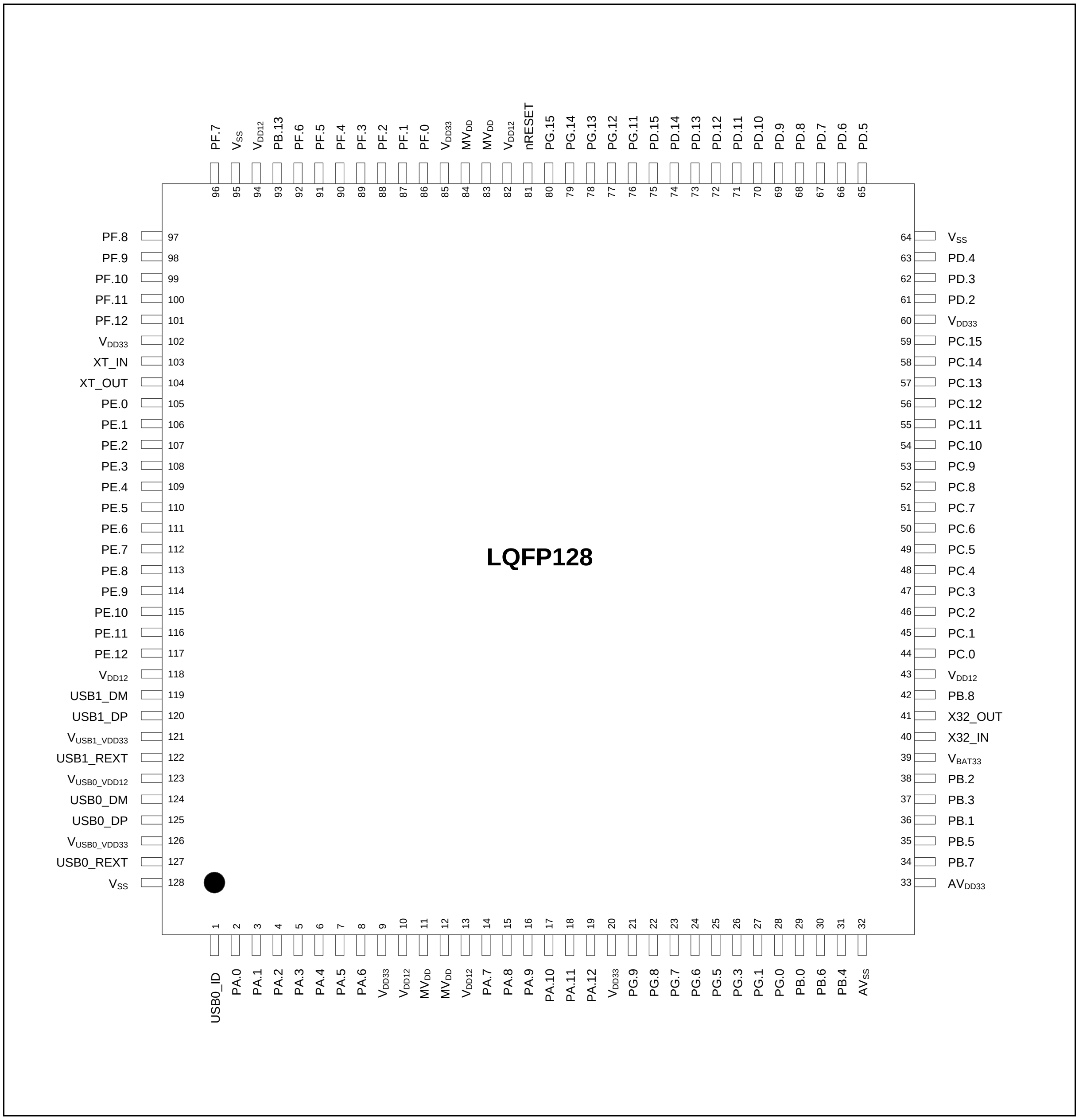


Figure 4.1-2 NUC980DKxxY LQFP 128-pin Diagram

4.1.3 NUC980DKxxYC LQFP128 Pin Diagram

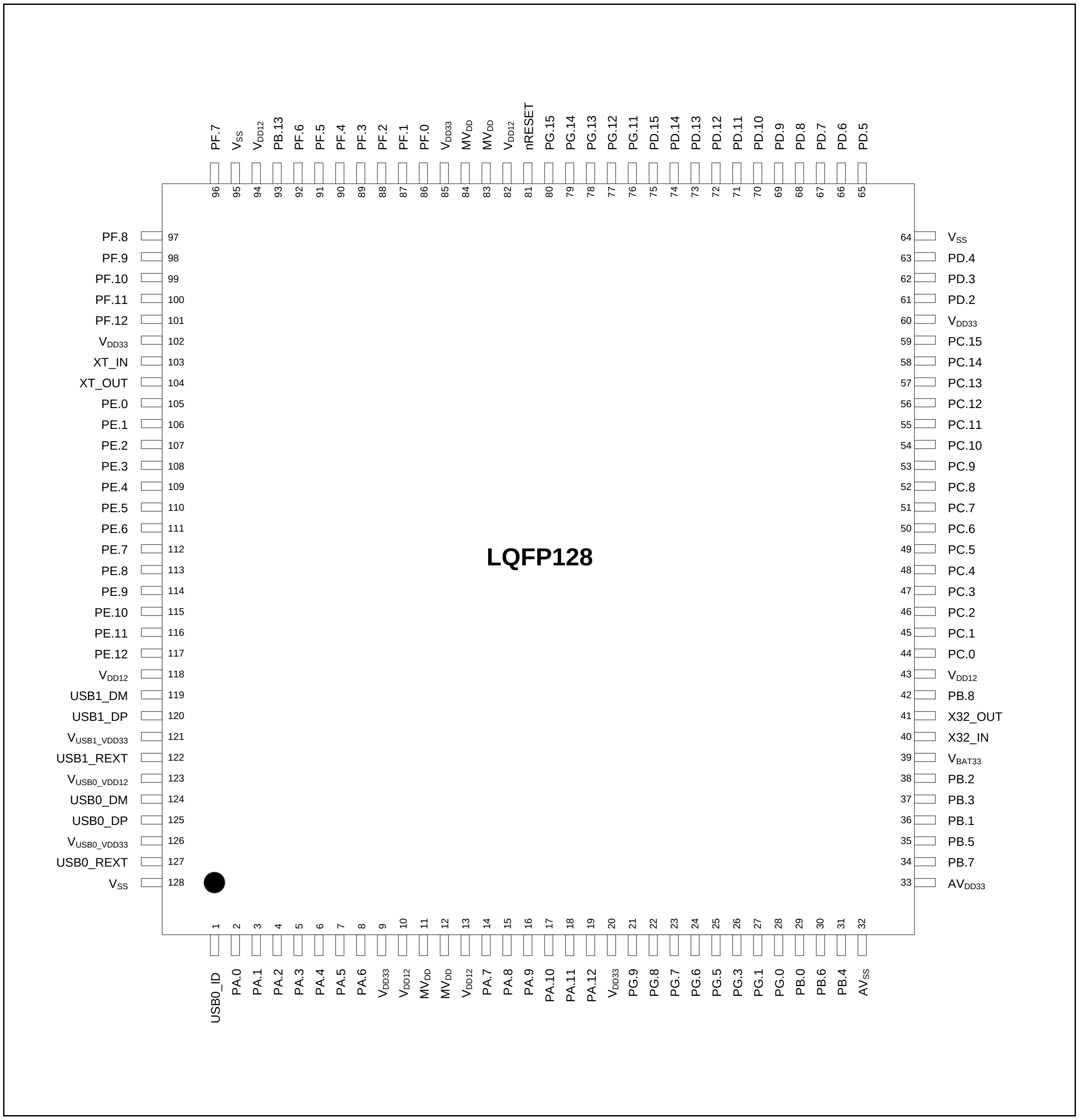


Figure 4.1-3 NUC980DKxxYC LQFP 128-pin Diagram

4.1.4 NUC980DFxxYC LQFP216 Pin Diagram

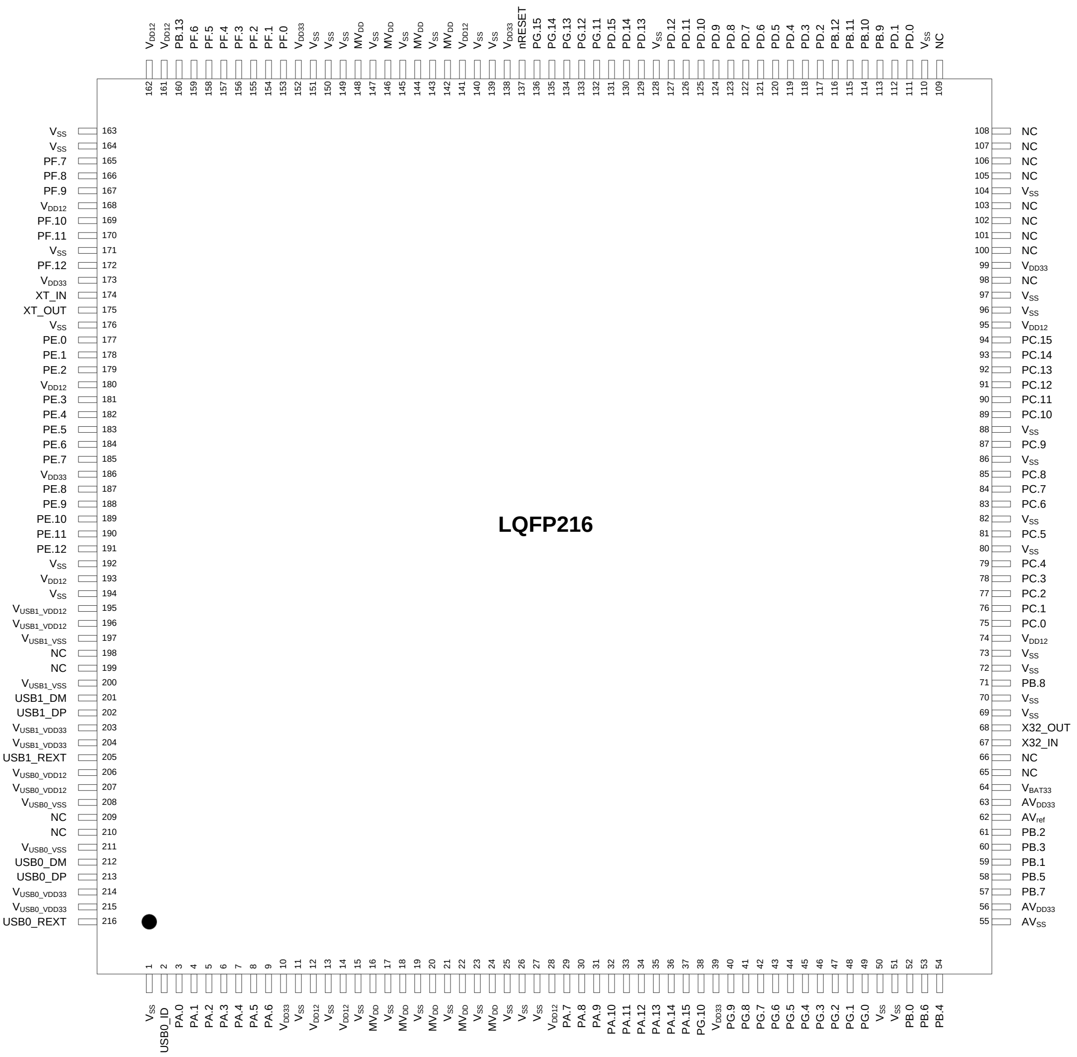


Figure 4.1-4 NUC980DFxxY LQFP 216-pin Diagram

4.2 Pin Description

4.2.1 NUC980 Pin Description

64 Pin	128 Pin	216 Pin	Pin Name	Type	MFP	Description
		1	VSS	P	MFP0	Ground pin.
1	1	2	USB0_ID	IU	-	USB0 Host/Device identification with an internal pull-up. 1 = Device (Default) 0 = Host
2	2	3	PA.0	I/O	MFP0	General purpose digital I/O pin.
			QSPI0_SS1	I/O	MFP1	Quad SPI0 slave select 1 pin.
			I2C0_SDA	I/O	MFP3	I^{2}C0 data input/output pin.
			UART1_RXD	I	MFP4	UART1 data receiver input pin.
			EINT0	I	MFP5	External interrupt 0 input pin.
			TM0_ECNT	I/O	MFP6	Timer0 event counter input/toggle output pin.
			CAN3_RXD	I	MFP7	CAN3 bus receiver input.
3	3	4	PA.1	I/O	MFP0	General purpose digital I/O pin.
			EBI_nCS2	O	MFP1	EBI chip select 2 output pin.
			EBI_MCLK	O	MFP2	EBI external clock output pin.
			I2C0_SCL	I/O	MFP3	I^{2}C0 clock pin.
			UART1_TXD	O	MFP4	UART1 data transmitter output pin.
			EINT1	I	MFP5	External interrupt 1 input pin.
			TM1_ECNT	I/O	MFP6	Timer1 event counter input/toggle output pin.
			CAN3_TXD	O	MFP7	CAN3 bus transmitter output.
4	4	5	PA.2	I/O	MFP0	General purpose digital I/O pin.
			UART6_CTS	I	MFP1	UART6 clear to Send input pin.
			I2S_LRCK	O	MFP2	I^{2}S_ left right channel clock output pin.
			SC0_CD	I	MFP3	Smart Card 0 card detect pin.
			JTAG1_TDO	O	MFP4	JTAG1 data output pin.
			TM2_ECNT	I/O	MFP6	Timer2 event counter input/toggle output pin.
5	5	6	PA.3	I/O	MFP0	General purpose digital I/O pin.
			UART6_RTS	O	MFP1	UART6 request to Send output pin.
			I2S_BCLK	O	MFP2	I^{2}S_ bit clock output pin.
			SC0_PWR	O	MFP3	Smart Card 0 power pin.
			JTAG1_TCK	I	MFP4	JTAG1 clock input pin.
			TM3_ECNT	I/O	MFP6	Timer3 event counter input/toggle output pin.

64 Pin	128 Pin	216 Pin	Pin Name	Type	MFP	Description
6	6	7	PA.4	I/O	MFP0	General purpose digital I/O pin.
			UART6_RXD	I	MFP1	UART6 data receiver input pin.
			I2S_DI	I	MFP2	I^{2}S_ data input pin.
			SC0_DAT	I/O	MFP3	Smart Card 0 data pin.
			JTAG1_TMS	I	MFP4	JTAG1 test mode selection input pin.
			TM4_ECNT	I/O	MFP6	Timer4 event counter input/toggle output pin.
7	7	8	PA.5	I/O	MFP0	General purpose digital I/O pin.
			UART6_TXD	O	MFP1	UART6 data transmitter output pin.
			I2S_DO	O	MFP2	I^{2}S_ data output pin.
			SC0_CLK	O	MFP3	Smart Card 0 clock pin.
			JTAG1_TDI	I	MFP4	JTAG1 data input pin.
			TM5_ECNT	I/O	MFP6	Timer5 event counter input/toggle output pin.
8	8	9	PA.6	I/O	MFP0	General purpose digital I/O pin.
			EBI_nCS1	O	MFP1	EBI chip select 1 output pin.
			I2S_MCLK	O	MFP2	I^{2}S_ master clock output pin.
			SC0_RST	O	MFP3	Smart Card 0 reset pin.
			JTAG1_nTRST	I	MFP4	JTAG1 reset input pin.
9	9	10	VDD33	P	MFP0	Power supply for I/O ports.
		11	VSS	P	MFP0	Ground pin
10	10	12	VDD12	P	MFP0	Power supply for internal digital circuit.
		13	VSS	P	MFP0	Ground pin.
		14	VDD12	P	MFP0	Power supply for internal digital circuit.
		15	VSS	P	MFP0	Ground pin.
11	11	16	MVDD	P	MFP0	Power supply for Memory ports.
		17	VSS	P	MFP0	Ground pin.
		18	MVDD	P	MFP0	Power supply for Memory ports.
		19	VSS	P	MFP0	Ground pin.
		20	MVDD	P	MFP0	Power supply for Memory ports.
		21	VSS	P	MFP0	Ground pin.
	12	22	MVDD	P	MFP0	Power supply for Memory ports.
		23	VSS	P	MFP0	Ground pin.
		24	MVDD	P	MFP0	Power supply for Memory ports.
		25	VSS	P	MFP0	Ground pin.

64 Pin	128 Pin	216 Pin	Pin Name	Type	MFP	Description
		26	VSS	P	MFP0	Ground pin.
		27	VSS	P	MFP0	Ground pin.
12	13	28	VDD12	P	MFP0	Power supply for internal digital circuit.
	14	29	PA.7	I/O	MFP0	General purpose digital I/O pin.
			EBI_nWE	O	MFP1	EBI write enable output pin.
			UART2_CTS	I	MFP2	UART2 clear to Send input pin.
			TM3_EXT	I/O	MFP3	Timer3 external capture input/toggle output pin.
	15	30	PA.8	I/O	MFP0	General purpose digital I/O pin.
			EBI_nRE	O	MFP1	EBI read enable output pin.
			UART2_RTS	O	MFP2	UART2 request to Send output pin.
			TM3_TGL	I/O	MFP3	Timer3 event counter input/toggle output pin.
	16	31	PA.9	I/O	MFP0	General purpose digital I/O pin.
			EBI_nCS0	O	MFP1	EBI chip select 0 output pin.
			UART2_RXD	I	MFP2	UART2 data receiver input pin.
			TM2_EXT	I/O	MFP3	Timer2 external capture input/toggle output pin.
	17	32	PA.10	I/O	MFP0	General purpose digital I/O pin.
			EBI_ADDR10	O	MFP1	EBI address bus bit 10.
			UART2_TXD	O	MFP2	UART2 data transmitter output pin.
			TM2_TGL	I/O	MFP3	Timer2 event counter input/toggle output pin.
	18	33	PA.11	I/O	MFP0	General purpose digital I/O pin.
			EBI_ADDR9	O	MFP1	EBI address bus bit 9.
			UART8_RXD	I	MFP2	UART8 data receiver input pin.
			TM4_EXT	I/O	MFP3	Timer4 external capture input/toggle output pin.
	19	34	PA.12	I/O	MFP0	General purpose digital I/O pin.
			EBI_ADDR8	O	MFP1	EBI address bus bit 8.
			UART8_TXD	O	MFP2	UART8 data transmitter output pin.
			TM4_TGL	I/O	MFP3	Timer4 event counter input/toggle output pin.
		35	PA.13	I/O	MFP0	General purpose digital I/O pin.
			EBI_ADDR13	O	MFP1	EBI address bus bit 13.
			I2C1_SDA	I/O	MFP2	I^{2}C1 data input/output pin.
			TM1_EXT	I/O	MFP3	Timer1 external capture input/toggle output pin.
			USBHL5_DM	A	MFP4	USB 1.1 Host Lite 5 differential signal D-.
			CAN1_RXD	I	MFP5	CAN1 bus receiver input.

64 Pin	128 Pin	216 Pin	Pin Name	Type	MFP	Description
			UART7_TXD	O	MFP6	UART7 data transmitter output pin.
			PWM03	O	MFP7	PWM03 counter synchronous trigger output pin.
			EINT0	I	MFP8	External interrupt 0 input pin.
		36	PA.14	I/O	MFP0	General purpose digital I/O pin.
			EBI_ADDR14	O	MFP1	EBI address bus bit 14.
			I2C1_SCL	I/O	MFP2	I^{2}C1 clock pin.
			TM1_TGL	I/O	MFP3	Timer1 event counter input/toggle output pin.
			USBHL5_DP	A	MFP4	USB 1.1 Host Lite 5 differential signal D+.
			CAN1_TXD	O	MFP5	CAN1 bus transmitter output.
			UART7_RXD	I	MFP6	UART7 data receiver input pin.
			PWM02	O	MFP7	PWM02 counter synchronous trigger output pin.
			EINT1	I	MFP8	External interrupt 1 input pin.
		37	PA.15	I/O	MFP0	General purpose digital I/O pin.
			EBI_ADDR19	O	MFP1	EBI address bus bit 19.
			I2C0_SDA	I/O	MFP2	I^{2}C0 data input/output pin.
			TM5_EXT	I/O	MFP3	Timer5 external capture input/toggle output pin.
			USBHL4_DM	A	MFP4	USB 1.1 Host Lite 4 differential signal D-.
			CAN2_RXD	I	MFP5	CAN2 bus receiver input.
			SPI1_SS0	I/O	MFP6	SPI1 slave select 0 pin.
			PWM01	O	MFP7	PWM01 counter synchronous trigger output pin.
			I2S_LRCK	O	MFP8	I^{2}S_ left right channel clock output pin.
		38	PG.10	I/O	MFP0	General purpose digital I/O pin.
			EBI_DATA0	I/O	MFP1	EBI data bus bit 0.
			I2C0_SCL	I/O	MFP2	I^{2}C0 clock pin.
			TM5_TGL	I/O	MFP3	Timer5 event counter input/toggle output pin.
			USBHL4_DP	A	MFP4	USB 1.1 Host Lite 4 differential signal D+.
			CAN2_TXD	O	MFP5	CAN2 bus transmitter output.
			SPI1_CLK	I/O	MFP6	SPI1 serial clock pin.
			PWM00	O	MFP7	PWM00 counter synchronous trigger output pin.
			I2S_BCLK	O	MFP8	I^{2}S_ bit clock output pin.
13	20	39	VDD33	P	MFP0	Power supply for I/O ports.
	21	40	PG.9	I/O	MFP0	General purpose digital I/O pin.
			EBI_ADDR7	O	MFP1	EBI address bus bit 7.

64 Pin	128 Pin	216 Pin	Pin Name	Type	MFP	Description
			UART8_CTS	I	MFP2	UART8 clear to Send input pin.
			PWM13	O	MFP6	PWM13 counter synchronous trigger output pin.
			CFG.9_PwrOnSet9	IU	-	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
	22	41	PG.8	I/O	MFP0	General purpose digital I/O pin.
			EBI_ADDR6	O	MFP1	EBI address bus bit 6.
			UART8_RTS	O	MFP2	UART8 request to Send output pin.
			PWM12	O	MFP6	PWM12 counter synchronous trigger output pin.
			CFG.8_PwrOnSet8	IU	-	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
	23	42	PG.7	I/O	MFP0	General purpose digital I/O pin.
			EBI_ADDR5	O	MFP1	EBI address bus bit 5.
			UART5_TXD	O	MFP2	UART5 data transmitter output pin.
			PWM11	O	MFP6	PWM11 counter synchronous trigger output pin.
			CFG.7_PwrOnSet7	IU	-	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
	24	43	PG.6	I/O	MFP0	General purpose digital I/O pin.
			EBI_ADDR4	O	MFP1	EBI address bus bit 4.
			UART5_RXD	I	MFP2	UART5 data receiver input pin.
			PWM10	O	MFP6	PWM10 counter synchronous trigger output pin.
			CFG.6_PwrOnSet6	IU	-	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
	25	44	PG.5	I/O	MFP0	General purpose digital I/O pin.
			EBI_ADDR12	O	MFP1	EBI address bus bit 12.
			UART5_RTS	O	MFP2	UART5 request to Send output pin.
			CFG.5_PwrOnSet5	IU	-	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
		45	PG.4	I/O	MFP0	General purpose digital I/O pin.
			EBI_ADDR18	O	MFP1	EBI address bus bit 18.
			UART5_CTS	I	MFP2	UART5 clear to Send input pin.
			CFG.4_PwrOnSet4	IU	-	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
	26	46	PG.3	I/O	MFP0	General purpose digital I/O pin.

64 Pin	128 Pin	216 Pin	Pin Name	Type	MFP	Description
			EBI_ADDR3	O	MFP1	EBI address bus bit 3.
			UART2_RTS	O	MFP2	UART2 request to Send output pin.
			PWM03	O	MFP6	PWM03 counter synchronous trigger output pin.
			CFG.3_PwrOnSet3	IU	-	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
		47	PG.2	I/O	MFP0	General purpose digital I/O pin.
			EBI_ADDR2	O	MFP1	EBI address bus bit 2.
			UART2_CTS	I	MFP2	UART2 clear to Send input pin.
			PWM02	O	MFP6	PWM02 counter synchronous trigger output pin.
			CFG.2_PwrOnSet2	IU	-	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
14	27	48	PG.1	I/O	MFP0	General purpose digital I/O pin.
			EBI_ADDR1	O	MFP1	EBI address bus bit 1.
			UART2_TXD	O	MFP2	UART2 data transmitter output pin.
			PWM01	O	MFP6	PWM01 counter synchronous trigger output pin.
			CFG.1_PwrOnSet1	IU	-	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
15	28	49	PG.0	I/O	MFP0	General purpose digital I/O pin.
			EBI_ADDR0	O	MFP1	EBI address bus bit 0.
			UART2_RXD	I	MFP2	UART2 data receiver input pin.
			CLK_OUT	O	MFP3	Internal clock selection output pin.
			PWM00	O	MFP6	PWM00 counter synchronous trigger output pin.
			CFG.0_PwrOnSet0	IU	-	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
		50	VSS	P	MFP0	Ground pin.
		51	VSS	P	MFP0	Ground pin.
	29	52	PB.0	I/O	MFP0	General purpose digital I/O pin.
			EBI_ADDR12	O	MFP1	EBI address bus bit 12.
			UART2_CTS	I	MFP2	UART2 clear to Send input pin.
			ADC_AIN0	A	MFP8	ADC channel 0 analog input.
16	30	53	PB.6	I/O	MFP0	General purpose digital I/O pin.
			EBI_ADDR13	O	MFP1	EBI address bus bit 13.
			I2C1_SDA	I/O	MFP2	I^{2}C1 data input/output pin.

64 Pin	128 Pin	216 Pin	Pin Name	Type	MFP	Description
			I2S_LRCK	O	MFP3	I^{2}S_ left right channel clock output pin.
			USBHL0_DM	A	MFP4	USB 1.1 Host Lite 0 differential signal D-.
			UART7_TXD	O	MFP5	UART7 data transmitter output pin.
			SPI1_SS0	I/O	MFP6	SPI1 slave select 0 pin.
			ADC_AIN6	A	MFP8	ADC channel 6 analog input.
17	31	54	PB.4	I/O	MFP0	General purpose digital I/O pin.
			EBI_ADDR14	O	MFP1	EBI address bus bit 14.
			I2C1_SCL	I/O	MFP2	I^{2}C1 clock pin.
			I2S_BCLK	O	MFP3	I^{2}S_ bit clock output pin.
			USBHL0_DP	A	MFP4	USB 1.1 Host Lite 0 differential signal D+.
			UART7_RXD	I	MFP5	UART7 data receiver input pin.
			SPI1_CLK	I/O	MFP6	SPI1 serial clock pin.
			ADC_AIN4	A	MFP8	ADC channel 4 analog input.
	32	55	AVSS	P	MFP0	Ground pin for analog SAR-ADC.
18	33	56	AVDD33	P	MFP0	Power supply for analog SAR-ADC, DC 3.3V.
	34	57	PB.7	I/O	MFP0	General purpose digital I/O pin.
			EBI_ADDR15	O	MFP1	EBI address bus bit 15.
			I2C2_SDA	I/O	MFP2	I^{2}C2 data input/output pin.
			I2S_DI	I	MFP3	I^{2}S_ data input pin.
			USBHL0_DM	A	MFP4	USB 1.1 Host Lite 0 differential signal D-.
			UART7_CTS	I	MFP5	UART7 clear to Send input pin.
			SPI1_MOSI	I/O	MFP6	SPI1 MOSI (Master Out, Slave In) pin.
			ADC_AIN7	A	MFP8	ADC channel 7 analog input.
	35	58	PB.5	I/O	MFP0	General purpose digital I/O pin.
			EBI_ADDR16	O	MFP1	EBI address bus bit 16.
			I2C2_SCL	I/O	MFP2	I^{2}C2 clock pin.
			I2S_DO	O	MFP3	I^{2}S_ data output pin.
			USBHL0_DP	A	MFP4	USB 1.1 Host Lite 0 differential signal D+.
			UART7_RTS	O	MFP5	UART7 request to Send output pin.
			SPI1_MISO	I/O	MFP6	SPI1 MISO (Master In, Slave Out) pin.
			ADC_AIN5	A	MFP8	ADC channel 5 analog input.
	36	59	PB.1	I/O	MFP0	General purpose digital I/O pin.
			EBI_ADDR17	O	MFP1	EBI address bus bit 17.

64 Pin	128 Pin	216 Pin	Pin Name	Type	MFP	Description
			I2C3_SDA	I/O	MFP2	I^{2}C3 data input/output pin.
			I2S_MCLK	O	MFP3	I^{2}S_ master clock output pin.
			CAN2_RXD	I	MFP4	CAN2 bus receiver input.
			TM0_EXT	I/O	MFP5	Timer0 external capture input/toggle output pin.
			SPI1_SS1	I/O	MFP6	SPI1 slave select 1 pin.
			UART9_TXD	O	MFP7	UART9 data transmitter output pin.
			ADC_AIN1	A	MFP8	ADC channel 1 analog input.
	37	60	PB.3	I/O	MFP0	General purpose digital I/O pin.
			EBI_ADDR18	O	MFP1	EBI address bus bit 18.
			I2C3_SCL	I/O	MFP2	I^{2}C3 clock pin.
			EINT2	I	MFP3	External interrupt 2 input pin.
			CAN2_TXD	O	MFP4	CAN2 bus transmitter output.
			TM0_TGL	I/O	MFP5	Timer0 event counter input/toggle output pin.
			SPI0_SS1	I/O	MFP6	SPI0 slave select 1 pin.
			UART9_RXD	I	MFP7	UART9 data receiver input pin.
			ADC_AIN3	A	MFP8	ADC channel 3 analog input.
	38	61	PB.2	I/O	MFP0	General purpose digital I/O pin.
			EBI_ADDR2	O	MFP1	EBI address bus bit 2.
			UART9_RTS	O	MFP7	UART9 request to Send output pin.
			ADC_AIN2	A	MFP8	ADC channel 2 analog input.
		62	AVref	A	MFP0	ADC reference voltage input.
		63	AVDD33	P	MFP0	Power supply for analog SAR-ADC, DC 3.3V.
	39	64	VBAT33	P	MFP0	Power supply by batteries for RTC.
		65	NC			
		66	NC			
	40	67	X32_IN	I	MFP0	External 32.768 kHz crystal input pin.
	41	68	X32_OUT	O	MFP0	External 32.768 kHz crystal output pin.
		69	VSS	P	MFP0	Ground pin.
		70	VSS	P	MFP0	Ground pin.
	42	71	PB.8	I/O	MFP0	General purpose digital I/O pin.
			EBI_ADDR11	O	MFP1	EBI address bus bit 11.
			I2C2_SCL	I/O	MFP2	I^{2}C2 clock pin.
			CAN2_RXD	I	MFP3	CAN2 bus receiver input.

NUC980 SERIES TECHNICAL REFERENCE MANUAL

64 Pin	128 Pin	216 Pin	Pin Name	Type	MFP	Description
			UART8_TXD	O	MFP4	UART8 data transmitter output pin.
			SD0_nCD	I	MFP6	SD/SDIO0 card detect input pin
			TM0_EXT	I/O	MFP7	Timer0 external capture input/toggle output pin.
		72	VSS	P	MFP0	Ground pin.
		73	VSS	P	MFP0	Ground pin.
	43	74	VDD12	P	MFP0	Power supply for internal digital circuit.
	44	75	PC.0	I/O	MFP0	General purpose digital I/O pin.
			EBI_DATA0	I/O	MFP1	EBI data bus bit 0.
			I2C2_SDA	I/O	MFP2	I^{2}C2 data input/output pin.
			CAN2_TXD	O	MFP3	CAN2 bus transmitter output.
			UART8_RXD	I	MFP4	UART8 data receiver input pin.
			SPI0_SS1	I/O	MFP5	SPI0 slave select 1 pin.
			TM0_TGL	I/O	MFP7	Timer0 event counter input/toggle output pin.
	45	76	PC.1	I/O	MFP0	General purpose digital I/O pin.
			EBI_DATA1	I/O	MFP1	EBI data bus bit 1.
			NAND_nCS0	O	MFP3	NAND Flash chip enable input.
			UART7_TXD	O	MFP4	UART7 data transmitter output pin.
	46	77	PC.2	I/O	MFP0	General purpose digital I/O pin.
			EBI_DATA2	I/O	MFP1	EBI data bus bit 2.
			NAND_nWP	O	MFP3	NAND Flash write protect input.
			UART7_RXD	I	MFP4	UART7 data receiver input pin.
19	47	78	PC.3	I/O	MFP0	General purpose digital I/O pin.
			EBI_DATA3	I/O	MFP1	EBI data bus bit 3.
			VCAP0_CLKO	O	MFP2	Video image interface 0 sensor clock pin.
			NAND_ALE	O	MFP3	NAND Flash address latch enable.
			I2C1_SCL	I/O	MFP4	I^{2}C1 clock pin.
			UART3_TXD	O	MFP5	UART3 data transmitter output pin.
			CAN0_RXD	I	MFP7	CAN0 bus receiver input.
20	48	79	PC.4	I/O	MFP0	General purpose digital I/O pin.
			EBI_DATA4	I/O	MFP1	EBI data bus bit 4.
			VCAP0_PCLK	I	MFP2	Video image interface 0 pixel clock pin.
			NAND_CLE	O	MFP3	NAND Flash command latch enable.
			I2C1_SDA	I/O	MFP4	I^{2}C1 data input/output pin.

64 Pin	128 Pin	216 Pin	Pin Name	Type	MFP	Description
			UART3_RXD	I	MFP5	UART3 data receiver input pin.
			SPI0_MOSI	I/O	MFP6	SPI0 MOSI (Master Out, Slave In) pin.
			CAN0_TXD	O	MFP7	CAN0 bus transmitter output.
		80	VSS	P	MFP0	Ground pin.
21	49	81	PC.5	I/O	MFP0	General purpose digital I/O pin.
			EBI_DATA5	I/O	MFP1	EBI data bus bit 5.
			VCAP0_HSYNC	I	MFP2	Video image interface 0 horizontal sync. pin.
			NAND_nWE	O	MFP3	NAND Flash write enable.
			SPI0_SS0	I/O	MFP5	SPI0 slave select 0 pin.
			SD0_CMD/eMMC0_CMD	I/O	MFP6	SD/SDIO0 command/response pin eMMC0 command/response pin
			UART1_TXD	O	MFP7	UART1 data transmitter output pin.
		82	VSS	P	MFP0	Ground pin.
22	50	83	PC.6	I/O	MFP0	General purpose digital I/O pin.
			EBI_DATA6	I/O	MFP1	EBI data bus bit 6.
			VCAP0_VSYNC	I	MFP2	Video image interface 0 vertical sync. pin.
			NAND_nRE	O	MFP3	NAND Flash read enable.
			SC1_RST	O	MFP4	Smart Card 1 reset pin.
			SPI0_CLK	I/O	MFP5	SPI0 serial clock pin.
			SD0_CLK/eMMC0_CLK	O	MFP6	SD/SDIO0 clock output pin eMMC0 clock output pin
			UART1_RXD	I	MFP7	UART1 data receiver input pin.
	51	84	PC.7	I/O	MFP0	General purpose digital I/O pin.
			EBI_DATA7	I/O	MFP1	EBI data bus bit 7.
			VCAP0_FIELD	I	MFP2	Video image interface 0 frame sync. pin.
			NAND_RDY0	I	MFP3	NAND Flash ready/busy pin.
			SC1_CLK	O	MFP4	Smart Card 1 clock pin.
			SPI0_MOSI	I/O	MFP5	SPI0 MOSI (Master Out, Slave In) pin.
			SD0_DATA0/eMMC0_DATA0	I/O	MFP6	SD/SDIO0 data line bit 0. eMMC0 data line bit 0.
			UART1_RTS	O	MFP7	UART1 request to Send output pin.
23	52	85	PC.8	I/O	MFP0	General purpose digital I/O pin.
			EBI_DATA8	I/O	MFP1	EBI data bus bit 8.
			VCAP0_DATA0	I	MFP2	Video image interface 0 data 0 pin.

64 Pin	128 Pin	216 Pin	Pin Name	Type	MFP	Description
			NAND_DATA0	I/O	MFP3	NAND Flash data bus bit 0.
			SC1_DAT	I/O	MFP4	Smart Card 1 data pin.
			SPI0_MISO	I/O	MFP5	SPI0 MISO (Master In, Slave Out) pin.
			SD0_DATA1/eMMC0_DATA1	I/O	MFP6	SD/SDIO0 data line bit 1. eMMC0 data line bit 1.
			UART1_CTS	I	MFP7	UART1 clear to Send input pin.
		86	VSS	P	MFP0	Ground pin.
24	53	87	PC.9	I/O	MFP0	General purpose digital I/O pin.
			EBI_DATA9	I/O	MFP1	EBI data bus bit 9.
			VCAP0_DATA1	I	MFP2	Video image interface 0 data 1 pin.
			NAND_DATA1	I/O	MFP3	NAND Flash data bus bit 1.
			SC1_PWR	O	MFP4	Smart Card 1 power pin.
			SD0_DATA2/eMMC0_DATA2	I/O	MFP6	SD/SDIO0 data line bit 2. eMMC0 data line bit 2.
			UART4_TXD	O	MFP7	UART4 data transmitter output pin.
		88	VSS	P	MFP0	Ground pin.
25	54	89	PC.10	I/O	MFP0	General purpose digital I/O pin.
			EBI_DATA10	I/O	MFP1	EBI data bus bit 10.
			VCAP0_DATA2	I	MFP2	Video image interface 0 data 2 pin.
			NAND_DATA2	I/O	MFP3	NAND Flash data bus bit 2.
			SC1_CD	I	MFP4	Smart Card 1 card detect pin.
			SD0_DATA3/eMMC0_DATA3	I/O	MFP6	SD/SDIO0 data line bit 3. eMMC0 data line bit 3.
			UART4_RXD	I	MFP7	UART4 data receiver input pin.
26	55	90	PC.11	I/O	MFP0	General purpose digital I/O pin.
			EBI_DATA11	I/O	MFP1	EBI data bus bit 11.
			VCAP0_DATA3	I	MFP2	Video image interface 0 data 3 pin.
			NAND_DATA3	I/O	MFP3	NAND Flash data bus bit 3.
			SC0_RST	O	MFP4	Smart Card 0 reset pin.
27	56	91	PC.12	I/O	MFP0	General purpose digital I/O pin.
			EBI_DATA12	I/O	MFP1	EBI data bus bit 12.
			VCAP0_DATA4	I	MFP2	Video image interface 0 data 4 pin.
			NAND_DATA4	I/O	MFP3	NAND Flash data bus bit 4.
			SC0_CLK	O	MFP4	Smart Card 0 clock pin.

64 Pin	128 Pin	216 Pin	Pin Name	Type	MFP	Description
			SD0_nCD	I	MFP6	SD/SDIO0 card detect input pin
			UART8_TXD	O	MFP7	UART8 data transmitter output pin.
28	57	92	PC.13	I/O	MFP0	General purpose digital I/O pin.
			EBI_DATA13	I/O	MFP1	EBI data bus bit 13.
			VCAP0_DATA5	I	MFP2	Video image interface 0 data 5 pin.
			NAND_DATA5	I/O	MFP3	NAND Flash data bus bit 5.
			SC0_DAT	I/O	MFP4	Smart Card 0 data pin.
			UART8_RXD	I	MFP7	UART8 data receiver input pin.
29	58	93	PC.14	I/O	MFP0	General purpose digital I/O pin.
			EBI_DATA14	I/O	MFP1	EBI data bus bit 14.
			VCAP0_DATA6	I	MFP2	Video image interface 0 data 6 pin.
			NAND_DATA6	I/O	MFP3	NAND Flash data bus bit 6.
			SC0_PWR	O	MFP4	Smart Card 0 power pin.
			SPI0_MOSI	I/O	MFP5	SPI0 MOSI (Master Out, Slave In) pin.
			UART8_RTS	O	MFP7	UART8 request to Send output pin.
30	59	94	PC.15	I/O	MFP0	General purpose digital I/O pin.
			EBI_DATA15	I/O	MFP1	EBI data bus bit 15.
			VCAP0_DATA7	I	MFP2	Video image interface 0 data 7 pin.
			NAND_DATA7	I/O	MFP3	NAND Flash data bus bit 7.
			SC0_CD	I	MFP4	Smart Card 0 card detect pin.
			UART8_CTS	I	MFP7	UART8 clear to Send input pin.
31		95	VDD12	P	MFP0	Power supply for internal digital circuit.
		96	VSS	P	MFP0	Ground pin.
		97	VSS	P	MFP0	Ground pin.
		98	NC			
32	60	99	VDD33	P	MFP0	Power supply for I/O ports.
		100	NC			
		101	NC			
		102	NC			
		103	NC			
		104	VSS	P	MFP0	Ground pin.
		105	NC			
		106	NC			

64 Pin	128 Pin	216 Pin	Pin Name	Type	MFP	Description
		107	NC			
		108	NC			
		109	NC			
		110	VSS	P	MFP0	Ground pin.
		111	PD.0	I/O	MFP0	General purpose digital I/O pin.
			QSPI0_SS1	I/O	MFP1	Quad SPI0 slave select 1 pin.
			UART5_TXD	O	MFP2	UART5 data transmitter output pin.
			TM1_TGL	I/O	MFP3	Timer1 event counter input/toggle output pin.
			EINT2	I	MFP4	External interrupt 2 input pin.
		112	PD.1	I/O	MFP0	General purpose digital I/O pin.
			SPI0_SS1	I/O	MFP1	SPI0 slave select 1 pin.
			UART5_RXD	I	MFP2	UART5 data receiver input pin.
			TM1_EXT	I/O	MFP3	Timer1 external capture input/toggle output pin.
			EINT3	I	MFP4	External interrupt 3 input pin.
		113	PB.9	I/O	MFP0	General purpose digital I/O pin.
			UART3_TXD	O	MFP1	UART3 data transmitter output pin.
			PWM13	O	MFP2	PWM13 counter synchronous trigger output pin.
			TM0_TGL	I/O	MFP3	Timer0 event counter input/toggle output pin.
			USBHL0_DM	A	MFP4	USB 1.1 Host Lite 0 differential signal D-.
			SPI1_SS0	I/O	MFP5	SPI1 slave select 0 pin.
		114	PB.10	I/O	MFP0	General purpose digital I/O pin.
			UART3_RXD	I	MFP1	UART3 data receiver input pin.
			PWM12	O	MFP2	PWM12 counter synchronous trigger output pin.
			TM0_EXT	I/O	MFP3	Timer0 external capture input/toggle output pin.
			USBHL0_DP	A	MFP4	USB 1.1 Host Lite 0 differential signal D+.
			SPI1_CLK	I/O	MFP5	SPI1 serial clock pin.
		115	PB.11	I/O	MFP0	General purpose digital I/O pin.
			UART3_RTS	O	MFP1	UART3 request to Send output pin.
			PWM11	O	MFP2	PWM11 counter synchronous trigger output pin.
			TM2_EXT	I/O	MFP3	Timer2 external capture input/toggle output pin.
			USBHL5_DM	A	MFP4	USB 1.1 Host Lite 5 differential signal D-.
			SPI1_MOSI	I/O	MFP5	SPI1 MOSI (Master Out, Slave In) pin.
		116	PB.12	I/O	MFP0	General purpose digital I/O pin.

64 Pin	128 Pin	216 Pin	Pin Name	Type	MFP	Description
			UART3_CTS	I	MFP1	UART3 clear to Send input pin.
			PWM10	O	MFP2	PWM10 counter synchronous trigger output pin.
			TM2_TGL	I/O	MFP3	Timer2 event counter input/toggle output pin.
			USBHL5_DP	A	MFP4	USB 1.1 Host Lite 5 differential signal D+.
			SPI1_MISO	I/O	MFP5	SPI1 MISO (Master In, Slave Out) pin.
33	61	117	PD.2	I/O	MFP0	General purpose digital I/O pin.
			QSPI0_SS0	I/O	MFP1	Quad SPI0 slave select 0 pin.
			UART3_TXD	O	MFP2	UART3 data transmitter output pin.
			TM4_EXT	I/O	MFP3	Timer4 external capture input/toggle output pin.
34	62	118	PD.3	I/O	MFP0	General purpose digital I/O pin.
			QSPI0_CLK	I/O	MFP1	Quad SPI0 serial clock pin.
			UART3_RXD	I	MFP2	UART3 data receiver input pin.
			TM4_TGL	I/O	MFP3	Timer4 event counter input/toggle output pin.
35	63	119	PD.4	I/O	MFP0	General purpose digital I/O pin.
			QSPI0_MOSI0	I/O	MFP1	Quad SPI0 MOSI0 (Master Out, Slave In) pin. Data 0 of quad mode. (Booting)
			UART3_RTS	O	MFP2	UART3 request to Send output pin.
			TM5_EXT	I/O	MFP3	Timer5 external capture input/toggle output pin.
	64		VSS	P	MFP0	Ground pin.
36	65	120	PD.5	I/O	MFP0	General purpose digital I/O pin.
			QSPI0_MISO0	I/O	MFP1	Quad SPI0 MISO0 (Master In, Slave Out) pin. Data 1 of quad mode. (Booting)
			UART3_CTS	I	MFP2	UART3 clear to Send input pin.
			TM5_TGL	I/O	MFP3	Timer5 event counter input/toggle output pin.
	66	121	PD.6	I/O	MFP0	General purpose digital I/O pin.
			QSPI0_MOSI1	I/O	MFP1	Quad SPI0 MOSI1 (Master Out, Slave In) pin. Data 2 of quad mode. (Booting)
			UART2_TXD	O	MFP2	UART2 data transmitter output pin.
			TM0_ECNT	I/O	MFP3	Timer0 event counter input/toggle output pin.
			CAN0_RXD	I	MFP4	CAN0 bus receiver input.
	67	122	PD.7	I/O	MFP0	General purpose digital I/O pin.
			QSPI0_MISO1	I/O	MFP1	Quad SPI0 MISO1 (Master In, Slave Out) pin. Data 3 of quad mode. (Booting)
			UART2_RXD	I	MFP2	UART2 data receiver input pin.
			TM1_ECNT	I/O	MFP3	Timer1 event counter input/toggle output pin.

64 Pin	128 Pin	216 Pin	Pin Name	Type	MFP	Description
			CAN0_TXD	O	MFP4	CAN0 bus transmitter output.
	68	123	PD.8	I/O	MFP0	General purpose digital I/O pin.
			SPI0_SS0	I/O	MFP1	SPI0 slave select 0 pin.
			UART6_CTS	I	MFP2	UART6 clear to Send input pin.
			TM2_ECNT	I/O	MFP3	Timer2 event counter input/toggle output pin.
	69	124	PD.9	I/O	MFP0	General purpose digital I/O pin.
			SPI0_CLK	I/O	MFP1	SPI0 serial clock pin.
			UART6_RTS	O	MFP2	UART6 request to Send output pin.
			TM3_ECNT	I/O	MFP3	Timer3 event counter input/toggle output pin.
	70	125	PD.10	I/O	MFP0	General purpose digital I/O pin.
			SPI0_MOSI	I/O	MFP1	SPI0 MOSI (Master Out, Slave In) pin.
			UART6_TXD	O	MFP2	UART6 data transmitter output pin.
			TM4_ECNT	I/O	MFP3	Timer4 event counter input/toggle output pin.
	71	126	PD.11	I/O	MFP0	General purpose digital I/O pin.
			SPI0_MISO	I/O	MFP1	SPI0 MISO (Master In, Slave Out) pin.
			UART6_RXD	I	MFP2	UART6 data receiver input pin.
			TM5_ECNT	I/O	MFP3	Timer5 event counter input/toggle output pin.
	72	127	PD.12	I/O	MFP0	General purpose digital I/O pin.
			UART4_TXD	O	MFP1	UART4 data transmitter output pin.
			TM2_TGL	I/O	MFP2	Timer2 event counter input/toggle output pin.
			CAN2_RXD	I	MFP4	CAN2 bus receiver input.
			PWM00	O	MFP6	PWM00 counter synchronous trigger output pin.
			EBI_DATA1	I/O	MFP8	EBI data bus bit 1.
		128	VSS	P	MFP0	Ground pin
	73	129	PD.13	I/O	MFP0	General purpose digital I/O pin.
			UART4_RXD	I	MFP1	UART4 data receiver input pin.
			TM2_EXT	I/O	MFP2	Timer2 external capture input/toggle output pin.
			CAN2_TXD	O	MFP4	CAN2 bus transmitter output.
			PWM01	O	MFP6	PWM01 counter synchronous trigger output pin.
			EBI_DATA2	I/O	MFP8	EBI data bus bit 2.
	74	130	PD.14	I/O	MFP0	General purpose digital I/O pin.
			UART4_RTS	O	MFP1	UART4 request to Send output pin.
			TM3_TGL	I/O	MFP2	Timer3 event counter input/toggle output pin.

64 Pin	128 Pin	216 Pin	Pin Name	Type	MFP	Description
			I2C3_SCL	I/O	MFP3	I^{2}C3 clock pin.
			CAN1_RXD	I	MFP4	CAN1 bus receiver input.
			USBHL0_DM	A	MFP5	USB 1.1 Host Lite 0 differential signal D-.
			PWM02	O	MFP6	PWM02 counter synchronous trigger output pin.
			EBI_DATA3	I/O	MFP8	EBI data bus bit 3.
	75	131	PD.15	I/O	MFP0	General purpose digital I/O pin.
			UART4_CTS	I	MFP1	UART4 clear to Send input pin.
			TM3_EXT	I/O	MFP2	Timer3 external capture input/toggle output pin.
			I2C3_SDA	I/O	MFP3	I^{2}C3 data input/output pin.
			CAN1_TXD	O	MFP4	CAN1 bus transmitter output.
			USBHL0_DP	A	MFP5	USB 1.1 Host Lite 0 differential signal D+.
			PWM03	O	MFP6	PWM03 counter synchronous trigger output pin.
			EBI_DATA4	I/O	MFP8	EBI data bus bit 4.
	76	132	PG.11	I/O	MFP0	General purpose digital I/O pin.
			SPI1_SS0	I/O	MFP2	SPI1 slave select 0 pin.
			TM1_TGL	I/O	MFP3	Timer1 event counter input/toggle output pin.
			CAN0_RXD	I	MFP4	CAN0 bus receiver input.
			UART5_CTS	I	MFP5	UART5 clear to Send input pin.
			PWM10	O	MFP6	PWM10 counter synchronous trigger output pin.
			JTAG0_TDO	O	MFP7	JTAG0 data output pin.
	77	133	PG.12	I/O	MFP0	General purpose digital I/O pin.
			SPI1_CLK	I/O	MFP2	SPI1 serial clock pin.
			TM1_EXT	I/O	MFP3	Timer1 external capture input/toggle output pin.
			CAN0_TXD	O	MFP4	CAN0 bus transmitter output.
			UART5_RTS	O	MFP5	UART5 request to Send output pin.
			PWM11	O	MFP6	PWM11 counter synchronous trigger output pin.
			JTAG0_TCK	I	MFP7	JTAG0 clock input pin.
	78	134	PG.13	I/O	MFP0	General purpose digital I/O pin.
			SPI1_MOSI	I/O	MFP2	SPI1 MOSI (Master Out, Slave In) pin.
			CAN1_RXD	I	MFP4	CAN1 bus receiver input.
			UART5_RXD	I	MFP5	UART5 data receiver input pin.
			PWM12	O	MFP6	PWM12 counter synchronous trigger output pin.
			JTAG0_TMS	I	MFP7	JTAG0 test mode selection input pin.

64 Pin	128 Pin	216 Pin	Pin Name	Type	MFP	Description
	79	135	PG.14	I/O	MFP0	General purpose digital I/O pin.
			SPI1_MISO	I/O	MFP2	SPI1 MISO (Master In, Slave Out) pin.
			CAN1_TXD	O	MFP4	CAN1 bus transmitter output.
			UART5_TXD	O	MFP5	UART5 data transmitter output pin.
			PWM13	O	MFP6	PWM13 counter synchronous trigger output pin.
			JTAG0_TDI	I	MFP7	JTAG0 data input pin.
	80	136	PG.15	I/O	MFP0	General purpose digital I/O pin.
			SPI0_SS1	I/O	MFP1	SPI0 slave select 1 pin.
			SPI1_SS1	I/O	MFP2	SPI1 slave select 1 pin.
			EINT3	I	MFP4	External interrupt 3 input pin.
			JTAG0_nTRST	I	MFP7	JTAG0 reset input pin.
37	81	137	nRESET	IU	MFP0	External reset input: active LOW, with an internal pull-up. Set this pin low reset to initial state.
			WDT_nRST	O	MFP1	Watch dog timer reset trigger output.
		138	VDD33	P	MFP0	Power supply for I/O ports.
		139	VSS	P	MFP0	Ground pin.
		140	VSS	P	MFP0	Ground pin.
38	82	141	VDD12	P	MFP0	Power supply for internal digital circuit.
39	83	142	MVDD	P	MFP0	Power supply for Memory ports.
		143	VSS	P	MFP0	Ground pin.
		144	MVDD	P	MFP0	Power supply for Memory ports.
		145	VSS	P	MFP0	Ground pin.
	84	146	MVDD	P	MFP0	Power supply for Memory ports.
		147	VSS	P	MFP0	Ground pin.
		148	MVDD	P	MFP0	Power supply for Memory ports.
		149	VSS	P	MFP0	Ground pin.
		150	VSS	P	MFP0	Ground pin.
		151	VSS	P	MFP0	Ground pin.
40	85	152	VDD33	P	MFP0	Power supply for I/O ports.
41	86	153	PF.0	I/O	MFP0	General purpose digital I/O pin.
			RMII1_RXERR	I	MFP1	RMII1 Receive Data Error input pin.
			SD1_CMD/eMMC1_CMD	I/O	MFP2	SD/SDIO1 command/response pin eMMC1 command/response pin
			TM0_ECNT	I/O	MFP3	Timer0 event counter input/toggle output pin.

64 Pin	128 Pin	216 Pin	Pin Name	Type	MFP	Description
			SC1_RST	O	MFP4	Smart Card 1 reset pin.
			UART7_CTS	I	MFP5	UART7 clear to Send input pin.
			USBHL1_DM	A	MFP6	USB 1.1 Host Lite 1 differential signal D-.
			EBI_DATA5	I/O	MFP8	EBI data bus bit 5.
42	87	154	PF.1	I/O	MFP0	General purpose digital I/O pin.
			RMII1_CRSDV	I	MFP1	RMII1 Carrier Sense/Receive Data input pin.
			SD1_CLK/eMMC1_CLK	O	MFP2	SD/SDIO1 clock output pin eMMC1 clock output pin
			TM1_ECNT	I/O	MFP3	Timer1 event counter input/toggle output pin.
			SC1_CLK	O	MFP4	Smart Card 1 clock pin.
			UART7_RTS	O	MFP5	UART7 request to Send output pin.
			USBHL1_DP	A	MFP6	USB 1.1 Host Lite 1 differential signal D+.
			EBI_DATA6	I/O	MFP8	EBI data bus bit 6.
43	88	155	PF.2	I/O	MFP0	General purpose digital I/O pin.
			RMII1_RXD1	I	MFP1	RMII1 Receive Data bus bit 1.
			SD1_DATA0/eMMC1_DATA0	I/O	MFP2	SD/SDIO1 data line bit 0. eMMC1 data line bit 0.
			TM2_ECNT	I/O	MFP3	Timer2 event counter input/toggle output pin.
			SC1_DAT	I/O	MFP4	Smart Card 1 data pin.
			UART7_RXD	I	MFP5	UART7 data receiver input pin.
			USBHL2_DM	A	MFP6	USB 1.1 Host Lite 2 differential signal D-.
			EBI_DATA7	I/O	MFP8	EBI data bus bit 7.
44	89	156	PF.3	I/O	MFP0	General purpose digital I/O pin.
			RMII1_RXD0	I	MFP1	RMII1 Receive Data bus bit 0.
			SD1_DATA1/eMMC1_DATA1	I/O	MFP2	SD/SDIO1 data line bit 1. eMMC1 data line bit 1.
			TM3_ECNT	I/O	MFP3	Timer3 event counter input/toggle output pin.
			SC1_PWR	O	MFP4	Smart Card 1 power pin.
			UART7_TXD	O	MFP5	UART7 data transmitter output pin.
			USBHL2_DP	A	MFP6	USB 1.1 Host Lite 2 differential signal D+.
			EBI_DATA8	I/O	MFP8	EBI data bus bit 8.
45	90	157	PF.4	I/O	MFP0	General purpose digital I/O pin.
			RMII1_REFCLK	I	MFP1	RMII1 mode clock input pin.
			SD1_DATA2/eMMC1_DATA	I/O	MFP2	SD/SDIO1 data line bit 2.

64 Pin	128 Pin	216 Pin	Pin Name	Type	MFP	Description
			2			eMMC1 data line bit 2.
			TM4_ECNT	I/O	MFP3	Timer4 event counter input/toggle output pin.
			SC1_CD	I	MFP4	Smart Card 1 card detect pin.
			UART3_CTS	I	MFP5	UART3 clear to Send input pin.
			USBHL3_DM	A	MFP6	USB 1.1 Host Lite 3 differential signal D-.
			EBI_DATA9	I/O	MFP8	EBI data bus bit 9.
46	91	158	PF.5	I/O	MFP0	General purpose digital I/O pin.
			RMII1_TXEN	O	MFP1	RMII1 Transmit Enable output pin.
			SD1_DATA3/eMMC1_DATA3	I/O	MFP2	SD/SDIO1 data line bit 3. eMMC1 data line bit 3.
			TM5_ECNT	I/O	MFP3	Timer5 event counter input/toggle output pin.
			PWM00	O	MFP4	PWM00 counter synchronous trigger output pin.
			UART3_RTS	O	MFP5	UART3 request to Send output pin.
			USBHL3_DP	A	MFP6	USB 1.1 Host Lite 3 differential signal D+.
			EBI_DATA10	I/O	MFP8	EBI data bus bit 10.
47	92	159	PF.6	I/O	MFP0	General purpose digital I/O pin.
			RMII1_TXD1	O	MFP1	RMII1 Transmit Data bus bit 1.
			SD1_nCD	I	MFP2	SD/SDIO1 card detect input pin
			TM4_EXT	I/O	MFP3	Timer4 external capture input/toggle output pin.
			PWM01	O	MFP4	PWM01 counter synchronous trigger output pin.
			UART3_RXD	I	MFP5	UART3 data receiver input pin.
			USBHL4_DM	A	MFP6	USB 1.1 Host Lite 4 differential signal D-.
			EBI_DATA11	I/O	MFP8	EBI data bus bit 11.
	93	160	PB.13	I/O	MFP0	General purpose digital I/O pin.
			EINT2	I	MFP2	External interrupt 2 input pin.
			TM4_TGL	I/O	MFP3	Timer4 event counter input/toggle output pin.
			PWM02	O	MFP4	PWM02 counter synchronous trigger output pin.
			UART3_TXD	O	MFP5	UART3 data transmitter output pin.
			USBHL4_DP	A	MFP6	USB 1.1 Host Lite 4 differential signal D+.
			EBI_DATA0	I/O	MFP8	EBI data bus bit 0.
48	94	161	VDD12	P	MFP0	Power supply for internal digital circuit.
		162	VDD12	P	MFP0	Power supply for internal digital circuit.
		163	VSS	P	MFP0	Ground pin.

64 Pin	128 Pin	216 Pin	Pin Name	Type	MFP	Description
	95	164	VSS	P	MFP0	Ground pin.
49	96	165	PF.7	I/O	MFP0	General purpose digital I/O pin.
			RMII1_TXD0	O	MFP1	RMII1 Transmit Data bus bit 0.
			UART1_CTS	I	MFP2	UART1 clear to Send input pin.
			TM5_EXT	I/O	MFP3	Timer5 external capture input/toggle output pin.
			PWM02	O	MFP4	PWM02 counter synchronous trigger output pin.
			UART3_TXD	O	MFP5	UART3 data transmitter output pin.
			USBHL4_DP	A	MFP6	USB 1.1 Host Lite 4 differential signal D+.
			EBI_DATA12	I/O	MFP8	EBI data bus bit 12.
50	97	166	PF.8	I/O	MFP0	General purpose digital I/O pin.
			RMII1_MDIO	I/O	MFP1	RMII1 PHY Management Data pin.
			UART1_RTS	O	MFP2	UART1 request to Send output pin.
			TM1_TGL	I/O	MFP3	Timer1 event counter input/toggle output pin.
			PWM03	O	MFP4	PWM03 counter synchronous trigger output pin.
			USBHL5_DM	A	MFP6	USB 1.1 Host Lite 5 differential signal D-.
			EBI_DATA13	I/O	MFP8	EBI data bus bit 13.
51	98	167	PF.9	I/O	MFP0	General purpose digital I/O pin.
			RMII1_MDC	O	MFP1	RMII1 PHY Management Clock output pin.
			UART1_RXD	I	MFP2	UART1 data receiver input pin.
			TM1_EXT	I/O	MFP3	Timer1 external capture input/toggle output pin.
			PWM10	O	MFP4	PWM10 counter synchronous trigger output pin.
			USBHL5_DP	A	MFP6	USB 1.1 Host Lite 5 differential signal D+.
			EBI_DATA14	I/O	MFP8	EBI data bus bit 14.
		168	VDD12	P	MFP0	Power supply for internal digital circuit.
	99	169	PF.10	I/O	MFP0	General purpose digital I/O pin.
			UART1_TXD	O	MFP2	UART1 data transmitter output pin.
			TM5_TGL	I/O	MFP3	Timer5 event counter input/toggle output pin.
			PWM11	O	MFP4	PWM11 counter synchronous trigger output pin.
			VCAP1_PCLK	I	MFP7	Video image interface 1 pixel clock pin.
			EBI_DATA15	I/O	MFP8	EBI data bus bit 15.
52	100	170	PF.11	I/O	MFP0	General purpose digital I/O pin.
			UART0_RXD	I	MFP1	UART0 data receiver input pin.
		171	VSS	P	MFP0	Ground pin.

64 Pin	128 Pin	216 Pin	Pin Name	Type	MFP	Description
53	101	172	PF.12	I/O	MFP0	General purpose digital I/O pin.
			UART0_TXD	O	MFP1	UART0 data transmitter output pin.
			VSS	P	MFP0	Ground pin.
54	102	173	VDD33	P	MFP0	Power supply for I/O ports.
55	103	174	XT_IN	I	MFP0	External 12 MHz (high speed) crystal input pin.
56	104	175	XT_OUT	O	MFP0	External 12 MHz (high speed) crystal output pin.
		176	VSS	P	MFP0	Ground pin.
	105	177	PE.0	I/O	MFP0	General purpose digital I/O pin.
			RMII0_RXERR	I	MFP1	RMII0 Receive Data Error input pin.
			CAN0_RXD	I	MFP2	CAN0 bus receiver input.
			UART4_CTS	I	MFP5	UART4 clear to Send input pin.
			USBHL1_DM	A	MFP6	USB 1.1 Host Lite 1 differential signal D-.
			VCAP1_HSYNC	I	MFP7	Video image interface 1 horizontal sync. pin.
	106	178	PE.1	I/O	MFP0	General purpose digital I/O pin.
			RMII0_CRSDV	I	MFP1	RMII0 Carrier Sense/Receive Data input pin.
			CAN0_TXD	O	MFP2	CAN0 bus transmitter output.
			UART4_RTS	O	MFP5	UART4 request to Send output pin.
			USBHL1_DP	A	MFP6	USB 1.1 Host Lite 1 differential signal D+.
			VCAP1_VSYNC	I	MFP7	Video image interface 1 vertical sync. pin.
	107	179	PE.2	I/O	MFP0	General purpose digital I/O pin.
			RMII0_RXD1	I	MFP1	RMII0 Receive Data bus bit 1.
			CAN1_RXD	I	MFP2	CAN1 bus receiver input.
			UART4_RXD	I	MFP5	UART4 data receiver input pin.
			USBHL2_DM	A	MFP6	USB 1.1 Host Lite 2 differential signal D-.
			VCAP1_DATA0	I	MFP7	Video image interface 1 data 0 pin.
		180	VDD12	P	MFP0	Power supply for internal digital circuit.
	108	181	PE.3	I/O	MFP0	General purpose digital I/O pin.
			RMII0_RXD0	I	MFP1	RMII0 Receive Data bus bit 0.
			CAN1_TXD	O	MFP2	CAN1 bus transmitter output.
			UART4_TXD	O	MFP5	UART4 data transmitter output pin.
			USBHL2_DP	A	MFP6	USB 1.1 Host Lite 2 differential signal D+.
			VCAP1_DATA1	I	MFP7	Video image interface 1 data 1 pin.
	109	182	PE.4	I/O	MFP0	General purpose digital I/O pin.

64 Pin	128 Pin	216 Pin	Pin Name	Type	MFP	Description
			RMII0_REFCLK	I	MFP1	RMII0 mode clock input pin.
			CAN2_RXD	I	MFP2	CAN2 bus receiver input.
			UART9_CTS	I	MFP5	UART9 clear to Send input pin.
			USBHL3_DM	A	MFP6	USB 1.1 Host Lite 3 differential signal D-.
			VCAP1_DATA2	I	MFP7	Video image interface 1 data 2 pin.
	110	183	PE.5	I/O	MFP0	General purpose digital I/O pin.
			RMII0_TXEN	O	MFP1	RMII0 Transmit Enable output pin.
			CAN2_TXD	O	MFP2	CAN2 bus transmitter output.
			UART9_RTS	O	MFP5	UART9 request to Send output pin.
			USBHL3_DP	A	MFP6	USB 1.1 Host Lite 3 differential signal D+.
			VCAP1_DATA3	I	MFP7	Video image interface 1 data 3 pin.
	111	184	PE.6	I/O	MFP0	General purpose digital I/O pin.
			RMII0_TXD1	O	MFP1	RMII0 Transmit Data bus bit 1.
			CAN3_RXD	I	MFP2	CAN3 bus receiver input.
			UART9_RXD	I	MFP5	UART9 data receiver input pin.
			USBHL4_DM	A	MFP6	USB 1.1 Host Lite 4 differential signal D-.
			VCAP1_DATA4	I	MFP7	Video image interface 1 data 4 pin.
	112	185	PE.7	I/O	MFP0	General purpose digital I/O pin.
			RMII0_TXD0	O	MFP1	RMII0 Transmit Data bus bit 0.
			CAN3_TXD	O	MFP2	CAN3 bus transmitter output.
			UART9_TXD	O	MFP5	UART9 data transmitter output pin.
			USBHL4_DP	A	MFP6	USB 1.1 Host Lite 4 differential signal D+.
			VCAP1_DATA5	I	MFP7	Video image interface 1 data 5 pin.
		186	VDD33	P	MFP0	Power supply for I/O ports.
	113	187	PE.8	I/O	MFP0	General purpose digital I/O pin.
			RMII0_MDIO	I/O	MFP1	RMII0 PHY Management Data pin.
			UART6_RXD	I	MFP5	UART6 data receiver input pin.
			USBHL5_DM	A	MFP6	USB 1.1 Host Lite 5 differential signal D-.
			VCAP1_DATA6	I	MFP7	Video image interface 1 data 6 pin.
	114	188	PE.9	I/O	MFP0	General purpose digital I/O pin.
			RMII0_MDC	O	MFP1	RMII0 PHY Management Clock output pin.
			UART6_TXD	O	MFP5	UART6 data transmitter output pin.
			USBHL5_DP	A	MFP6	USB 1.1 Host Lite 5 differential signal D+.

64 Pin	128 Pin	216 Pin	Pin Name	Type	MFP	Description
			VCAP1_DATA7	I	MFP7	Video image interface 1 data 7 pin.
	115	189	PE.10	I/O	MFP0	General purpose digital I/O pin.
			USB_OVC	I	MFP1	USB host bus power over voltage detector.
			CAN3_RXD	I	MFP2	CAN3 bus receiver input.
			UART9_RXD	I	MFP3	UART9 data receiver input pin.
			PWM12	O	MFP4	PWM12 counter synchronous trigger output pin.
			EINT2	I	MFP5	External interrupt 2 input pin.
			I2C0_SDA	I/O	MFP6	I^{2}C0 data input/output pin.
			VCAP1_FIELD	I	MFP7	Video image interface 1 frame sync. pin.
57	116	190	PE.11	I/O	MFP0	General purpose digital I/O pin.
			USB0_VBUSVLD	I	MFP1	USB0 external VBUS regulator status pin.
	117	191	PE.12	I/O	MFP0	General purpose digital I/O pin.
			USBH_PWREN	O	MFP1	HSUSB external VBUS regulator enable pin.
			CAN3_TXD	O	MFP2	CAN3 bus transmitter output.
			UART9_TXD	O	MFP3	UART9 data transmitter output pin.
			PWM13	O	MFP4	PWM13 counter synchronous trigger output pin.
			EINT3	I	MFP5	External interrupt 3 input pin.
			I2C0_SCL	I/O	MFP6	I^{2}C0 clock pin.
			VCAP1_CLKO	O	MFP7	Video image interface 1 sensor clock pin.
		192	VSS	P	MFP0	Ground pin.
58	118	193	VDD12	P	MFP0	Power supply for internal digital circuit.
		194	VSS	P	MFP0	Ground pin.
		195	VUSB1_VDD12	P	MFP0	Power supply for USB1 V_{DD}12
		196	VUSB1_VDD12	P	MFP0	Power supply for USB1 V_{DD}12
		197	VUSB1_VSS	P	MFP0	Ground pin for USB1.
		198	NC			
		199	NC			
		200	VUSB1_VSS	P	MFP0	Ground pin for USB1.
	119	201	USB1_DM	A	MFP0	USB1 differential signal D-.
	120	202	USB1_DP	A	MFP0	USB1 differential signal D+.
59	121	203	VUSB1_VDD33	P	MFP0	Power supply for USB1 V_{DD}33
		204	VUSB1_VDD33	P	MFP0	Power supply for USB1 V_{DD}33
	122	205	USB1_REXT	A	MFP0	USB1 module reference resister

64 Pin	128 Pin	216 Pin	Pin Name	Type	MFP	Description
60	123	206	VUSB0_VDD12	P	MFP0	Power supply for USB0 V_{DD}12
		207	VUSB0_VDD12	P	MFP0	Power supply for USB0 V_{DD}12
		208	VUSB0_VSS	P	MFP0	Ground pin for USB0.
		209	NC			
		210	NC			
		211	VUSB0_VSS	P	MFP0	Ground pin for USB0.
61	124	212	USB0_DM	A	MFP0	USB0 differential signal D-.
62	125	213	USB0_DP	A	MFP0	USB0 differential signal D+.
63	126	214	VUSB0_VDD33	P	MFP0	Power supply for USB0 V_{DD}33
		215	VUSB0_VDD33	P	MFP0	Power supply for USB0 V_{DD}33
64	127	216	USB0_REXT	A	MFP0	USB0 module reference resister
	128		VSS	P	MFP0	Ground Pin.

4.2.2 NUC980 Multi-function Summary Table

Group	Pin Name	GPIO	MFP	Type	Description
ADC	ADC_AIN0	PB.0	MFP8	A	ADC channel 0 analog input.
	ADC_AIN1	PB.1	MFP8	A	ADC channel 1 analog input.
	ADC_AIN2	PB.2	MFP8	A	ADC channel 2 analog input.
	ADC_AIN3	PB.3	MFP8	A	ADC channel 3 analog input.
	ADC_AIN4	PB.4	MFP8	A	ADC channel 4 analog input.
	ADC_AIN5	PB.5	MFP8	A	ADC channel 5 analog input.
	ADC_AIN6	PB.6	MFP8	A	ADC channel 6 analog input.
	ADC_AIN7	PB.7	MFP8	A	ADC channel 7 analog input.
CAN0	CAN0_RXD	PC.3	MFP7	I	CAN0 bus receiver input.
		PD.6	MFP4	I	
		PG.11	MFP4	I	
		PE.0	MFP2	I	
	CAN0_TXD	PC.4	MFP7	O	CAN0 bus transmitter output.
		PD.7	MFP4	O	
		PG.12	MFP4	O	
		PE.1	MFP2	O	
CAN1	CAN1_RXD	PA.13	MFP5	I	CAN1 bus receiver input.
		PD.14	MFP4	I	
		PG.13	MFP4	I	
		PE.2	MFP2	I	
	CAN1_TXD	PA.14	MFP5	O	CAN1 bus transmitter output.
		PD.15	MFP4	O	
		PG.14	MFP4	O	
		PE.3	MFP2	O	
CAN2	CAN2_RXD	PA.15	MFP5	I	CAN2 bus receiver input.
		PB.1	MFP4	I	
		PB.8	MFP3	I	
		PD.12	MFP4	I	
		PE.4	MFP2	I	
	CAN2_TXD	PG.10	MFP5	O	CAN2 bus transmitter output.
		PB.3	MFP4	O	
		PC.0	MFP3	O	

Group	Pin Name	GPIO	MFP	Type	Description
		PD.13	MFP4	O	
		PE.5	MFP2	O	
CAN3	CAN3_RXD	PA.0	MFP7	I	CAN3 bus receiver input.
		PE.6	MFP2	I	
		PE.10	MFP2	I	
	CAN3_TXD	PA.1	MFP7	O	CAN3 bus transmitter output.
		PE.7	MFP2	O	
		PE.12	MFP2	O	
CFG.0	CFG.0_PwrOnSet0	PG.0	-	IU	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
CFG.1	CFG.1_PwrOnSet1	PG.1	-	IU	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
CFG.2	CFG.2_PwrOnSet2	PG.2	-	IU	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
CFG.3	CFG.3_PwrOnSet3	PG.3	-	IU	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
CFG.4	CFG.4_PwrOnSet4	PG.4	-	IU	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
CFG.5	CFG.5_PwrOnSet5	PG.5	-	IU	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
CFG.6	CFG.6_PwrOnSet6	PG.6	-	IU	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
CFG.7	CFG.7_PwrOnSet7	PG.7	-	IU	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
CFG.8	CFG.8_PwrOnSet8	PG.8	-	IU	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
CFG.9	CFG.9_PwrOnSet9	PG.9	-	IU	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
CLK	CLK_OUT	PG.0	MFP3	O	Internal clock selection output pin.
EBI	EBI_ADDR0	PG.0	MFP1	O	EBI address bus bit 0.
	EBI_ADDR1	PG.1	MFP1	O	EBI address bus bit 1.
	EBI_ADDR2	PG.2	MFP1	O	EBI address bus bit 2.
		PB.2	MFP1	O	

Group	Pin Name	GPIO	MFP	Type	Description
	EBI_ADDR3	PG.3	MFP1	O	EBI address bus bit 3.
	EBI_ADDR4	PG.6	MFP1	O	EBI address bus bit 4.
	EBI_ADDR5	PG.7	MFP1	O	EBI address bus bit 5.
	EBI_ADDR6	PG.8	MFP1	O	EBI address bus bit 6.
	EBI_ADDR7	PG.9	MFP1	O	EBI address bus bit 7.
	EBI_ADDR8	PA.12	MFP1	O	EBI address bus bit 8.
	EBI_ADDR9	PA.11	MFP1	O	EBI address bus bit 9.
	EBI_ADDR10	PA.10	MFP1	O	EBI address bus bit 10.
	EBI_ADDR11	PB.8	MFP1	O	EBI address bus bit 11.
	EBI_ADDR12	PG.5	MFP1	O	EBI address bus bit 12.
		PB.0	MFP1	O	
	EBI_ADDR13	PA.13	MFP1	O	EBI address bus bit 13.
		PB.6	MFP1	O	
	EBI_ADDR14	PA.14	MFP1	O	EBI address bus bit 14.
		PB.4	MFP1	O	
	EBI_ADDR15	PB.7	MFP1	O	EBI address bus bit 15.
	EBI_ADDR16	PB.5	MFP1	O	EBI address bus bit 16.
	EBI_ADDR17	PB.1	MFP1	O	EBI address bus bit 17.
	EBI_ADDR18	PG.4	MFP1	O	EBI address bus bit 18.
		PB.3	MFP1	O	
	EBI_ADDR19	PA.15	MFP1	O	EBI address bus bit 19.
	EBI_DATA0	PG.10	MFP1	I/O	EBI data bus bit 0.
		PC.0	MFP1	I/O	
		PB.13	MFP8	I/O	
	EBI_DATA1	PC.1	MFP1	I/O	EBI data bus bit 1.
		PD.12	MFP8	I/O	
	EBI_DATA2	PC.2	MFP1	I/O	EBI data bus bit 2.
		PD.13	MFP8	I/O	
	EBI_DATA3	PC.3	MFP1	I/O	EBI data bus bit 3.
		PD.14	MFP8	I/O	
	EBI_DATA4	PC.4	MFP1	I/O	EBI data bus bit 4.
		PD.15	MFP8	I/O	
	EBI_DATA5	PC.5	MFP1	I/O	EBI data bus bit 5.

Group	Pin Name	GPIO	MFP	Type	Description
		PF.0	MFP8	I/O	
	EBI_DATA6	PC.6	MFP1	I/O	EBI data bus bit 6.
		PF.1	MFP8	I/O	
	EBI_DATA7	PC.7	MFP1	I/O	EBI data bus bit 7.
		PF.2	MFP8	I/O	
	EBI_DATA8	PC.8	MFP1	I/O	EBI data bus bit 8.
		PF.3	MFP8	I/O	
	EBI_DATA9	PC.9	MFP1	I/O	EBI data bus bit 9.
		PF.4	MFP8	I/O	
	EBI_DATA10	PC.10	MFP1	I/O	EBI data bus bit 10.
		PF.5	MFP8	I/O	
	EBI_DATA11	PC.11	MFP1	I/O	EBI data bus bit 11.
		PF.6	MFP8	I/O	
	EBI_DATA12	PC.12	MFP1	I/O	EBI data bus bit 12.
		PF.7	MFP8	I/O	
	EBI_DATA13	PC.13	MFP1	I/O	EBI data bus bit 13.
		PF.8	MFP8	I/O	
	EBI_DATA14	PC.14	MFP1	I/O	EBI data bus bit 14.
		PF.9	MFP8	I/O	
	EBI_DATA15	PC.15	MFP1	I/O	EBI data bus bit 15.
		PF.10	MFP8	I/O	
	EBI_MCLK	PA.1	MFP2	O	EBI external clock output pin.
	EBI_nCS0	PA.9	MFP1	O	EBI chip select 0 output pin.
	EBI_nCS1	PA.6	MFP1	O	EBI chip select 1 output pin.
	EBI_nCS2	PA.1	MFP1	O	EBI chip select 2 output pin.
	EBI_nRE	PA.8	MFP1	O	EBI read enable output pin.
	EBI_nWE	PA.7	MFP1	O	EBI write enable output pin.
EINT0	EINT0	PA.0	MFP5	I	External interrupt 0 input pin.
		PA.13	MFP8	I	
EINT1	EINT1	PA.1	MFP5	I	External interrupt 1 input pin.
		PA.14	MFP8	I	
EINT2	EINT2	PB.3	MFP3	I	External interrupt 2 input pin.
		PD.0	MFP4	I	

Group	Pin Name	GPIO	MFP	Type	Description
		PB.13	MFP2	I	
		PE.10	MFP5	I	
EINT3	EINT3	PD.1	MFP4	I	External interrupt 3 input pin.
		PG.15	MFP4	I	
		PE.12	MFP5	I	
I^2C0	I2C0_SCL	PA.1	MFP3	I/O	I^2C0 clock pin.
		PG.10	MFP2	I/O	
		PE.12	MFP6	I/O	
	I2C0_SDA	PA.0	MFP3	I/O	I^2C0 data input/output pin.
		PA.15	MFP2	I/O	
		PE.10	MFP6	I/O	
I^2C1	I2C1_SCL	PA.14	MFP2	I/O	I^2C1 clock pin.
		PB.4	MFP2	I/O	
		PC.3	MFP4	I/O	
	I2C1_SDA	PA.13	MFP2	I/O	I^2C1 data input/output pin.
		PB.6	MFP2	I/O	
		PC.4	MFP4	I/O	
I^2C2	I2C2_SCL	PB.5	MFP2	I/O	I^2C2 clock pin.
		PB.8	MFP2	I/O	
	I2C2_SDA	PB.7	MFP2	I/O	I^2C2 data input/output pin.
		PC.0	MFP2	I/O	
I^2C3	I2C3_SCL	PB.3	MFP2	I/O	I^2C3 clock pin.
		PD.14	MFP3	I/O	
	I2C3_SDA	PB.1	MFP2	I/O	I^2C3 data input/output pin.
		PD.15	MFP3	I/O	
I^2S	I2S_BCLK	PA.3	MFP2	O	I^2S_ bit clock output pin.
		PG.10	MFP8	O	
		PB.4	MFP3	O	
	I2S_DI	PA.4	MFP2	I	I^2S_ data input pin.
		PB.7	MFP3	I	
	I2S_DO	PA.5	MFP2	O	I^2S_ data output pin.
		PB.5	MFP3	O	
	I2S_LRCK	PA.2	MFP2	O	I^2S_ left right channel clock output pin.

Group	Pin Name	GPIO	MFP	Type	Description
		PA.15	MFP8	O	
		PB.6	MFP3	O	
	I2S_MCLK	PA.6	MFP2	O	I^{2}S_ master clock output pin.
		PB.1	MFP3	O	
JTAG0	JTAG0_TCK	PG.12	MFP7	I	JTAG0 clock input pin.
	JTAG0_TDI	PG.14	MFP7	I	JTAG0 data input pin.
	JTAG0_TDO	PG.11	MFP7	O	JTAG0 data output pin.
	JTAG0_TMS	PG.13	MFP7	I	JTAG0 test mode selection input pin.
	JTAG0_nTRST	PG.15	MFP7	I	JTAG0 reset input pin.
JTAG1	JTAG1_TCK	PA.3	MFP4	I	JTAG1 clock input pin.
	JTAG1_TDI	PA.5	MFP4	I	JTAG1 data input pin.
	JTAG1_TDO	PA.2	MFP4	O	JTAG1 data output pin.
	JTAG1_TMS	PA.4	MFP4	I	JTAG1 test mode selection input pin.
	JTAG1_nTRST	PA.6	MFP4	I	JTAG1 reset input pin.
NAND	NAND_ALE	PC.3	MFP3	O	NAND Flash address latch enable.
	NAND_CLE	PC.4	MFP3	O	NAND Flash command latch enable.
	NAND_DATA0	PC.8	MFP3	I/O	NAND Flash data bus bit 0.
	NAND_DATA1	PC.9	MFP3	I/O	NAND Flash data bus bit 1.
	NAND_DATA2	PC.10	MFP3	I/O	NAND Flash data bus bit 2.
	NAND_DATA3	PC.11	MFP3	I/O	NAND Flash data bus bit 3.
	NAND_DATA4	PC.12	MFP3	I/O	NAND Flash data bus bit 4.
	NAND_DATA5	PC.13	MFP3	I/O	NAND Flash data bus bit 5.
	NAND_DATA6	PC.14	MFP3	I/O	NAND Flash data bus bit 6.
	NAND_DATA7	PC.15	MFP3	I/O	NAND Flash data bus bit 7.
	NAND_RDY0	PC.7	MFP3	I	NAND Flash ready/busy pin.
	NAND_nCS0	PC.1	MFP3	O	NAND Flash chip enable input.
	NAND_nRE	PC.6	MFP3	O	NAND Flash read enable.
	NAND_nWE	PC.5	MFP3	O	NAND Flash write enable.
	NAND_nWP	PC.2	MFP3	O	NAND Flash write protect input.
PWM0	PWM00	PG.10	MFP7	O	PWM00 counter synchronous trigger output pin.
		PG.0	MFP6	O	
		PD.12	MFP6	O	
		PF.5	MFP4	O	

Group	Pin Name	GPIO	MFP	Type	Description
	PWM01	PA.15	MFP7	O	PWM01 counter synchronous trigger output pin.
		PG.1	MFP6	O	
		PD.13	MFP6	O	
		PF.6	MFP4	O	
	PWM02	PA.14	MFP7	O	PWM02 counter synchronous trigger output pin.
		PG.2	MFP6	O	
		PD.14	MFP6	O	
		PB.13	MFP4	O	
		PF.7	MFP4	O	
	PWM03	PA.13	MFP7	O	PWM03 counter synchronous trigger output pin.
		PG.3	MFP6	O	
		PD.15	MFP6	O	
		PF.8	MFP4	O	
PWM1	PWM10	PG.6	MFP6	O	PWM10 counter synchronous trigger output pin.
		PB.12	MFP2	O	
		PG.11	MFP6	O	
		PF.9	MFP4	O	
	PWM11	PG.7	MFP6	O	PWM11 counter synchronous trigger output pin.
		PB.11	MFP2	O	
		PG.12	MFP6	O	
		PF.10	MFP4	O	
	PWM12	PG.8	MFP6	O	PWM12 counter synchronous trigger output pin.
		PB.10	MFP2	O	
		PG.13	MFP6	O	
		PE.10	MFP4	O	
	PWM13	PG.9	MFP6	O	PWM13 counter synchronous trigger output pin.
		PB.9	MFP2	O	
		PG.14	MFP6	O	
		PE.12	MFP4	O	
QSPI0	QSPI0_CLK	PD.3	MFP1	I/O	Quad SPI0 serial clock pin.
	QSPI0_MISO0	PD.5	MFP1	I/O	Quad SPI0 MISO0 (Master In, Slave Out) pin.
	QSPI0_MISO1	PD.7	MFP1	I/O	Quad SPI0 MISO1 (Master In, Slave Out) pin.
	QSPI0_MOSI0	PD.4	MFP1	I/O	Quad SPI0 MOSI0 (Master Out, Slave In) pin.

Group	Pin Name	GPIO	MFP	Type	Description
	QSPI0_MOSI1	PD.6	MFP1	I/O	Quad SPI0 MOSI1 (Master Out, Slave In) pin.
	QSPI0_SS0	PD.2	MFP1	I/O	Quad SPI0 slave select 0 pin.
	QSPI0_SS1	PA.0	MFP1	I/O	Quad SPI0 slave select 1 pin.
		PD.0	MFP1	I/O	
RMII0	RMII0_CRSDV	PE.1	MFP1	I	RMII0 Carrier Sense/Receive Data input pin.
	RMII0_MDC	PE.9	MFP1	O	RMII0 PHY Management Clock output pin.
	RMII0_MDIO	PE.8	MFP1	I/O	RMII0 PHY Management Data pin.
	RMII0_REFCLK	PE.4	MFP1	I	RMII0 mode clock input pin.
	RMII0_RXD0	PE.3	MFP1	I	RMII0 Receive Data bus bit 0.
	RMII0_RXD1	PE.2	MFP1	I	RMII0 Receive Data bus bit 1.
	RMII0_RXERR	PE.0	MFP1	I	RMII0 Receive Data Error input pin.
	RMII0_TXD0	PE.7	MFP1	O	RMII0 Transmit Data bus bit 0.
	RMII0_TXD1	PE.6	MFP1	O	RMII0 Transmit Data bus bit 1.
	RMII0_TXEN	PE.5	MFP1	O	RMII0 Transmit Enable output pin.
RMII1	RMII1_CRSDV	PF.1	MFP1	I	RMII1 Carrier Sense/Receive Data input pin.
	RMII1_MDC	PF.9	MFP1	O	RMII1 PHY Management Clock output pin.
	RMII1_MDIO	PF.8	MFP1	I/O	RMII1 PHY Management Data pin.
	RMII1_REFCLK	PF.4	MFP1	I	RMII1 mode clock input pin.
	RMII1_RXD0	PF.3	MFP1	I	RMII1 Receive Data bus bit 0.
	RMII1_RXD1	PF.2	MFP1	I	RMII1 Receive Data bus bit 1.
	RMII1_RXERR	PF.0	MFP1	I	RMII1 Receive Data Error input pin.
	RMII1_TXD0	PF.7	MFP1	O	RMII1 Transmit Data bus bit 0.
	RMII1_TXD1	PF.6	MFP1	O	RMII1 Transmit Data bus bit 1.
	RMII1_TXEN	PF.5	MFP1	O	RMII1 Transmit Enable output pin.
SC0	SC0_CD	PA.2	MFP3	I	Smart Card 0 card detect pin.
		PC.15	MFP4	I	
	SC0_CLK	PA.5	MFP3	O	Smart Card 0 clock pin.
		PC.12	MFP4	O	
	SC0_DAT	PA.4	MFP3	I/O	Smart Card 0 data pin.
		PC.13	MFP4	I/O	
	SC0_PWR	PA.3	MFP3	O	Smart Card 0 power pin.
		PC.14	MFP4	O	
	SC0_RST	PA.6	MFP3	O	Smart Card 0 reset pin.

Group	Pin Name	GPIO	MFP	Type	Description
		PC.11	MFP4	O	
SC1	SC1_CD	PC.10	MFP4	I	Smart Card 1 card detect pin.
		PF.4	MFP4	I	
	SC1_CLK	PC.7	MFP4	O	Smart Card 1 clock pin.
		PF.1	MFP4	O	
	SC1_DAT	PC.8	MFP4	I/O	Smart Card 1 data pin.
		PF.2	MFP4	I/O	
	SC1_PWR	PC.9	MFP4	O	Smart Card 1 power pin.
		PF.3	MFP4	O	
	SC1_RST	PC.6	MFP4	O	Smart Card 1 reset pin.
		PF.0	MFP4	O	
SD0	SD0_CLK	PC.6	MFP6	O	SD/SDIO0 clock output pin
	SD0_CMD	PC.5	MFP6	I/O	SD/SDIO0 command/response pin
	SD0_DATA0	PC.7	MFP6	I/O	SD/SDIO0 data line bit 0.
	SD0_DATA1	PC.8	MFP6	I/O	SD/SDIO0 data line bit 1.
	SD0_DATA2	PC.9	MFP6	I/O	SD/SDIO0 data line bit 2.
	SD0_DATA3	PC.10	MFP6	I/O	SD/SDIO0 data line bit 3.
	SD0_nCD	PB.8	MFP6	I	SD/SDIO0 card detect input pin
		PC.12	MFP6	I	
SD1	SD1_CLK	PF.1	MFP2	O	SD/SDIO1 clock output pin
	SD1_CMD	PF.0	MFP2	I/O	SD/SDIO1 command/response pin
	SD1_DATA0	PF.2	MFP2	I/O	SD/SDIO1 data line bit 0.
	SD1_DATA1	PF.3	MFP2	I/O	SD/SDIO1 data line bit 1.
	SD1_DATA2	PF.4	MFP2	I/O	SD/SDIO1 data line bit 2.
	SD1_DATA3	PF.5	MFP2	I/O	SD/SDIO1 data line bit 3.
	SD1_nCD	PF.6	MFP2	I	SD/SDIO1 card detect input pin
SPI0	SPI0_CLK	PC.6	MFP5	I/O	SPI0 serial clock pin.
		PD.9	MFP1	I/O	
	SPI0_MISO	PC.8	MFP5	I/O	SPI0 MISO (Master In, Slave Out) pin.
		PD.11	MFP1	I/O	
	SPI0_MOSI	PC.4	MFP6	I/O	SPI0 MOSI (Master Out, Slave In) pin.
		PC.7	MFP5	I/O	
		PC.14	MFP5	I/O	

Group	Pin Name	GPIO	MFP	Type	Description
		PD.10	MFP1	I/O	
	SPI0_SS0	PC.5	MFP5	I/O	SPI0 slave select 0 pin.
		PD.8	MFP1	I/O	
	SPI0_SS1	PB.3	MFP6	I/O	SPI0 slave select 1 pin.
		PC.0	MFP5	I/O	
		PD.1	MFP1	I/O	
		PG.15	MFP1	I/O	
SPI1	SPI1_CLK	PG.10	MFP6	I/O	SPI1 serial clock pin.
		PB.4	MFP6	I/O	
		PB.10	MFP5	I/O	
		PG.12	MFP2	I/O	
	SPI1_MISO	PB.5	MFP6	I/O	SPI1 MISO (Master In, Slave Out) pin.
		PB.12	MFP5	I/O	
		PG.14	MFP2	I/O	
	SPI1_MOSI	PB.7	MFP6	I/O	SPI1 MOSI (Master Out, Slave In) pin.
		PB.11	MFP5	I/O	
		PG.13	MFP2	I/O	
	SPI1_SS0	PA.15	MFP6	I/O	SPI1 slave select 0 pin.
		PB.6	MFP6	I/O	
		PB.9	MFP5	I/O	
		PG.11	MFP2	I/O	
	SPI1_SS1	PB.1	MFP6	I/O	SPI1 slave select 1 pin.
		PG.15	MFP2	I/O	
TM0	TM0_EXT	PB.1	MFP5	I/O	Timer0 external capture input/toggle output pin.
		PB.8	MFP7	I/O	
		PB.10	MFP3	I/O	
	TM0_ECNT	PA.0	MFP6	I/O	Timer0 event counter input/toggle output pin.
		PD.6	MFP3	I/O	
		PF.0	MFP3	I/O	
	TM0_TGL	PB.3	MFP5	I/O	Timer0 event counter input/toggle output pin.
		PC.0	MFP7	I/O	
		PB.9	MFP3	I/O	
TM1	TM1_EXT	PA.13	MFP3	I/O	Timer1 external capture input/toggle output

Group	Pin Name	GPIO	MFP	Type	Description
		PD.1	MFP3	I/O	pin.
		PG.12	MFP3	I/O	
		PF.9	MFP3	I/O	
	TM1_ECNT	PA.1	MFP6	I/O	Timer1 event counter input/toggle output pin.
		PD.7	MFP3	I/O	
		PF.1	MFP3	I/O	
	TM1_TGL	PA.14	MFP3	I/O	Timer1 event counter input/toggle output pin.
		PD.0	MFP3	I/O	
		PG.11	MFP3	I/O	
		PF.8	MFP3	I/O	
TM2	TM2_EXT	PA.9	MFP3	I/O	Timer2 external capture input/toggle output pin.
		PB.11	MFP3	I/O	
		PD.13	MFP2	I/O	
	TM2_ECNT	PA.2	MFP6	I/O	Timer2 event counter input/toggle output pin.
		PD.8	MFP3	I/O	
		PF.2	MFP3	I/O	
	TM2_TGL	PA.10	MFP3	I/O	Timer2 event counter input/toggle output pin.
		PB.12	MFP3	I/O	
		PD.12	MFP2	I/O	
TM3	TM3_EXT	PA.7	MFP3	I/O	Timer3 external capture input/toggle output pin.
		PD.15	MFP2	I/O	
	TM3_ECNT	PA.3	MFP6	I/O	Timer3 event counter input/toggle output pin.
		PD.9	MFP3	I/O	
		PF.3	MFP3	I/O	
	TM3_TGL	PA.8	MFP3	I/O	Timer3 event counter input/toggle output pin.
		PD.14	MFP2	I/O	
TM4	TM4_EXT	PA.11	MFP3	I/O	Timer4 external capture input/toggle output pin.
		PD.2	MFP3	I/O	
		PF.6	MFP3	I/O	
	TM4_ECNT	PA.4	MFP6	I/O	Timer4 event counter input/toggle output pin.
		PD.10	MFP3	I/O	
		PF.4	MFP3	I/O	
	TM4_TGL	PA.12	MFP3	I/O	Timer4 event counter input/toggle output pin.

Group	Pin Name	GPIO	MFP	Type	Description
		PD.3	MFP3	I/O	
		PB.13	MFP3	I/O	
TM5	TM5_EXT	PA.15	MFP3	I/O	Timer5 external capture input/toggle output pin.
		PD.4	MFP3	I/O	
		PF.7	MFP3	I/O	
	TM5_ECNT	PA.5	MFP6	I/O	Timer5 event counter input/toggle output pin.
		PD.11	MFP3	I/O	
		PF.5	MFP3	I/O	
	TM5_TGL	PG.10	MFP3	I/O	Timer5 event counter input/toggle output pin.
		PD.5	MFP3	I/O	
		PF.10	MFP3	I/O	
UART0	UART0_RXD	PF.11	MFP1	I	UART0 data receiver input pin.
	UART0_TXD	PF.12	MFP1	O	UART0 data transmitter output pin.
UART1	UART1_CTS	PC.8	MFP7	I	UART1 clear to Send input pin.
		PF.7	MFP2	I	
	UART1_RTS	PC.7	MFP7	O	UART1 request to Send output pin.
		PF.8	MFP2	O	
	UART1_RXD	PA.0	MFP4	I	UART1 data receiver input pin.
		PC.6	MFP7	I	
		PF.9	MFP2	I	
	UART1_TXD	PA.1	MFP4	O	UART1 data transmitter output pin.
		PC.5	MFP7	O	
		PF.10	MFP2	O	
UART2	UART2_CTS	PA.7	MFP2	I	UART2 clear to Send input pin.
		PG.2	MFP2	I	
		PB.0	MFP2	I	
	UART2_RTS	PA.8	MFP2	O	UART2 request to Send output pin.
		PG.3	MFP2	O	
	UART2_RXD	PA.9	MFP2	I	UART2 data receiver input pin.
		PG.0	MFP2	I	
		PD.7	MFP2	I	
	UART2_TXD	PA.10	MFP2	O	UART2 data transmitter output pin.
		PG.1	MFP2	O	

Group	Pin Name	GPIO	MFP	Type	Description
		PD.6	MFP2	O	
UART3	UART3_CTS	PB.12	MFP1	I	UART3 clear to Send input pin.
		PD.5	MFP2	I	
		PF.4	MFP5	I	
	UART3_RTS	PB.11	MFP1	O	UART3 request to Send output pin.
		PD.4	MFP2	O	
		PF.5	MFP5	O	
	UART3_RXD	PC.4	MFP5	I	UART3 data receiver input pin.
		PB.10	MFP1	I	
		PD.3	MFP2	I	
		PF.6	MFP5	I	
	UART3_TXD	PC.3	MFP5	O	UART3 data transmitter output pin.
		PB.9	MFP1	O	
		PD.2	MFP2	O	
		PB.13	MFP5	O	
		PF.7	MFP5	O	
UART4	UART4_CTS	PD.15	MFP1	I	UART4 clear to Send input pin.
		PE.0	MFP5	I	
	UART4_RTS	PD.14	MFP1	O	UART4 request to Send output pin.
		PE.1	MFP5	O	
	UART4_RXD	PC.10	MFP7	I	UART4 data receiver input pin.
		PD.13	MFP1	I	
		PE.2	MFP5	I	
	UART4_TXD	PC.9	MFP7	O	UART4 data transmitter output pin.
		PD.12	MFP1	O	
		PE.3	MFP5	O	
UART5	UART5_CTS	PG.4	MFP2	I	UART5 clear to Send input pin.
		PG.11	MFP5	I	
	UART5_RTS	PG.5	MFP2	O	UART5 request to Send output pin.
		PG.12	MFP5	O	
	UART5_RXD	PG.6	MFP2	I	UART5 data receiver input pin.
		PD.1	MFP2	I	
		PG.13	MFP5	I	

Group	Pin Name	GPIO	MFP	Type	Description
	UART5_TXD	PG.7	MFP2	O	UART5 data transmitter output pin.
		PD.0	MFP2	O	
		PG.14	MFP5	O	
UART6	UART6_CTS	PA.2	MFP1	I	UART6 clear to Send input pin.
		PD.8	MFP2	I	
	UART6_RTS	PA.3	MFP1	O	UART6 request to Send output pin.
		PD.9	MFP2	O	
	UART6_RXD	PA.4	MFP1	I	UART6 data receiver input pin.
		PD.11	MFP2	I	
		PE.8	MFP5	I	
	UART6_TXD	PA.5	MFP1	O	UART6 data transmitter output pin.
		PD.10	MFP2	O	
		PE.9	MFP5	O	
UART7	UART7_CTS	PB.7	MFP5	I	UART7 clear to Send input pin.
		PF.0	MFP5	I	
	UART7_RTS	PB.5	MFP5	O	UART7 request to Send output pin.
		PF.1	MFP5	O	
	UART7_RXD	PA.14	MFP6	I	UART7 data receiver input pin.
		PB.4	MFP5	I	
		PC.2	MFP4	I	
		PF.2	MFP5	I	
	UART7_TXD	PA.13	MFP6	O	UART7 data transmitter output pin.
		PB.6	MFP5	O	
		PC.1	MFP4	O	
		PF.3	MFP5	O	
UART8	UART8_CTS	PG.9	MFP2	I	UART8 clear to Send input pin.
		PC.15	MFP7	I	
	UART8_RTS	PG.8	MFP2	O	UART8 request to Send output pin.
		PC.14	MFP7	O	
	UART8_RXD	PA.11	MFP2	I	UART8 data receiver input pin.
		PC.0	MFP4	I	
		PC.13	MFP7	I	
	UART8_TXD	PA.12	MFP2	O	UART8 data transmitter output pin.

Group	Pin Name	GPIO	MFP	Type	Description
		PB.8	MFP4	O	
		PC.12	MFP7	O	
UART9	UART9_CTS	PE.4	MFP5	I	UART9 clear to Send input pin.
	UART9_RTS	PB.2	MFP7	O	UART9 request to Send output pin.
		PE.5	MFP5	O	
	UART9_RXD	PB.3	MFP7	I	UART9 data receiver input pin.
		PE.6	MFP5	I	
		PE.10	MFP3	I	
	UART9_TXD	PB.1	MFP7	O	UART9 data transmitter output pin.
		PE.7	MFP5	O	
		PE.12	MFP3	O	
USB0	USB0_VBUSVLD	PE.11	MFP1	I	USB0 external VBUS regulator status pin.
USBHL0	USBHL0_DM	PB.6	MFP4	A	USB 1.1 Host Lite 0 differential signal D-.
		PB.7	MFP4	A	
		PB.9	MFP4	A	
		PD.14	MFP5	A	
	USBHL0_DP	PB.4	MFP4	A	USB 1.1 Host Lite 0 differential signal D+.
		PB.5	MFP4	A	
		PB.10	MFP4	A	
		PD.15	MFP5	A	
USBHL1	USBHL1_DM	PF.0	MFP6	A	USB 1.1 Host Lite 1 differential signal D-.
		PE.0	MFP6	A	
	USBHL1_DP	PF.1	MFP6	A	USB 1.1 Host Lite 1 differential signal D+.
		PE.1	MFP6	A	
USBHL2	USBHL2_DM	PF.2	MFP6	A	USB 1.1 Host Lite 2 differential signal D-.
		PE.2	MFP6	A	
	USBHL2_DP	PF.3	MFP6	A	USB 1.1 Host Lite 2 differential signal D+.
		PE.3	MFP6	A	
USBHL3	USBHL3_DM	PF.4	MFP6	A	USB 1.1 Host Lite 3 differential signal D-.
		PE.4	MFP6	A	
	USBHL3_DP	PF.5	MFP6	A	USB 1.1 Host Lite 3 differential signal D+.
		PE.5	MFP6	A	
USBHL4	USBHL4_DM	PA.15	MFP4	A	USB 1.1 Host Lite 4 differential signal D-.

Group	Pin Name	GPIO	MFP	Type	Description
		PF.6	MFP6	A	
		PE.6	MFP6	A	
	USBHL4_DP	PG.10	MFP4	A	USB 1.1 Host Lite 4 differential signal D+.
		PB.13	MFP6	A	
		PF.7	MFP6	A	
		PE.7	MFP6	A	
USBHL5	USBHL5_DM	PA.13	MFP4	A	USB 1.1 Host Lite 5 differential signal D-.
		PB.11	MFP4	A	
		PF.8	MFP6	A	
		PE.8	MFP6	A	
	USBHL5_DP	PA.14	MFP4	A	USB 1.1 Host Lite 5 differential signal D+.
		PB.12	MFP4	A	
		PF.9	MFP6	A	
		PE.9	MFP6	A	
USBH	USBH_PWREN	PE.12	MFP1	O	HSUSB external VBUS regulator enable pin.
USB	USB_OVC	PE.10	MFP1	I	USB host bus power over voltage detector.
VCAP0	VCAP0_CLKO	PC.3	MFP2	O	Video image interface 0 sensor clock pin.
	VCAP0_DATA0	PC.8	MFP2	I	Video image interface 0 data 0 pin.
	VCAP0_DATA1	PC.9	MFP2	I	Video image interface 0 data 1 pin.
	VCAP0_DATA2	PC.10	MFP2	I	Video image interface 0 data 2 pin.
	VCAP0_DATA3	PC.11	MFP2	I	Video image interface 0 data 3 pin.
	VCAP0_DATA4	PC.12	MFP2	I	Video image interface 0 data 4 pin.
	VCAP0_DATA5	PC.13	MFP2	I	Video image interface 0 data 5 pin.
	VCAP0_DATA6	PC.14	MFP2	I	Video image interface 0 data 6 pin.
	VCAP0_DATA7	PC.15	MFP2	I	Video image interface 0 data 7 pin.
	VCAP0_FIELD	PC.7	MFP2	I	Video image interface 0 frame sync. pin.
	VCAP0_HSYNC	PC.5	MFP2	I	Video image interface 0 horizontal sync. pin.
	VCAP0_PCLK	PC.4	MFP2	I	Video image interface 0 pixel clock pin.
	VCAP0_VSYNC	PC.6	MFP2	I	Video image interface 0 vertical sync. pin.
VCAP1	VCAP1_CLKO	PE.12	MFP7	O	Video image interface 1 sensor clock pin.
	VCAP1_DATA0	PE.2	MFP7	I	Video image interface 1 data 0 pin.
	VCAP1_DATA1	PE.3	MFP7	I	Video image interface 1 data 1 pin.
	VCAP1_DATA2	PE.4	MFP7	I	Video image interface 1 data 2 pin.

Group	Pin Name	GPIO	MFP	Type	Description
	VCAP1_DATA3	PE.5	MFP7	I	Video image interface 1 data 3 pin.
	VCAP1_DATA4	PE.6	MFP7	I	Video image interface 1 data 4 pin.
	VCAP1_DATA5	PE.7	MFP7	I	Video image interface 1 data 5 pin.
	VCAP1_DATA6	PE.8	MFP7	I	Video image interface 1 data 6 pin.
	VCAP1_DATA7	PE.9	MFP7	I	Video image interface 1 data 7 pin.
	VCAP1_FIELD	PE.10	MFP7	I	Video image interface 1 frame sync. pin.
	VCAP1_HSYNC	PE.0	MFP7	I	Video image interface 1 horizontal sync. pin.
	VCAP1_PCLK	PF.10	MFP7	I	Video image interface 1 pixel clock pin.
	VCAP1_VSYNC	PE.1	MFP7	I	Video image interface 1 vertical sync. pin.
WDT	WDT_nRST	nRESET	MFP1	O	Watch dog timer reset trigger output.
eMMC0	eMMC0_CLK	PC.6	MFP6	O	eMMC0 clock output pin
	eMMC0_CMD	PC.5	MFP6	I/O	eMMC0 command/response pin
	eMMC0_DATA0	PC.7	MFP6	I/O	eMMC0 data line bit 0.
	eMMC0_DATA1	PC.8	MFP6	I/O	eMMC0 data line bit 1.
	eMMC0_DATA2	PC.9	MFP6	I/O	eMMC0 data line bit 2.
	eMMC0_DATA3	PC.10	MFP6	I/O	eMMC0 data line bit 3.
eMMC1	eMMC1_CLK	PF.1	MFP2	O	eMMC1 clock output pin
	eMMC1_CMD	PF.0	MFP2	I/O	eMMC1 command/response pin
	eMMC1_DATA0	PF.2	MFP2	I/O	eMMC1 data line bit 0.
	eMMC1_DATA1	PF.3	MFP2	I/O	eMMC1 data line bit 1.
	eMMC1_DATA2	PF.4	MFP2	I/O	eMMC1 data line bit 2.
	eMMC1_DATA3	PF.5	MFP2	I/O	eMMC1 data line bit 3.

4.2.3 NUC980 Multi-function Summary Table Sorted by GPIO

	Pin Name	Type	MFP	Description
PA.0	PA.0	I/O	MFP0	General purpose digital I/O pin.
	QSPI0_SS1	I/O	MFP1	Quad SPI0 slave select 1 pin.
	I2C0_SDA	I/O	MFP3	I^{2}C0 data input/output pin.
	UART1_RXD	I	MFP4	UART1 data receiver input pin.
	EINT0	I	MFP5	External interrupt 0 input pin.
	TM0_ECNT	I/O	MFP6	Timer0 event counter input/toggle output pin.
	CAN3_RXD	I	MFP7	CAN3 bus receiver input.
PA.1	PA.1	I/O	MFP0	General purpose digital I/O pin.
	EBI_nCS2	O	MFP1	EBI chip select 2 output pin.
	EBI_MCLK	O	MFP2	EBI external clock output pin.
	I2C0_SCL	I/O	MFP3	I^{2}C0 clock pin.
	UART1_TXD	O	MFP4	UART1 data transmitter output pin.
	EINT1	I	MFP5	External interrupt 1 input pin.
	TM1_ECNT	I/O	MFP6	Timer1 event counter input/toggle output pin.
	CAN3_TXD	O	MFP7	CAN3 bus transmitter output.
PA.2	PA.2	I/O	MFP0	General purpose digital I/O pin.
	UART6_CTS	I	MFP1	UART6 clear to Send input pin.
	I2S_LRCK	O	MFP2	I^{2}S_ left right channel clock output pin.
	SC0_CD	I	MFP3	Smart Card 0 card detect pin.
	JTAG1_TDO	O	MFP4	JTAG1 data output pin.
	TM2_ECNT	I/O	MFP6	Timer2 event counter input/toggle output pin.
PA.3	PA.3	I/O	MFP0	General purpose digital I/O pin.
	UART6_RTS	O	MFP1	UART6 request to Send output pin.
	I2S_BCLK	O	MFP2	I^{2}S_ bit clock output pin.
	SC0_PWR	O	MFP3	Smart Card 0 power pin.
	JTAG1_TCK	I	MFP4	JTAG1 clock input pin.
	TM3_ECNT	I/O	MFP6	Timer3 event counter input/toggle output pin.
PA.4	PA.4	I/O	MFP0	General purpose digital I/O pin.
	UART6_RXD	I	MFP1	UART6 data receiver input pin.
	I2S_DI	I	MFP2	I^{2}S_ data input pin.
	SC0_DAT	I/O	MFP3	Smart Card 0 data pin.
	JTAG1_TMS	I	MFP4	JTAG1 test mode selection input pin.

	Pin Name	Type	MFP	Description
	TM4_ECNT	I/O	MFP6	Timer4 event counter input/toggle output pin.
PA.5	PA.5	I/O	MFP0	General purpose digital I/O pin.
	UART6_TXD	O	MFP1	UART6 data transmitter output pin.
	I2S_DO	O	MFP2	I^{2}S_ data output pin.
	SC0_CLK	O	MFP3	Smart Card 0 clock pin.
	JTAG1_TDI	I	MFP4	JTAG1 data input pin.
	TM5_ECNT	I/O	MFP6	Timer5 event counter input/toggle output pin.
PA.6	PA.6	I/O	MFP0	General purpose digital I/O pin.
	EBI_nCS1	O	MFP1	EBI chip select 1 output pin.
	I2S_MCLK	O	MFP2	I^{2}S_ master clock output pin.
	SC0_RST	O	MFP3	Smart Card 0 reset pin.
	JTAG1_nTRST	I	MFP4	JTAG1 reset input pin.
PA.7	PA.7	I/O	MFP0	General purpose digital I/O pin.
	EBI_nWE	O	MFP1	EBI write enable output pin.
	UART2_CTS	I	MFP2	UART2 clear to Send input pin.
	TM3_EXT	I/O	MFP3	Timer3 external capture input/toggle output pin.
PA.8	PA.8	I/O	MFP0	General purpose digital I/O pin.
	EBI_nRE	O	MFP1	EBI read enable output pin.
	UART2_RTS	O	MFP2	UART2 request to Send output pin.
	TM3_TGL	I/O	MFP3	Timer3 event counter input/toggle output pin.
PA.9	PA.9	I/O	MFP0	General purpose digital I/O pin.
	EBI_nCS0	O	MFP1	EBI chip select 0 output pin.
	UART2_RXD	I	MFP2	UART2 data receiver input pin.
	TM2_EXT	I/O	MFP3	Timer2 external capture input/toggle output pin.
PA.10	PA.10	I/O	MFP0	General purpose digital I/O pin.
	EBI_ADDR10	O	MFP1	EBI address bus bit 10.
	UART2_TXD	O	MFP2	UART2 data transmitter output pin.
	TM2_TGL	I/O	MFP3	Timer2 event counter input/toggle output pin.
PA.11	PA.11	I/O	MFP0	General purpose digital I/O pin.
	EBI_ADDR9	O	MFP1	EBI address bus bit 9.
	UART8_RXD	I	MFP2	UART8 data receiver input pin.
	TM4_EXT	I/O	MFP3	Timer4 external capture input/toggle output pin.
PA.12	PA.12	I/O	MFP0	General purpose digital I/O pin.

	Pin Name	Type	MFP	Description
	EBI_ADDR8	O	MFP1	EBI address bus bit 8.
	UART8_TXD	O	MFP2	UART8 data transmitter output pin.
	TM4_TGL	I/O	MFP3	Timer4 event counter input/toggle output pin.
PA.13	PA.13	I/O	MFP0	General purpose digital I/O pin.
	EBI_ADDR13	O	MFP1	EBI address bus bit 13.
	I2C1_SDA	I/O	MFP2	I^{2}C1 data input/output pin.
	TM1_EXT	I/O	MFP3	Timer1 external capture input/toggle output pin.
	USBHL5_DM	A	MFP4	USB 1.1 Host Lite 5 differential signal D-.
	CAN1_RXD	I	MFP5	CAN1 bus receiver input.
	UART7_TXD	O	MFP6	UART7 data transmitter output pin.
	PWM03	O	MFP7	PWM03 counter synchronous trigger output pin.
	EINT0	I	MFP8	External interrupt 0 input pin.
PA.14	PA.14	I/O	MFP0	General purpose digital I/O pin.
	EBI_ADDR14	O	MFP1	EBI address bus bit 14.
	I2C1_SCL	I/O	MFP2	I^{2}C1 clock pin.
	TM1_TGL	I/O	MFP3	Timer1 event counter input/toggle output pin.
	USBHL5_DP	A	MFP4	USB 1.1 Host Lite 5 differential signal D+.
	CAN1_TXD	O	MFP5	CAN1 bus transmitter output.
	UART7_RXD	I	MFP6	UART7 data receiver input pin.
	PWM02	O	MFP7	PWM02 counter synchronous trigger output pin.
	EINT1	I	MFP8	External interrupt 1 input pin.
PA.15	PA.15	I/O	MFP0	General purpose digital I/O pin.
	EBI_ADDR19	O	MFP1	EBI address bus bit 19.
	I2C0_SDA	I/O	MFP2	I^{2}C0 data input/output pin.
	TM5_EXT	I/O	MFP3	Timer5 external capture input/toggle output pin.
	USBHL4_DM	A	MFP4	USB 1.1 Host Lite 4 differential signal D-.
	CAN2_RXD	I	MFP5	CAN2 bus receiver input.
	SPI1_SS0	I/O	MFP6	SPI1 slave select 0 pin.
	PWM01	O	MFP7	PWM01 counter synchronous trigger output pin.
	I2S_LRCK	O	MFP8	I^{2}S_ left right channel clock output pin.
PB.0	PB.0	I/O	MFP0	General purpose digital I/O pin.
	EBI_ADDR12	O	MFP1	EBI address bus bit 12.
	UART2_CTS	I	MFP2	UART2 clear to Send input pin.

	Pin Name	Type	MFP	Description
	ADC_AIN0	A	MFP8	ADC channel 0 analog input.
PB.1	PB.1	I/O	MFP0	General purpose digital I/O pin.
	EBI_ADDR17	O	MFP1	EBI address bus bit 17.
	I2C3_SDA	I/O	MFP2	I^{2}C3 data input/output pin.
	I2S_MCLK	O	MFP3	I^{2}S_ master clock output pin.
	CAN2_RXD	I	MFP4	CAN2 bus receiver input.
	TM0_EXT	I/O	MFP5	Timer0 external capture input/toggle output pin.
	SPI1_SS1	I/O	MFP6	SPI1 slave select 1 pin.
	UART9_TXD	O	MFP7	UART9 data transmitter output pin.
	ADC_AIN1	A	MFP8	ADC channel 1 analog input.
PB.2	PB.2	I/O	MFP0	General purpose digital I/O pin.
	EBI_ADDR2	O	MFP1	EBI address bus bit 2.
	UART9_RTS	O	MFP7	UART9 request to Send output pin.
	ADC_AIN2	A	MFP8	ADC channel 2 analog input.
PB.3	PB.3	I/O	MFP0	General purpose digital I/O pin.
	EBI_ADDR18	O	MFP1	EBI address bus bit 18.
	I2C3_SCL	I/O	MFP2	I^{2}C3 clock pin.
	EINT2	I	MFP3	External interrupt 2 input pin.
	CAN2_TXD	O	MFP4	CAN2 bus transmitter output.
	TM0_TGL	I/O	MFP5	Timer0 event counter input/toggle output pin.
	SPI0_SS1	I/O	MFP6	SPI0 slave select 1 pin.
	UART9_RXD	I	MFP7	UART9 data receiver input pin.
	ADC_AIN3	A	MFP8	ADC channel 3 analog input.
PB.4	PB.4	I/O	MFP0	General purpose digital I/O pin.
	EBI_ADDR14	O	MFP1	EBI address bus bit 14.
	I2C1_SCL	I/O	MFP2	I^{2}C1 clock pin.
	I2S_BCLK	O	MFP3	I^{2}S_ bit clock output pin.
	USBHL0_DP	A	MFP4	USB 1.1 Host Lite 0 differential signal D+.
	UART7_RXD	I	MFP5	UART7 data receiver input pin.
	SPI1_CLK	I/O	MFP6	SPI1 serial clock pin.
	ADC_AIN4	A	MFP8	ADC channel 4 analog input.
PB.5	PB.5	I/O	MFP0	General purpose digital I/O pin.
	EBI_ADDR16	O	MFP1	EBI address bus bit 16.

	Pin Name	Type	MFP	Description
	I2C2_SCL	I/O	MFP2	I^{2}C2 clock pin.
	I2S_DO	O	MFP3	I^{2}S_ data output pin.
	USBHL0_DP	A	MFP4	USB 1.1 Host Lite 0 differential signal D+.
	UART7_RTS	O	MFP5	UART7 request to Send output pin.
	SPI1_MISO	I/O	MFP6	SPI1 MISO (Master In, Slave Out) pin.
	ADC_AIN5	A	MFP8	ADC channel 5 analog input.
PB.6	PB.6	I/O	MFP0	General purpose digital I/O pin.
	EBI_ADDR13	O	MFP1	EBI address bus bit 13.
	I2C1_SDA	I/O	MFP2	I^{2}C1 data input/output pin.
	I2S_LRCK	O	MFP3	I^{2}S_ left right channel clock output pin.
	USBHL0_DM	A	MFP4	USB 1.1 Host Lite 0 differential signal D-.
	UART7_TXD	O	MFP5	UART7 data transmitter output pin.
	SPI1_SS0	I/O	MFP6	SPI1 slave select 0 pin.
	ADC_AIN6	A	MFP8	ADC channel 6 analog input.
PB.7	PB.7	I/O	MFP0	General purpose digital I/O pin.
	EBI_ADDR15	O	MFP1	EBI address bus bit 15.
	I2C2_SDA	I/O	MFP2	I^{2}C2 data input/output pin.
	I2S_DI	I	MFP3	I^{2}S_ data input pin.
	USBHL0_DM	A	MFP4	USB 1.1 Host Lite 0 differential signal D-.
	UART7_CTS	I	MFP5	UART7 clear to Send input pin.
	SPI1_MOSI	I/O	MFP6	SPI1 MOSI (Master Out, Slave In) pin.
	ADC_AIN7	A	MFP8	ADC channel 7 analog input.
PB.8	PB.8	I/O	MFP0	General purpose digital I/O pin.
	EBI_ADDR11	O	MFP1	EBI address bus bit 11.
	I2C2_SCL	I/O	MFP2	I^{2}C2 clock pin.
	CAN2_RXD	I	MFP3	CAN2 bus receiver input.
	UART8_TXD	O	MFP4	UART8 data transmitter output pin.
	SD0_nCD	I	MFP6	SD/SDIO0 card detect input pin
	TM0_EXT	I/O	MFP7	Timer0 external capture input/toggle output pin.
PB.9	PB.9	I/O	MFP0	General purpose digital I/O pin.
	UART3_TXD	O	MFP1	UART3 data transmitter output pin.
	PWM13	O	MFP2	PWM13 counter synchronous trigger output pin.
	TM0_TGL	I/O	MFP3	Timer0 event counter input/toggle output pin.

	Pin Name	Type	MFP	Description
	USBHL0_DM	A	MFP4	USB 1.1 Host Lite 0 differential signal D-.
	SPI1_SS0	I/O	MFP5	SPI1 slave select 0 pin.
PB.10	PB.10	I/O	MFP0	General purpose digital I/O pin.
	UART3_RXD	I	MFP1	UART3 data receiver input pin.
	PWM12	O	MFP2	PWM12 counter synchronous trigger output pin.
	TM0_EXT	I/O	MFP3	Timer0 external capture input/toggle output pin.
	USBHL0_DP	A	MFP4	USB 1.1 Host Lite 0 differential signal D+.
	SPI1_CLK	I/O	MFP5	SPI1 serial clock pin.
PB.11	PB.11	I/O	MFP0	General purpose digital I/O pin.
	UART3_RTS	O	MFP1	UART3 request to Send output pin.
	PWM11	O	MFP2	PWM11 counter synchronous trigger output pin.
	TM2_EXT	I/O	MFP3	Timer2 external capture input/toggle output pin.
	USBHL5_DM	A	MFP4	USB 1.1 Host Lite 5 differential signal D-.
	SPI1_MOSI	I/O	MFP5	SPI1 MOSI (Master Out, Slave In) pin.
PB.12	PB.12	I/O	MFP0	General purpose digital I/O pin.
	UART3_CTS	I	MFP1	UART3 clear to Send input pin.
	PWM10	O	MFP2	PWM10 counter synchronous trigger output pin.
	TM2_TGL	I/O	MFP3	Timer2 event counter input/toggle output pin.
	USBHL5_DP	A	MFP4	USB 1.1 Host Lite 5 differential signal D+.
	SPI1_MISO	I/O	MFP5	SPI1 MISO (Master In, Slave Out) pin.
PB.13	PB.13	I/O	MFP0	General purpose digital I/O pin.
	EINT2	I	MFP2	External interrupt 2 input pin.
	TM4_TGL	I/O	MFP3	Timer4 event counter input/toggle output pin.
	PWM02	O	MFP4	PWM02 counter synchronous trigger output pin.
	UART3_TXD	O	MFP5	UART3 data transmitter output pin.
	USBHL4_DP	A	MFP6	USB 1.1 Host Lite 4 differential signal D+.
	EBI_DATA0	I/O	MFP8	EBI data bus bit 0.
PC.0	PC.0	I/O	MFP0	General purpose digital I/O pin.
	EBI_DATA0	I/O	MFP1	EBI data bus bit 0.
	I2C2_SDA	I/O	MFP2	I^{2}C2 data input/output pin.
	CAN2_TXD	O	MFP3	CAN2 bus transmitter output.
	UART8_RXD	I	MFP4	UART8 data receiver input pin.
	SPI0_SS1	I/O	MFP5	SPI0 slave select 1 pin.

	Pin Name	Type	MFP	Description
	TM0_TGL	I/O	MFP7	Timer0 event counter input/toggle output pin.
PC.1	PC.1	I/O	MFP0	General purpose digital I/O pin.
	EBI_DATA1	I/O	MFP1	EBI data bus bit 1.
	NAND_nCS0	O	MFP3	NAND Flash chip enable input.
	UART7_TXD	O	MFP4	UART7 data transmitter output pin.
PC.2	PC.2	I/O	MFP0	General purpose digital I/O pin.
	EBI_DATA2	I/O	MFP1	EBI data bus bit 2.
	NAND_nWP	O	MFP3	NAND Flash write protect input.
	UART7_RXD	I	MFP4	UART7 data receiver input pin.
PC.3	PC.3	I/O	MFP0	General purpose digital I/O pin.
	EBI_DATA3	I/O	MFP1	EBI data bus bit 3.
	VCAP0_CLKO	O	MFP2	Video image interface 0 sensor clock pin.
	NAND_ALE	O	MFP3	NAND Flash address latch enable.
	I2C1_SCL	I/O	MFP4	I^{2}C1 clock pin.
	UART3_TXD	O	MFP5	UART3 data transmitter output pin.
	CAN0_RXD	I	MFP7	CAN0 bus receiver input.
PC.4	PC.4	I/O	MFP0	General purpose digital I/O pin.
	EBI_DATA4	I/O	MFP1	EBI data bus bit 4.
	VCAP0_PCLK	I	MFP2	Video image interface 0 pixel clock pin.
	NAND_CLE	O	MFP3	NAND Flash command latch enable.
	I2C1_SDA	I/O	MFP4	I^{2}C1 data input/output pin.
	UART3_RXD	I	MFP5	UART3 data receiver input pin.
	SPI0_MOSI	I/O	MFP6	SPI0 MOSI (Master Out, Slave In) pin.
	CAN0_TXD	O	MFP7	CAN0 bus transmitter output.
PC.5	PC.5	I/O	MFP0	General purpose digital I/O pin.
	EBI_DATA5	I/O	MFP1	EBI data bus bit 5.
	VCAP0_HSYNC	I	MFP2	Video image interface 0 horizontal sync. pin.
	NAND_nWE	O	MFP3	NAND Flash write enable.
	SPI0_SS0	I/O	MFP5	SPI0 slave select 0 pin.
	SD0_CMD/eMMC0_CMD	I/O	MFP6	SD/SDIO0 command/response pin eMMC0 command/response pin
	UART1_TXD	O	MFP7	UART1 data transmitter output pin.
PC.6	PC.6	I/O	MFP0	General purpose digital I/O pin.
	EBI_DATA6	I/O	MFP1	EBI data bus bit 6.

	Pin Name	Type	MFP	Description
	VCAP0_VSYNC	I	MFP2	Video image interface 0 vertical sync. pin.
	NAND_nRE	O	MFP3	NAND Flash read enable.
	SC1_RST	O	MFP4	Smart Card 1 reset pin.
	SPI0_CLK	I/O	MFP5	SPI0 serial clock pin.
	SD0_CLK/eMMC0_CLK	O	MFP6	SD/SDIO0 clock output pin eMMC0 clock output pin
	UART1_RXD	I	MFP7	UART1 data receiver input pin.
PC.7	PC.7	I/O	MFP0	General purpose digital I/O pin.
	EBI_DATA7	I/O	MFP1	EBI data bus bit 7.
	VCAP0_FIELD	I	MFP2	Video image interface 0 frame sync. pin.
	NAND_RDY0	I	MFP3	NAND Flash ready/busy pin.
	SC1_CLK	O	MFP4	Smart Card 1 clock pin.
	SPI0_MOSI	I/O	MFP5	SPI0 MOSI (Master Out, Slave In) pin.
	SD0_DATA0/eMMC0_DATA0	I/O	MFP6	SD/SDIO0 data line bit 0. eMMC0 data line bit 0.
	UART1_RTS	O	MFP7	UART1 request to Send output pin.
PC.8	PC.8	I/O	MFP0	General purpose digital I/O pin.
	EBI_DATA8	I/O	MFP1	EBI data bus bit 8.
	VCAP0_DATA0	I	MFP2	Video image interface 0 data 0 pin.
	NAND_DATA0	I/O	MFP3	NAND Flash data bus bit 0.
	SC1_DAT	I/O	MFP4	Smart Card 1 data pin.
	SPI0_MISO	I/O	MFP5	SPI0 MISO (Master In, Slave Out) pin.
	SD0_DATA1/eMMC0_DATA1	I/O	MFP6	SD/SDIO0 data line bit 1. eMMC0 data line bit 1.
	UART1_CTS	I	MFP7	UART1 clear to Send input pin.
PC.9	PC.9	I/O	MFP0	General purpose digital I/O pin.
	EBI_DATA9	I/O	MFP1	EBI data bus bit 9.
	VCAP0_DATA1	I	MFP2	Video image interface 0 data 1 pin.
	NAND_DATA1	I/O	MFP3	NAND Flash data bus bit 1.
	SC1_PWR	O	MFP4	Smart Card 1 power pin.
	SD0_DATA2/eMMC0_DATA2	I/O	MFP6	SD/SDIO0 data line bit 2. eMMC0 data line bit 2.
	UART4_TXD	O	MFP7	UART4 data transmitter output pin.
PC.10	PC.10	I/O	MFP0	General purpose digital I/O pin.

	Pin Name	Type	MFP	Description
	EBI_DATA10	I/O	MFP1	EBI data bus bit 10.
	VCAP0_DATA2	I	MFP2	Video image interface 0 data 2 pin.
	NAND_DATA2	I/O	MFP3	NAND Flash data bus bit 2.
	SC1_CD	I	MFP4	Smart Card 1 card detect pin.
	SD0_DATA3/eMMC0_DATA3	I/O	MFP6	SD/SDIO0 data line bit 3. eMMC0 data line bit 3.
	UART4_RXD	I	MFP7	UART4 data receiver input pin.
PC.11	PC.11	I/O	MFP0	General purpose digital I/O pin.
	EBI_DATA11	I/O	MFP1	EBI data bus bit 11.
	VCAP0_DATA3	I	MFP2	Video image interface 0 data 3 pin.
	NAND_DATA3	I/O	MFP3	NAND Flash data bus bit 3.
	SC0_RST	O	MFP4	Smart Card 0 reset pin.
PC.12	PC.12	I/O	MFP0	General purpose digital I/O pin.
	EBI_DATA12	I/O	MFP1	EBI data bus bit 12.
	VCAP0_DATA4	I	MFP2	Video image interface 0 data 4 pin.
	NAND_DATA4	I/O	MFP3	NAND Flash data bus bit 4.
	SC0_CLK	O	MFP4	Smart Card 0 clock pin.
	SD0_nCD	I	MFP6	SD/SDIO0 card detect input pin
	UART8_TXD	O	MFP7	UART8 data transmitter output pin.
PC.13	PC.13	I/O	MFP0	General purpose digital I/O pin.
	EBI_DATA13	I/O	MFP1	EBI data bus bit 13.
	VCAP0_DATA5	I	MFP2	Video image interface 0 data 5 pin.
	NAND_DATA5	I/O	MFP3	NAND Flash data bus bit 5.
	SC0_DAT	I/O	MFP4	Smart Card 0 data pin.
	UART8_RXD	I	MFP7	UART8 data receiver input pin.
PC.14	PC.14	I/O	MFP0	General purpose digital I/O pin.
	EBI_DATA14	I/O	MFP1	EBI data bus bit 14.
	VCAP0_DATA6	I	MFP2	Video image interface 0 data 6 pin.
	NAND_DATA6	I/O	MFP3	NAND Flash data bus bit 6.
	SC0_PWR	O	MFP4	Smart Card 0 power pin.
	SPI0_MOSI	I/O	MFP5	SPI0 MOSI (Master Out, Slave In) pin.
	UART8_RTS	O	MFP7	UART8 request to Send output pin.
PC.15	PC.15	I/O	MFP0	General purpose digital I/O pin.
	EBI_DATA15	I/O	MFP1	EBI data bus bit 15.

	Pin Name	Type	MFP	Description
	VCAP0_DATA7	I	MFP2	Video image interface 0 data 7 pin.
	NAND_DATA7	I/O	MFP3	NAND Flash data bus bit 7.
	SC0_CD	I	MFP4	Smart Card 0 card detect pin.
	UART8_CTS	I	MFP7	UART8 clear to Send input pin.
PD.0	PD.0	I/O	MFP0	General purpose digital I/O pin.
	QSPI0_SS1	I/O	MFP1	Quad SPI0 slave select 1 pin.
	UART5_TXD	O	MFP2	UART5 data transmitter output pin.
	TM1_TGL	I/O	MFP3	Timer1 event counter input/toggle output pin.
	EINT2	I	MFP4	External interrupt 2 input pin.
PD.1	PD.1	I/O	MFP0	General purpose digital I/O pin.
	SPI0_SS1	I/O	MFP1	SPI0 slave select 1 pin.
	UART5_RXD	I	MFP2	UART5 data receiver input pin.
	TM1_EXT	I/O	MFP3	Timer1 external capture input/toggle output pin.
	EINT3	I	MFP4	External interrupt 3 input pin.
PD.2	PD.2	I/O	MFP0	General purpose digital I/O pin.
	QSPI0_SS0	I/O	MFP1	Quad SPI0 slave select 0 pin.
	UART3_TXD	O	MFP2	UART3 data transmitter output pin.
	TM4_EXT	I/O	MFP3	Timer4 external capture input/toggle output pin.
PD.3	PD.3	I/O	MFP0	General purpose digital I/O pin.
	QSPI0_CLK	I/O	MFP1	Quad SPI0 serial clock pin.
	UART3_RXD	I	MFP2	UART3 data receiver input pin.
	TM4_TGL	I/O	MFP3	Timer4 event counter input/toggle output pin.
PD.4	PD.4	I/O	MFP0	General purpose digital I/O pin.
	QSPI0_MOSI0	I/O	MFP1	Quad SPI0 MOSI0 (Master Out, Slave In) pin.
	UART3_RTS	O	MFP2	UART3 request to Send output pin.
	TM5_EXT	I/O	MFP3	Timer5 external capture input/toggle output pin.
PD.5	PD.5	I/O	MFP0	General purpose digital I/O pin.
	QSPI0_MISO0	I/O	MFP1	Quad SPI0 MISO0 (Master In, Slave Out) pin.
	UART3_CTS	I	MFP2	UART3 clear to Send input pin.
	TM5_TGL	I/O	MFP3	Timer5 event counter input/toggle output pin.
PD.6	PD.6	I/O	MFP0	General purpose digital I/O pin.
	QSPI0_MOSI1	I/O	MFP1	Quad SPI0 MOSI1 (Master Out, Slave In) pin.
	UART2_TXD	O	MFP2	UART2 data transmitter output pin.

	Pin Name	Type	MFP	Description
	TM0_ECNT	I/O	MFP3	Timer0 event counter input/toggle output pin.
	CAN0_RXD	I	MFP4	CAN0 bus receiver input.
PD.7	PD.7	I/O	MFP0	General purpose digital I/O pin.
	QSPI0_MISO1	I/O	MFP1	Quad SPI0 MISO1 (Master In, Slave Out) pin.
	UART2_RXD	I	MFP2	UART2 data receiver input pin.
	TM1_ECNT	I/O	MFP3	Timer1 event counter input/toggle output pin.
	CAN0_TXD	O	MFP4	CAN0 bus transmitter output.
PD.8	PD.8	I/O	MFP0	General purpose digital I/O pin.
	SPI0_SS0	I/O	MFP1	SPI0 slave select 0 pin.
	UART6_CTS	I	MFP2	UART6 clear to Send input pin.
	TM2_ECNT	I/O	MFP3	Timer2 event counter input/toggle output pin.
PD.9	PD.9	I/O	MFP0	General purpose digital I/O pin.
	SPI0_CLK	I/O	MFP1	SPI0 serial clock pin.
	UART6_RTS	O	MFP2	UART6 request to Send output pin.
	TM3_ECNT	I/O	MFP3	Timer3 event counter input/toggle output pin.
PD.10	PD.10	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MOSI	I/O	MFP1	SPI0 MOSI (Master Out, Slave In) pin.
	UART6_TXD	O	MFP2	UART6 data transmitter output pin.
	TM4_ECNT	I/O	MFP3	Timer4 event counter input/toggle output pin.
PD.11	PD.11	I/O	MFP0	General purpose digital I/O pin.
	SPI0_MISO	I/O	MFP1	SPI0 MISO (Master In, Slave Out) pin.
	UART6_RXD	I	MFP2	UART6 data receiver input pin.
	TM5_ECNT	I/O	MFP3	Timer5 event counter input/toggle output pin.
PD.12	PD.12	I/O	MFP0	General purpose digital I/O pin.
	UART4_TXD	O	MFP1	UART4 data transmitter output pin.
	TM2_TGL	I/O	MFP2	Timer2 event counter input/toggle output pin.
	CAN2_RXD	I	MFP4	CAN2 bus receiver input.
	PWM00	O	MFP6	PWM00 counter synchronous trigger output pin.
	EBI_DATA1	I/O	MFP8	EBI data bus bit 1.
PD.13	PD.13	I/O	MFP0	General purpose digital I/O pin.
	UART4_RXD	I	MFP1	UART4 data receiver input pin.
	TM2_EXT	I/O	MFP2	Timer2 external capture input/toggle output pin.
	CAN2_TXD	O	MFP4	CAN2 bus transmitter output.

	Pin Name	Type	MFP	Description
	PWM01	O	MFP6	PWM01 counter synchronous trigger output pin.
	EBI_DATA2	I/O	MFP8	EBI data bus bit 2.
PD.14	PD.14	I/O	MFP0	General purpose digital I/O pin.
	UART4_RTS	O	MFP1	UART4 request to Send output pin.
	TM3_TGL	I/O	MFP2	Timer3 event counter input/toggle output pin.
	I2C3_SCL	I/O	MFP3	I^{2}C3 clock pin.
	CAN1_RXD	I	MFP4	CAN1 bus receiver input.
	USBHL0_DM	A	MFP5	USB 1.1 Host Lite 0 differential signal D-.
	PWM02	O	MFP6	PWM02 counter synchronous trigger output pin.
	EBI_DATA3	I/O	MFP8	EBI data bus bit 3.
PD.15	PD.15	I/O	MFP0	General purpose digital I/O pin.
	UART4_CTS	I	MFP1	UART4 clear to Send input pin.
	TM3_EXT	I/O	MFP2	Timer3 external capture input/toggle output pin.
	I2C3_SDA	I/O	MFP3	I^{2}C3 data input/output pin.
	CAN1_TXD	O	MFP4	CAN1 bus transmitter output.
	USBHL0_DP	A	MFP5	USB 1.1 Host Lite 0 differential signal D+.
	PWM03	O	MFP6	PWM03 counter synchronous trigger output pin.
	EBI_DATA4	I/O	MFP8	EBI data bus bit 4.
PE.0	PE.0	I/O	MFP0	General purpose digital I/O pin.
	RMII0_RXERR	I	MFP1	RMII0 Receive Data Error input pin.
	CAN0_RXD	I	MFP2	CAN0 bus receiver input.
	UART4_CTS	I	MFP5	UART4 clear to Send input pin.
	USBHL1_DM	A	MFP6	USB 1.1 Host Lite 1 differential signal D-.
	VCAP1_HSYNC	I	MFP7	Video image interface 1 horizontal sync. pin.
PE.1	PE.1	I/O	MFP0	General purpose digital I/O pin.
	RMII0_CRSDV	I	MFP1	RMII0 Carrier Sense/Receive Data input pin.
	CAN0_TXD	O	MFP2	CAN0 bus transmitter output.
	UART4_RTS	O	MFP5	UART4 request to Send output pin.
	USBHL1_DP	A	MFP6	USB 1.1 Host Lite 1 differential signal D+.
	VCAP1_VSYNC	I	MFP7	Video image interface 1 vertical sync. pin.
PE.2	PE.2	I/O	MFP0	General purpose digital I/O pin.
	RMII0_RXD1	I	MFP1	RMII0 Receive Data bus bit 1.
	CAN1_RXD	I	MFP2	CAN1 bus receiver input.

	Pin Name	Type	MFP	Description
	UART4_RXD	I	MFP5	UART4 data receiver input pin.
	USBHL2_DM	A	MFP6	USB 1.1 Host Lite 2 differential signal D-.
	VCAP1_DATA0	I	MFP7	Video image interface 1 data 0 pin.
PE.3	PE.3	I/O	MFP0	General purpose digital I/O pin.
	RMII0_RXD0	I	MFP1	RMII0 Receive Data bus bit 0.
	CAN1_TXD	O	MFP2	CAN1 bus transmitter output.
	UART4_TXD	O	MFP5	UART4 data transmitter output pin.
	USBHL2_DP	A	MFP6	USB 1.1 Host Lite 2 differential signal D+.
	VCAP1_DATA1	I	MFP7	Video image interface 1 data 1 pin.
PE.4	PE.4	I/O	MFP0	General purpose digital I/O pin.
	RMII0_REFCLK	I	MFP1	RMII0 mode clock input pin.
	CAN2_RXD	I	MFP2	CAN2 bus receiver input.
	UART9_CTS	I	MFP5	UART9 clear to Send input pin.
	USBHL3_DM	A	MFP6	USB 1.1 Host Lite 3 differential signal D-.
	VCAP1_DATA2	I	MFP7	Video image interface 1 data 2 pin.
PE.5	PE.5	I/O	MFP0	General purpose digital I/O pin.
	RMII0_TXEN	O	MFP1	RMII0 Transmit Enable output pin.
	CAN2_TXD	O	MFP2	CAN2 bus transmitter output.
	UART9_RTS	O	MFP5	UART9 request to Send output pin.
	USBHL3_DP	A	MFP6	USB 1.1 Host Lite 3 differential signal D+.
	VCAP1_DATA3	I	MFP7	Video image interface 1 data 3 pin.
PE.6	PE.6	I/O	MFP0	General purpose digital I/O pin.
	RMII0_TXD1	O	MFP1	RMII0 Transmit Data bus bit 1.
	CAN3_RXD	I	MFP2	CAN3 bus receiver input.
	UART9_RXD	I	MFP5	UART9 data receiver input pin.
	USBHL4_DM	A	MFP6	USB 1.1 Host Lite 4 differential signal D-.
	VCAP1_DATA4	I	MFP7	Video image interface 1 data 4 pin.
PE.7	PE.7	I/O	MFP0	General purpose digital I/O pin.
	RMII0_TXD0	O	MFP1	RMII0 Transmit Data bus bit 0.
	CAN3_TXD	O	MFP2	CAN3 bus transmitter output.
	UART9_TXD	O	MFP5	UART9 data transmitter output pin.
	USBHL4_DP	A	MFP6	USB 1.1 Host Lite 4 differential signal D+.
	VCAP1_DATA5	I	MFP7	Video image interface 1 data 5 pin.

	Pin Name	Type	MFP	Description
PE.8	PE.8	I/O	MFP0	General purpose digital I/O pin.
	RMII0_MDIO	I/O	MFP1	RMII0 PHY Management Data pin.
	UART6_RXD	I	MFP5	UART6 data receiver input pin.
	USBHL5_DM	A	MFP6	USB 1.1 Host Lite 5 differential signal D-.
	VCAP1_DATA6	I	MFP7	Video image interface 1 data 6 pin.
PE.9	PE.9	I/O	MFP0	General purpose digital I/O pin.
	RMII0_MDC	O	MFP1	RMII0 PHY Management Clock output pin.
	UART6_TXD	O	MFP5	UART6 data transmitter output pin.
	USBHL5_DP	A	MFP6	USB 1.1 Host Lite 5 differential signal D+.
	VCAP1_DATA7	1	MFP7	Video image interface 1 ata 7 pin.
PE.10	PE.10	I/O	MFP0	General purpose digital I/O pin.
	USB_OVC	I	MFP1	USB host bus power over voltage detector.
	CAN3_RXD	I	MFP2	CAN3 bus receiver input.
	UART9_RXD	I	MFP3	UART9 data receiver input pin.
	PWM12	O	MFP4	PWM12 counter synchronous trigger output pin.
	EINT2	I	MFP5	External interrupt 2 input pin.
	I2C0_SDA	I/O	MFP6	I^{2}C0 data input/output pin.
	VCAP1_FIELD	I	MFP7	Video image interface 1 frame sync. pin.
PE.11	PE.11	I/O	MFP0	General purpose digital I/O pin.
	USB0_VBUSVLD	I	MFP1	USB0 external VBUS regulator status pin.
PE.12	PE.12	I/O	MFP0	General purpose digital I/O pin.
	USBH_PWREN	O	MFP1	HSUSB external VBUS regulator enable pin.
	CAN3_TXD	O	MFP2	CAN3 bus transmitter output.
	UART9_TXD	O	MFP3	UART9 data transmitter output pin.
	PWM13	O	MFP4	PWM13 counter synchronous trigger output pin.
	EINT3	I	MFP5	External interrupt 3 input pin.
	I2C0_SCL	I/O	MFP6	I^{2}C0 clock pin.
	VCAP1_CLKO	O	MFP7	Video image interface 1 sensor clock pin.
PF.0	PF.0	I/O	MFP0	General purpose digital I/O pin.
	RMII1_RXERR	I	MFP1	RMII1 Receive Data Error input pin.
	SD1_CMD/eMMC1_CMD	I/O	MFP2	SD/SDIO1 command/response pin eMMC1 command/response pin
	TM0_ECNT	I/O	MFP3	Timer0 event counter input/toggle output pin.
	SC1_RST	O	MFP4	Smart Card 1 reset pin.

	Pin Name	Type	MFP	Description
	UART7_CTS	I	MFP5	UART7 clear to Send input pin.
	USBHL1_DM	A	MFP6	USB 1.1 Host Lite 1 differential signal D-.
	EBI_DATA5	I/O	MFP8	EBI data bus bit 5.
PF.1	PF.1	I/O	MFP0	General purpose digital I/O pin.
	RMII1_CRSDV	I	MFP1	RMII1 Carrier Sense/Receive Data input pin.
	SD1_CLK/eMMC1_CLK	O	MFP2	SD/SDIO1 clock output pin eMMC1 clock output pin
	TM1_ECNT	I/O	MFP3	Timer1 event counter input/toggle output pin.
	SC1_CLK	O	MFP4	Smart Card 1 clock pin.
	UART7_RTS	O	MFP5	UART7 request to Send output pin.
	USBHL1_DP	A	MFP6	USB 1.1 Host Lite 1 differential signal D+.
	EBI_DATA6	I/O	MFP8	EBI data bus bit 6.
PF.2	PF.2	I/O	MFP0	General purpose digital I/O pin.
	RMII1_RXD1	I	MFP1	RMII1 Receive Data bus bit 1.
	SD1_DATA0/eMMC1_DATA0	I/O	MFP2	SD/SDIO1 data line bit 0. eMMC1 data line bit 0.
	TM2_ECNT	I/O	MFP3	Timer2 event counter input/toggle output pin.
	SC1_DAT	I/O	MFP4	Smart Card 1 data pin.
	UART7_RXD	I	MFP5	UART7 data receiver input pin.
	USBHL2_DM	A	MFP6	USB 1.1 Host Lite 2 differential signal D-.
	EBI_DATA7	I/O	MFP8	EBI data bus bit 7.
PF.3	PF.3	I/O	MFP0	General purpose digital I/O pin.
	RMII1_RXD0	I	MFP1	RMII1 Receive Data bus bit 0.
	SD1_DATA1/eMMC1_DATA1	I/O	MFP2	SD/SDIO1 data line bit 1. eMMC1 data line bit 1.
	TM3_ECNT	I/O	MFP3	Timer3 event counter input/toggle output pin.
	SC1_PWR	O	MFP4	Smart Card 1 power pin.
	UART7_TXD	O	MFP5	UART7 data transmitter output pin.
	USBHL2_DP	A	MFP6	USB 1.1 Host Lite 2 differential signal D+.
	EBI_DATA8	I/O	MFP8	EBI data bus bit 8.
PF.4	PF.4	I/O	MFP0	General purpose digital I/O pin.
	RMII1_REFCLK	I	MFP1	RMII1 mode clock input pin.
	SD1_DATA2/eMMC1_DATA2	I/O	MFP2	SD/SDIO1 data line bit 2. eMMC1 data line bit 2.

	Pin Name	Type	MFP	Description
	TM4_ECNT	I/O	MFP3	Timer4 event counter input/toggle output pin.
	SC1_CD	I	MFP4	Smart Card 1 card detect pin.
	UART3_CTS	I	MFP5	UART3 clear to Send input pin.
	USBHL3_DM	A	MFP6	USB 1.1 Host Lite 3 differential signal D-.
	EBI_DATA9	I/O	MFP8	EBI data bus bit 9.
PF.5	PF.5	I/O	MFP0	General purpose digital I/O pin.
	RMII1_TXEN	O	MFP1	RMII1 Transmit Enable output pin.
	SD1_DATA3/eMMC1_DATA3	I/O	MFP2	SD/SDIO1 data line bit 3. eMMC1 data line bit 3.
	TM5_ECNT	I/O	MFP3	Timer5 event counter input/toggle output pin.
	PWM00	O	MFP4	PWM00 counter synchronous trigger output pin.
	UART3_RTS	O	MFP5	UART3 request to Send output pin.
	USBHL3_DP	A	MFP6	USB 1.1 Host Lite 3 differential signal D+.
	EBI_DATA10	I/O	MFP8	EBI data bus bit 10.
PF.6	PF.6	I/O	MFP0	General purpose digital I/O pin.
	RMII1_TXD1	O	MFP1	RMII1 Transmit Data bus bit 1.
	SD1_nCD	I	MFP2	SD/SDIO1 card detect input pin
	TM4_EXT	I/O	MFP3	Timer4 external capture input/toggle output pin.
	PWM01	O	MFP4	PWM01 counter synchronous trigger output pin.
	UART3_RXD	I	MFP5	UART3 data receiver input pin.
	USBHL4_DM	A	MFP6	USB 1.1 Host Lite 4 differential signal D-.
	EBI_DATA11	I/O	MFP8	EBI data bus bit 11.
PF.7	PF.7	I/O	MFP0	General purpose digital I/O pin.
	RMII1_TXD0	O	MFP1	RMII1 Transmit Data bus bit 0.
	UART1_CTS	I	MFP2	UART1 clear to Send input pin.
	TM5_EXT	I/O	MFP3	Timer5 external capture input/toggle output pin.
	PWM02	O	MFP4	PWM02 counter synchronous trigger output pin.
	UART3_TXD	O	MFP5	UART3 data transmitter output pin.
	USBHL4_DP	A	MFP6	USB 1.1 Host Lite 4 differential signal D+.
	EBI_DATA12	I/O	MFP8	EBI data bus bit 12.
PF.8	PF.8	I/O	MFP0	General purpose digital I/O pin.
	RMII1_MDIO	I/O	MFP1	RMII1 PHY Management Data pin.
	UART1_RTS	O	MFP2	UART1 request to Send output pin.
	TM1_TGL	I/O	MFP3	Timer1 event counter input/toggle output pin.

	Pin Name	Type	MFP	Description
	PWM03	O	MFP4	PWM03 counter synchronous trigger output pin.
	USBHL5_DM	A	MFP6	USB 1.1 Host Lite 5 differential signal D-.
	EBI_DATA13	I/O	MFP8	EBI data bus bit 13.
PF.9	PF.9	I/O	MFP0	General purpose digital I/O pin.
	RMII1_MDC	O	MFP1	RMII1 PHY Management Clock output pin.
	UART1_RXD	I	MFP2	UART1 data receiver input pin.
	TM1_EXT	I/O	MFP3	Timer1 external capture input/toggle output pin.
	PWM10	O	MFP4	PWM10 counter synchronous trigger output pin.
	USBHL5_DP	A	MFP6	USB 1.1 Host Lite 5 differential signal D+.
	EBI_DATA14	I/O	MFP8	EBI data bus bit 14.
PF.10	PF.10	I/O	MFP0	General purpose digital I/O pin.
	UART1_TXD	O	MFP2	UART1 data transmitter output pin.
	TM5_TGL	I/O	MFP3	Timer5 event counter input/toggle output pin.
	PWM11	O	MFP4	PWM11 counter synchronous trigger output pin.
	VCAP1_PCLK	I	MFP7	Video image interface 1 pixel clock pin.
	EBI_DATA15	I/O	MFP8	EBI data bus bit 15.
PF.11	PF.11	I/O	MFP0	General purpose digital I/O pin.
	UART0_RXD	I	MFP1	UART0 data receiver input pin.
PF.12	PF.12	I/O	MFP0	General purpose digital I/O pin.
	UART0_TXD	O	MFP1	UART0 data transmitter output pin.
PG.0	PG.0	I/O	MFP0	General purpose digital I/O pin.
	EBI_ADDR0	O	MFP1	EBI address bus bit 0.
	UART2_RXD	I	MFP2	UART2 data receiver input pin.
	CLK_OUT	O	MFP3	Internal clock selection output pin.
	PWM00	O	MFP6	PWM00 counter synchronous trigger output pin.
	CFG.0_PwrOnSet0	IU	-	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
PG.1	PG.1	I/O	MFP0	General purpose digital I/O pin.
	EBI_ADDR1	O	MFP1	EBI address bus bit 1.
	UART2_TXD	O	MFP2	UART2 data transmitter output pin.
	PWM01	O	MFP6	PWM01 counter synchronous trigger output pin.
	CFG.1_PwrOnSet1	IU	-	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.

	Pin Name	Type	MFP	Description
PG.2	PG.2	I/O	MFP0	General purpose digital I/O pin.
	EBI_ADDR2	O	MFP1	EBI address bus bit 2.
	UART2_CTS	I	MFP2	UART2 clear to Send input pin.
	PWM02	O	MFP6	PWM02 counter synchronous trigger output pin.
	CFG.2_PwrOnSet2	IU	-	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
PG.3	PG.3	I/O	MFP0	General purpose digital I/O pin.
	EBI_ADDR3	O	MFP1	EBI address bus bit 3.
	UART2_RTS	O	MFP2	UART2 request to Send output pin.
	PWM03	O	MFP6	PWM03 counter synchronous trigger output pin.
	CFG.3_PwrOnSet3	IU	-	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
PG.4	PG.4	I/O	MFP0	General purpose digital I/O pin.
	EBI_ADDR18	O	MFP1	EBI address bus bit 18.
	UART5_CTS	I	MFP2	UART5 clear to Send input pin.
	CFG.4_PwrOnSet4	IU	-	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
PG.5	PG.5	I/O	MFP0	General purpose digital I/O pin.
	EBI_ADDR12	O	MFP1	EBI address bus bit 12.
	UART5_RTS	O	MFP2	UART5 request to Send output pin.
	CFG.5_PwrOnSet5	IU	-	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
PG.6	PG.6	I/O	MFP0	General purpose digital I/O pin.
	EBI_ADDR4	O	MFP1	EBI address bus bit 4.
	UART5_RXD	I	MFP2	UART5 data receiver input pin.
	PWM10	O	MFP6	PWM10 counter synchronous trigger output pin.
	CFG.6_PwrOnSet6	IU	-	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
PG.7	PG.7	I/O	MFP0	General purpose digital I/O pin.
	EBI_ADDR5	O	MFP1	EBI address bus bit 5.
	UART5_TXD	O	MFP2	UART5 data transmitter output pin.
	PWM11	O	MFP6	PWM11 counter synchronous trigger output pin.
	CFG.7_PwrOnSet7	IU	-	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during

	Pin Name	Type	MFP	Description
				reset.
PG.8	PG.8	I/O	MFP0	General purpose digital I/O pin.
	EBI_ADDR6	O	MFP1	EBI address bus bit 6.
	UART8_RTS	O	MFP2	UART8 request to Send output pin.
	PWM12	O	MFP6	PWM12 counter synchronous trigger output pin.
	CFG.8_PwrOnSet8	IU	-	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
PG.9	PG.9	I/O	MFP0	General purpose digital I/O pin.
	EBI_ADDR7	O	MFP1	EBI address bus bit 7.
	UART8_CTS	I	MFP2	UART8 clear to Send input pin.
	PWM13	O	MFP6	PWM13 counter synchronous trigger output pin.
	CFG.9_PwrOnSet9	IU	-	System configuration and power on setting with an internal pull-up. Internal pull-up only active automatically during reset.
PG.10	PG.10	I/O	MFP0	General purpose digital I/O pin.
	EBI_DATA0	I/O	MFP1	EBI data bus bit 0.
	I2C0_SCL	I/O	MFP2	I^{2}C0 clock pin.
	TM5_TGL	I/O	MFP3	Timer5 event counter input/toggle output pin.
	USBHL4_DP	A	MFP4	USB 1.1 Host Lite 4 differential signal D+.
	CAN2_TXD	O	MFP5	CAN2 bus transmitter output.
	SPI1_CLK	I/O	MFP6	SPI1 serial clock pin.
	PWM00	O	MFP7	PWM00 counter synchronous trigger output pin.
	I2S_BCLK	O	MFP8	I^{2}S_ bit clock output pin.
PG.11	PG.11	I/O	MFP0	General purpose digital I/O pin.
	SPI1_SS0	I/O	MFP2	SPI1 slave select 0 pin.
	TM1_TGL	I/O	MFP3	Timer1 event counter input/toggle output pin.
	CAN0_RXD	I	MFP4	CAN0 bus receiver input.
	UART5_CTS	I	MFP5	UART5 clear to Send input pin.
	PWM10	O	MFP6	PWM10 counter synchronous trigger output pin.
	JTAG0_TDO	O	MFP7	JTAG0 data output pin.
PG.12	PG.12	I/O	MFP0	General purpose digital I/O pin.
	SPI1_CLK	I/O	MFP2	SPI1 serial clock pin.
	TM1_EXT	I/O	MFP3	Timer1 external capture input/toggle output pin.
	CAN0_TXD	O	MFP4	CAN0 bus transmitter output.

	Pin Name	Type	MFP	Description
	UART5_RTS	O	MFP5	UART5 request to Send output pin.
	PWM11	O	MFP6	PWM11 counter synchronous trigger output pin.
	JTAG0_TCK	I	MFP7	JTAG0 clock input pin.
PG.13	PG.13	I/O	MFP0	General purpose digital I/O pin.
	SPI1_MOSI	I/O	MFP2	SPI1 MOSI (Master Out, Slave In) pin.
	CAN1_RXD	I	MFP4	CAN1 bus receiver input.
	UART5_RXD	I	MFP5	UART5 data receiver input pin.
	PWM12	O	MFP6	PWM12 counter synchronous trigger output pin.
	JTAG0_TMS	I	MFP7	JTAG0 test mode selection input pin.
PG.14	PG.14	I/O	MFP0	General purpose digital I/O pin.
	SPI1_MISO	I/O	MFP2	SPI1 MISO (Master In, Slave Out) pin.
	CAN1_TXD	O	MFP4	CAN1 bus transmitter output.
	UART5_TXD	O	MFP5	UART5 data transmitter output pin.
	PWM13	O	MFP6	PWM13 counter synchronous trigger output pin.
	JTAG0_TDI	I	MFP7	JTAG0 data input pin.
PG.15	PG.15	I/O	MFP0	General purpose digital I/O pin.
	SPI0_SS1	I/O	MFP1	SPI0 slave select 1 pin.
	SPI1_SS1	I/O	MFP2	SPI1 slave select 1 pin.
	EINT3	I	MFP4	External interrupt 3 input pin.
	JTAG0_nTRST	I	MFP7	JTAG0 reset input pin.

5 BLOCK DIAGRAM

5.1 NUC980 Series Block Diagram

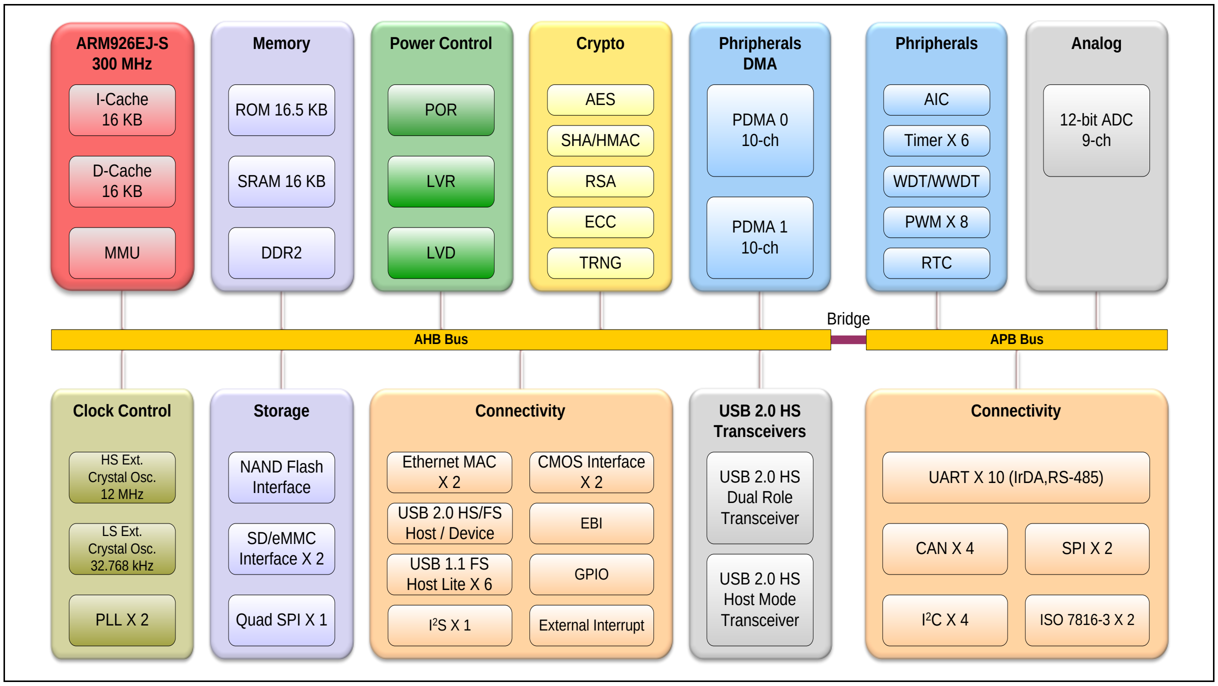


Figure 5.1-1 NUC980 Series Block Diagram

6 FUNCTIONAL DESCRIPTION

6.1 ARM926EJ-S™ Processor Core

6.1.1 Overview

The ARM926EJ-S™ processor core, a member of the ARM9 family general-purpose microprocessors, is targeted at multi-tasking applications where full memory management, high performance, and low power are all important.

The ARM926EJ-S™ processor core supports the 32-bit Arm® and 16-bit Thumb instruction sets, enabling the user to choose between high performance and high code density. The ARM926EJ-S™ core includes features for efficient execution of Java byte codes, providing Java performance similar to JIT, but without the associated code overhead.

The ARM926EJ-S™ processor provides support for external coprocessor enabling floating-point or other application-specific hardware acceleration to be added. The ARM926EJ-S™ processor core implements Arm® architecture version 5TEJ.

The ARM926EJ-S™ processor has a Harvard cached architecture and provides a complete high-performance processor subsystem, including:

- An ARM9EJ-S integer core.
- A Memory Management Unit (MMU).
- Separate instruction and data cache.
- Separate instruction and data AMBA AHB bus interfaces.

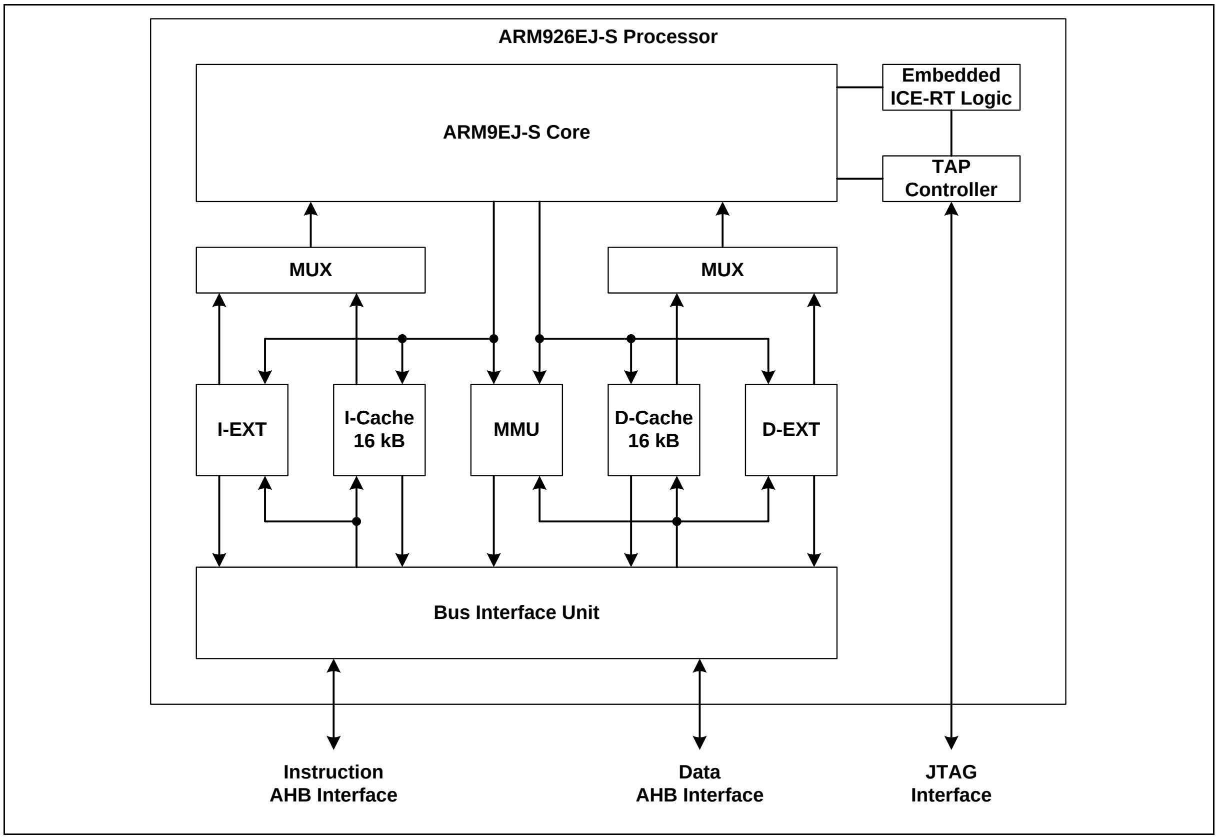


Figure 6.1-1 ARM926EJ-S™ Block Diagram

6.1.2 System Control Coprocessor (CP15)

The system control coprocessor (CP15) is used to configure and control the ARM926EJ-S™ processor. The caches, Memory Management Unit (MMU), and most other system options are controlled using CP15 registers. User can only access CP15 registers with MRC and MCR instruction in a privileged mode. Access CP15 registers with CDP, LDC, STC, MCRR, and MRRC instructions and unprivileged MRC or MCR instruction causes the undefined instruction exception to be taken.

6.1.3 Memory Management Unit (MMU)

The ARM926EJ-S™ MMU is an Arm® architecture v5 MMU. It provides virtual memory features required by systems operating on platforms such as Symbian OS, WindowsCE, and Linux. A single set of two-level page tables stored in main memory is used to control the address translation, permission checks, and memory region attributes for both data and instruction accesses.

The MMU uses a single unified Translation Lookaside Buffer (TLB) to cache the information held in the page tables. To support both sections and pages, there are two levels of address translation. The MMU puts the translated physical addresses into the MMU Translation Lookaside Buffer TLB.

The MMU TLB has two parts, the main TLB and the lockdown TLB. The main TLB is a two-way, set-associative cache for page table information. It has 32 entries per way for a total of 64 entries. The lockdown TLB is an eight-entry fully-associative cache that contains locked TLB entries. Locking TLB entries can ensure that a memory access to a given region never incurs the penalty of a page table walk.

The MMU features are:

- Standard Arm® architecture v4 and v5 MMU mapping sizes, domains, and access protection scheme
- Mapping sizes are 1MB (sections), 64KB (large pages), 4KB (small pages), and 1KB (tiny pages)
- Access permissions for large pages and small pages can be specified separately for each quarter of the page (subpage permissions)
- Hardware page table walks
- Invalidate entire TLB using CP15 c8
- Invalidate TLB entry selected by MVA, using CP15 c8
- Lockdown of TLB entries using CP15 c10.

6.1.4 Caches and Write Buffer

The ARM926EJ-S™ processor includes an Instruction Cache (I-Cache), a Data Cache (D-Cache) and a write buffer. The size of I-Cache and D-Cache in this chip is 16 KB, respectively.

The caches features are:

- The caches are virtual index, virtual tag, addressed using the Modified Virtual Address (MVA). This enables the avoidance of cache cleaning and/or invalidating on context switch.
- The caches are four-way set associative, with a cache line length of eight words per line (32 bytes per line), and with two dirty bits in the D-Cache.
- The D-Cache supports write-through and write-back (or copy back) cache operations, selected by memory region using the C and B bits in the MMU translation tables.
- Allocate on read-miss is supported. The caches perform critical-word first cache refilling.
- Pseudo-random or round-robin replacement selectable by the RR bit in CP15 c1.

- Cache lockdown registers enable control over which cache ways are used for allocation on a linefill, providing a mechanism for both lockdown and controlling cache pollution.
- The D-Cache stores the Physical Address (PA) tag corresponding to each D-Cache entry in the tag RAM for use during cache line write-backs, in addition to the Virtual Address tag stored in the tag RAM. This means that the MMU is not involved in D-Cache write-back operations, removing the possibility of TLB misses related to the write-back address.
- The PLD data preload instruction does not cause data cache linefills. It is treated as a NOP instruction.

6.1.5 Bus Interface Unit

The ARM926EJ-S™ Bus Interface Unit (BIU) arbitrates and schedules AHB requests. The BIU contains separate masters for both instruction and data access enabling complete AHB system flexibility. Each master is a fully compliant AHB bus master and implements the master functions as defined in the AMBA Specification (Rev 2.0).

To increase system performance, write buffers are used to prevent AHB writes stalling the ARM926EJ-S™ system.

6.1.6 Power Management

The ARM926EJ-S™ processor can be put into a low-power state by the wait for interrupt instruction:

MCR p15, 0, <Rd>, c7, c0, 4

This instruction switches the ARM926EJ-S™ processor into a low-power state until either an interrupt (IRQ or FIQ) or a debug request occurs.

In wait for interrupt mode, all internal ARM926EJ-S™ clocks are stopped. The switch into the low-power state is delayed until all write buffers have been drained, and the ARM926EJ-S™ memory system is in a quiescent state.

6.2 System Manager

6.2.1 Overview

The system management describes the following information and functions.

- System Resets
- System Power Architecture
- System Memory Map
- System management registers for Product Identifier (PDID), Power-On Setting, System Wake-Up, Reset Control for on-chip controllers/peripherals, and multi-function pin control.
- System Control registers

6.2.2 System Reset

The system reset can be issued by one of the following listed events. For these reset event flags can be read by RSTSTS register.

- Power-On Reset
- Low level on the /RESET pin
- Watchdog Time Out Reset
- Low Voltage Reset
- CPU Reset
- System Reset

6.2.3 System Power Distribution

In this chip, the power distribution is divided into six segments.

- Analog power from AV_{DD33} provides 3.3V voltage to analog components operation. These analog components including POR33, 12-bit SAR-ADC, LVD and LVR.
- Digital power from V_{DD12} provides 1.2V voltage to POR12, APLL, APLL, SRAM (16 kB) and all digital logic except RTC.
- Digital power from $VBAT_{33}$ provides 3.3V voltage to LXT and RTC logic.
- USB PHY power from V_{USB0_VDD33}, V_{USB0_VDD12} provides 3.3V and 1.2 respectively to USB 2.0 PHY 0, while V_{USB1_VDD33}, V_{USB1_VDD12} provides 3.3V and 1.2 respectively to USB 2.0 PHY 1.
- I/O power from MV_{DD} provides 1.8V/2.5V to I/O pins used to connect SDRAM.
- I/O power from V_{DD33} provides 3.3V to HXT and I/O pins (PA ~ PG).

Figure 6.2-1 shows the power distribution of the NUC980 series.

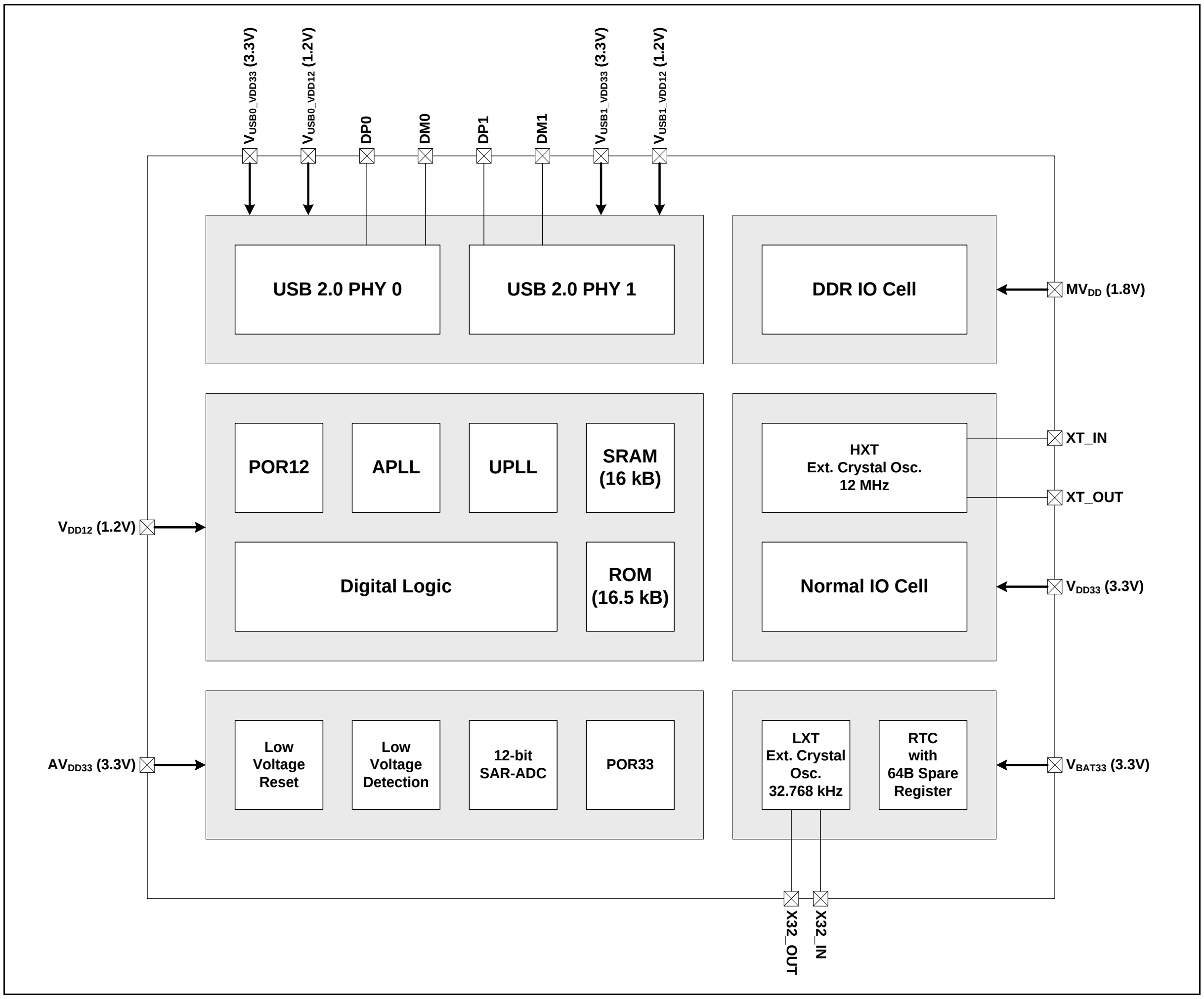


Figure 6.2-1 NUC980 Series Power Distribution Diagram

6.2.4 System Memory Map

This chip supports only little-endian data format and provides 4G-byte addressing space. Figure 6.2-2 describes the memory space definition.

The memory space from 0x0000_0000 to 0x1FFF_FFFF is for SDRAM and external devices. The memory space from 0x3C00_0000 to 0x3C00_3FFF is for embedded 16 Kbytes SRAM. The memory space for On-Chip Controllers and Peripherals is from 0xB000_0000 to 0xB00A_3FFF while the memory space from 0xFFFF_0000 to 0xFFFF_41FF is for 16.5 Kbytes internal Boot ROM.

This chip provides the shadow memory function. The memory space from 0x8000_0000 to 0x9FFF_FFFF is the shadow memory space for memory space from 0x0000_0000 to 0x1FFF_FFFF. The memory space from 0xBC00_0000 to 0xBC00_3FFF is the shadow memory space for memory space from 0x3C00_0000 to 0x3C00_3FFF. If the DMA of On-Chip Controller wants to access this 16 Kbytes embedded SRAM, it’s necessary to use memory space from 0xBC00_0000 to 0xBC00_3FFF

The reserved memory space is un-accessible. Chip’s behavior is undefined and unpredictable while accessing to reserved memory space.

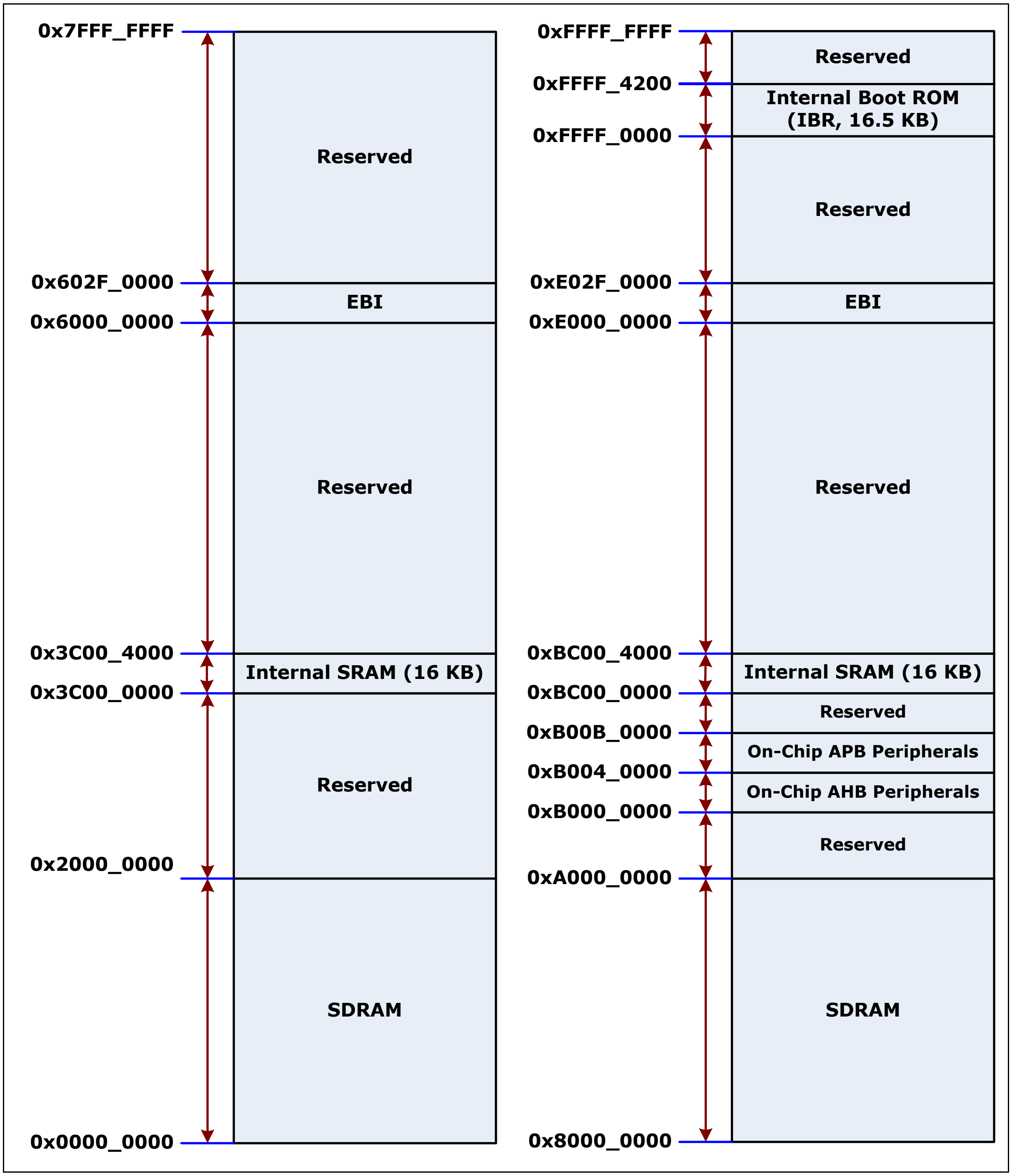


Figure 6.2-2 NUC980 System Memory Map Diagram

The addressing space assigned to each on-chip controller or peripheral described in Table 6.2-1. The detailed register definition, addressing space, and programming details will be described in the following sections.

Addressing Space	Token	Modules
SDRAM, External Devices and SRAM Memory Space		
0x0000_0000 – 0x1FFF_FFFF	SDRAM_BA	SDRAM Memory Space
0x6000_0000 – 0x602F_FFFF	EXDEV_BA	External Devices Memory Space
0x3C00_0000 – 0x3C00_3FFF	SRAM_BA	SRAM Memory Space (16 KB)
Internal Boot ROM (IBR) Memory Space (0xFFFF_0000 ~ 0xFFFF_41FF)		
0xFFFF_0000 – 0xFFFF_41FF	IBR_BA	Internal Boot ROM (IBR) Memory Space (16.5 KB)
AHB Modules Memory Space (0xB000_0000 – 0xB003_FFFF)		
0xB000_0000 – 0xB000_01FF	SYS_BA	System Global Control Registers
0xB000_0200 – 0xB000_02FF	CLK_BA	Clock Control Registers
0xB000_2000 – 0xB000_2FFF	SDIC_BA	SDRAM (SDR/DDR/DDR2) Control Registers
0xB000_4000 – 0xB000_4FFF	GPIO_BA	GPIO Control Registers
0xB000_8000 – 0xB000_8FFF	PDMA0_BA	PDMA 0 Control Registers
0xB000_9000 – 0xB000_9FFF	PDMA1_BA	PDMA 1 Control Registers
0xB001_0000 – 0xB001_0FFF	EBI_BA	EBI Control Registers
0xB001_2000 – 0xB001_2FFF	EMAC0_BA	Ethernet MAC 0 Control Registers
0xB002_4000 – 0xB002_4FFF	CAP0_BA	Capture Sensor Interface 0 Control Registers
0xB001_5000 – 0xB001_5FFF	HSUSBH_BA	High Speed USB 2.0 Host Control Registers
0xB001_6000 – 0xB001_6FFF	HSUSBD_BA	High Speed USB 2.0 Device Control Registers
0xB001_7000 – 0xB001_7FFF	USBH_BA	USB 2.0 Host Control Registers
0xB001_8000 – 0xB001_8FFF	SDH_BA	SD/SDIO Host Control Registers
0xB001_9000 – 0xB001_9FFF	FMI_BA	Flash Memory Interface (FMI) Control Registers
0xB001_C000 – 0xB001_EFFF	CRYPTO_BA	Cryptographic Accelerator Control Registers
0xB002_0000 – 0xB002_0FFF	I2S_BA	I^{2}S Interface Control Registers
0xB002_2000 – 0xB002_2FFF	EMAC1_BA	Ethernet MAC 1 Control Registers
0xB001_4000 – 0xB001_4FFF	CAP1_BA	Capture Sensor Interface 1 Control Registers
APB Modules Memory Space (0xB004_0000 ~ 0xB00A_FFFF)		
0xB004_0000 – 0xB004_00FF	WDT_BA	Watch-Dog Timer Control Registers
0xB004_0100 – 0xB004_01FF	WWDT_BA	Windowed Watch-Dog Timer Control Registers
0xB004_1000 – 0xB004_1FFF	RTC_BA	Real Time Clock (RTC) Control Registers
0xB004_2000 – 0xB004_2FFF	AIC_BA	Advance Interrupt Control Registers
0xB004_3000 – 0xB004_3FFF	ADC_BA	ADC Control Registers

0xB005_0000 – 0xB005_0FFF	TMR_BA01	Timer 0 and Timer 1 Control Registers
0xB005_1000 – 0xB005_1FFF	TMR_BA23	Timer 2 and Timer 3 Control Registers
0xB005_2000 – 0xB005_2FFF	TMR_BA45	Timer 4 and Timer 5 Control Registers
0xB005_8000 – 0xB005_8FFF	PWM0_BA	PWM 0 Control Registers
0xB005_9000 – 0xB005_9FFF	PWM1_BA	PWM 1 Control Registers
0xB006_0000 – 0xB006_0FFF	QSPI0_BA	QSPI 0 Control Registers
0xB006_1000 – 0xB006_1FFF	SPI0_BA	SPI 0 Control Registers
0xB006_2000 – 0xB006_2FFF	SPI1_BA	SPI 1 Control Registers
0xB007_0000 – 0xB007_0FFF	UART0_BA	UART 0 Control Registers
0xB007_1000 – 0xB007_1FFF	UART1_BA	UART 1 Control Registers
0xB007_2000 – 0xB007_2FFF	UART2_BA	UART 2 Control Registers
0xB007_3000 – 0xB007_3FFF	UART3_BA	UART 3 Control Registers
0xB007_4000 – 0xB007_4FFF	UART4_BA	UART 4 Control Registers
0xB007_5000 – 0xB007_5FFF	UART5_BA	UART 5 Control Registers
0xB007_6000 – 0xB007_6FFF	UART6_BA	UART 6 Control Registers
0xB007_7000 – 0xB007_7FFF	UART7_BA	UART 7 Control Registers
0xB007_8000 – 0xB007_8FFF	UART8_BA	UART 8 Control Registers
0xB007_9000 – 0xB007_9FFF	UART9_BA	UART 9 Control Registers
0xB008_0000 – 0xB008_0FFF	I2C0_BA	I^{2}C 0 Control Registers
0xB008_1000 – 0xB008_1FFF	I2C1_BA	I^{2}C 1 Control Registers
0xB008_2000 – 0xB008_2FFF	I2C2_BA	I^{2}C 2 Control Registers
0xB008_3000 – 0xB008_3FFF	I2C3_BA	I^{2}C 3 Control Registers
0xB009_0000 – 0xB009_0FFF	SC0_BA	Smart Card 0 Control Registers
0xB009_1000 – 0xB009_1FFF	SC1_BA	Smart Card 1 Control Registers
0xB00A_0000 – 0xB00A_0FFF	CAN0_BA	CAN 0 Control Registers
0xB00A_1000 – 0xB00A_1FFF	CAN1_BA	CAN 1 Control Registers
0xB00A_2000 – 0xB00A_2FFF	CAN2_BA	CAN 2 Control Registers
0xB00A_3000 – 0xB00A_3FFF	CAN3_BA	CAN 3 Control Registers

Table 6.2-1 Address Space Assignments for On-Chip Controllers

6.2.5 Power-On Setting

After power on reset, Power-On setting registers are latched to configure this chip. Table 6.2-2 describes the definition of each power-on setting bit.

Power-On Setting Pin	Description	Power-on Setting Register Bit
USB0_ID	USB Port 0 Role Selection 0 = USB Port 0 act as a USB host. 1 = USB Port 0 act as a USB device.	USBID (SYS_PWRON[16])
PG[1:0]	Boot Source Selection 00 = Boot from USB. 01 = Boor from SD0/eMMC. 10 = Boot from NAND Flash. 11 = Boot from SPI Flash.	BTSSEL (SYS_PWRON[1:0])
PG.2	QSPI0_CLK Frequency Selection 0 = QSPI0_CLK frequency is 30 MHz. 1 = QSPI0_CLK frequency is 50 MHz.	QSPI0CKSEL (SYS_PWRON[2])
PG.3	Watchdog Timer (WDT) Enabled/Disabled Selection 0 = After power-on, WDT Disabled. 1 = after power-on WDT Enabled.	WDTON (SYS_PWRON[3])
PG.4	JTAG Interface Selection 0 = Pin PA[6:2] used as JTAG interface. 1 = Pin PG[15:11] used as JTAG interface.	JTAGSEL (SYS_PWRON[4])
PG.5	UART 0 Debug Message Output ON/OFF Selection 0 = UART 0 debug message output ON and pin PF[12:11] used as the UART0 functionality. 1 = UART 0 debug message output OFF and pin PF[12:11] used as the GPIO functionality.	URDBGON (SYS_PWRON[5])
PG[7:6]	NAND Flash Page Size selection 00 = NAND Flash page size is 2KB. 01 = NAND Flash page size is 4KB. 10 = NAND Flash page size is 8KB. 11 = Ignore Power-On Setting.	NPAGESEL (SYS_PWRON[7:6])

PG[9:8]	Miscellaneous Configuration When BTSSEL = 01, Boot from SD/eMMC, the MISCCFG defines the GPC or GPF used as the booting source. 11 = GPC group used as the booting source. Others = GPF group used as the booting source. When BTSSEL = 10, Boot from NAND Flash, the MISCCFG defines the ECC type. 00 = No ECC 01 = ECC is BCH T12 10 = ECC is BCH T24 11 = Ignore power-on setting When BTSEL = 11, Boot from SPI Flash, the MISCCFG defines the SPI Flash type and data width. 00 = SPI-NAND Flash with 1-bit mode. 01 = SPI-NAND Flash with 4-bit mode. 10 = SPI-NOR Flash with 4-bit mode. 11 = SPI-NOR Flash with 1-bit mode.	MISCCFG (SYS_PWRON[9:8])

Table 6.2-2 Power-On Setting Bit Description

6.2.6 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 = 0xB000_0000**				
SYS_PDID	SYS_BA+0x000	R	Product Identifier Register	0x1030_D016[1]
SYS_PWRON	SYS_BA+0x004	R/W	Power-on Setting Register	0xXXXX_XXXX[2]
SYS_LVRDCR	SYS_BA+0x020	R/W	Low Voltage Reset & Detect Control Register	0x0000_0001
SYS_MISCFCR	SYS_BA+0x030	R/W	Miscellaneous Function Control Register	0x0000_0200
SYS_MISCIER	SYS_BA+0x040	R/W	Miscellaneous Interrupt Enable Register	0x0000_0000
SYS_MISCISR	SYS_BA+0x044	R/W	Miscellaneous Interrupt Status Register	0x0001_0000
SYS_WKUPSER0	SYS_BA+0x050	R/W	System Wakeup Source Enable Register 0	0x0000_0000
SYS_WKUPSER1	SYS_BA+0x054	R/W	System Wakeup Source Enable Register 1	0x0000_0000
SYS_WKUPSSR0	SYS_BA+0x058	R/W	System Wakeup Source Status Register 0	0x0000_0000
SYS_WKUPSSR1	SYS_BA+0x05C	R/W	System Wakeup Source Status Register 1	0x0000_0000
SYS_AHBIPRST	SYS_BA+0x060	R/W	AHB IP Reset Control Register	0x0000_0000
SYS_APBIPRST0	SYS_BA+0x064	R/W	APB IP Reset Control Register 0	0x0000_0000
SYS_APBIPRST1	SYS_BA+0x068	R/W	APB IP Reset Control Register 1	0x0000_0000
SYS_RSTSTS	SYS_BA+0x06C	R/W	Reset Source Active Status Register	0x0000_00XX
SYS_GPA_MFPL	SYS_BA+0x070	R/W	GPIOA Low Byte Multiple Function Control Register	0x0XXX_XX00
SYS_GPA_MFPH	SYS_BA+0x074	R/W	GPIOA High Byte Multiple Function Control Register	0x0000_0000
SYS_GPB_MFPL	SYS_BA+0x078	R/W	GPIOB Low Byte Multiple Function Control Register	0x0000_0000
SYS_GPB_MFPH	SYS_BA+0x07C	R/W	GPIOB High Byte Multiple Function Control Register	0x0000_0000
SYS_GPC_MFPL	SYS_BA+0x080	R/W	GPIOC Low Byte Multiple Function Control Register	0x0000_0000
SYS_GPC_MFPH	SYS_BA+0x084	R/W	GPIOC High Byte Multiple Function Control Register	0x0000_0000
SYS_GPD_MFPL	SYS_BA+0x088	R/W	GPIOD Low Byte Multiple Function Control Register	0x0000_0000
SYS_GPD_MFPH	SYS_BA+0x08C	R/W	GPIOD High Byte Multiple Function Control Register	0x0000_0000
SYS_GPE_MFPL	SYS_BA+0x090	R/W	GPIOE Low Byte Multiple Function Control Register	0x0000_0000
SYS_GPE_MFPH	SYS_BA+0x094	R/W	GPIOE High Byte Multiple Function Control Register	0x0000_0000
SYS_GPF_MFPL	SYS_BA+0x098	R/W	GPIOF Low Byte Multiple Function Control Register	0x0000_0000

SYS_GPF_MFPH	SYS_BA+0x09C	R/W	GPIOF High Byte Multiple Function Control Register	0x0000_0000
SYS_GPG_MFPL	SYS_BA+0x0A0	R/W	GPIOG Low Byte Multiple Function Control Register	0x0000_0000
SYS_GPG_MFPH	SYS_BA+0x0A4	R/W	GPIOG High Byte Multiple Function Control Register	0xXXXX_X000
SYS_DDR_DSCTL	SYS_BA+0x0F0	R/W	DDR I/O Driving Strength Control Register	0x0000_0000
SYS_GPBL_DSCTL	SYS_BA+0x0F4	R/W	GPIOB Low Byte Driving Strength Control Register	0x4444_4444
SYS_PORDISCR	SYS_BA+0x100	R/W	Power-On-reset Disable Control Register	0x0000_00XX
SYS_RSTDEBCTL	SYS_BA+0x10C	R/W	Reset Pin De-bounce Control Register	0x0000_04B0
SYS_REGWPCTL	SYS_BA+0x1FC	R/W	Register Write-protection Control Register	0x0000_0000

Note: [1] Dependents on part number.

Note: [2] Dependents on power-on setting.

6.2.7 Register Description

Product Identifier Register (SYS_PDID)

Register	Offset	R/W	Description	Reset Value
SYS_PDID	SYS_BA+0x000	R	Product Identifier Register	0x1030_D016

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PRDNUML6				PRDNUML5			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PRDNUML4				PRDNUML3			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PRDNUML2				PRDNUML1			

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:20]	**PRDNUML6**	**Product Number Letter 6** 0 = D. 1 = F. 2 = G. 3 = H.
[19:16]	**PRDNUML5**	**Product Number Letter 5** 0 = A. 1 = B.
[15:12]	**PRDNUML4**	**Product Number Letter 4** 0xD
[11:8]	**PRDNUML3**	**Product Number Letter 3** 0x0
[7:4]	**PRDNUML2**	**Product Number Letter 2** 0x1
[3:0]	**PRDNUML1**	**Product Number Letter 1** 0x6

Power-on Setting Register (SYS_PWRON)

Register	Offset	R/W	Description	Reset Value
SYS_PWRON	SYS_BA+0x004	R/W	Power-on Setting Register	0xXXXX_XXXX

31	30	29	28	27	26	25	24
Reserved			USERID				
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	DRAMSIZE			Reserved		TICMOD	USBID
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						MISCCFG	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NPAGESEL		URDBGON	JTAGSEL	WDTON	QSPI0CKSEL	BTSSEL	

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:24]	**USERID**	**User ID (Read Only)** A user defined ID.
[23]	**Reserved**	Reserved.
[22:20]	**DRAMSIZE**	**DRAM Size** 000 = Reserved. 001 = 2 MB 010 = Reserved. 011 = 8 MB 100 = 16 MB 101 = 32 MB 110 = 64 MB 111 = 128 MB
[19:18]	**Reserved**	Reserved.
[17]	**TICMOD**	**TIC Mode Enable Bit** 0= TIC interface Disabled. 1= TIC interface Enabled.
[16]	**USBID**	**USB ID Pin Status** 0= USB port 0 used as a USB device. 1= USB port 0 used as a USB host.

[9:8]	**MISCCFG**	**Miscellaneous Configuration** When pin nRESET transited from low to high, the value of pin PG[9:8] latched to MISCCFG. When BTSSEL = 01, Boot from SD/eMMC, the MISCCFG defines the SD0/eMMC0 or SD1/eMMC1 used as the booting source. 11 = SD0/eMMC0 (GPC group) used as the booting source. Others = SD1/eMMC1 (GPF group) used as the booting source. When BTSSEL = 10, Boot from NAND Flash, the MISCCFG defines the ECC type. 00 = ECC is BCH T8. 01 = ECC is BCH T12. 10 = ECC is BCH T24. 11 = Ignore power-on setting. When BTSEL = 11, Boot from SPI Flash, the MISCCFG defines the SPI Flash type and data width. 00 = SPI-NAND Flash with 1-bit mode. 01 = SPI-NAND Flash with 4-bit mode. 10 = SPI-NOR Flash with 4-bit mode. 11 = SPI-NOR Flash with 1-bit mode.
[7:6]	**NPAGESEL**	**NAND Flash Page Size Selection** When pin nRESET transited from low to high, the value of pin PG[7:6] latched to NPAGESEL. 00= NAND Flash page size is 2KB. 01= NAND Flash page size is 4KB. 10= NAND Flash page size is 8KB. 11= Ignore power-on setting.
[5]	**URDBGON**	**UART 0 Debug Message Output ON/OFF Selection** When pin nRESET transited from low to high, the value of pin PG.5 latched to URDBGON. 0= UART 0 debug message output ON. 1= UART 0 debug message output OFF.
[4]	**JTAGSEL**	**JTAG Interface Selection** When pin nRESET transited from low to high, the value of pin PG.4 latched to JTAGSEL. 0 = Pin PA[6:2] used as JTAG interface. 1 = Pin PG[15:11] used as JTAG interface.
[3]	**WDTON**	**Watchdog Timer (WDT) ON/OFF Selection** When pin nRESET transited from low to high, the value of pin PG.3 latched to WDTON. 0 = After power-on, WDT Disabled. 1 = after power-on WDT Enabled.
[2]	**QSPI0CKSEL**	**QSPI0_CLK Frequency Selection** When pin nRESET transited from low to high, the value of pin PG.2 latched to QSPI0CKSEL. 0 = QSPI0_CLK frequency is 37.5 MHz. 1 = QSPI0_CLK frequency is 75 MHz.

[1:0]	**BTSSEL**	**Boot Source Selection** When pin nRESET transited from low to high, the value of pin PG[1:0] latched to BTSSEL. 00= Boot from USB. 01= Boot from SD/eMMC. 10= Boot from NAND Flash. 11= Boot from SPI Flash.

Low Voltage Reset & Detect Control Register (SYS_LVRDCR)

Register	Offset	R/W	Description	Reset Value
SYS_LVRDCR	SYS_BA+0x020	R/W	Low Voltage Reset & Detect 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						**LVD_SEL**	**LVD_EN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**LVR_EN**

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**LVD_SEL**	**Low Voltage Detect Threshold Selection** 0 = Low voltage detection level is 2.6V. 1 = Low voltage detection level is 2.8V.
[8]	**LVD_EN**	**Low Voltage Detect Enable Bit** 0 = Low voltage detect function Disabled. 1 = Low voltage detect function Enabled.
[7:1]	**Reserved**	Reserved.
[0]	**LVR_EN**	**Low Voltage Reset Enable Bit** 0 = Low voltage reset function Disabled. 1 = Low voltage reset function Enabled.

NUC980 SERIES TECHNICAL REFERENCE MANUAL

Miscellaneous Function Control Register (SYS_MISCFCR)

Register	Offset	R/W	Description	Reset Value
SYS_MISCFCR	SYS_BA+0x030	R/W	Miscellaneous Function 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			GPIOLBEN	USRHDSEN	Reserved	HDSPUEN	WDTRSTEN
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:13]	**Reserved**	Reserved.
[12]	**GPIOLBEN**	**GPIO Pin Loop-back Enable Bit** 0 = GPIO input status didn’t reflect pin status if the GPIO configured as functional pin. 1 = GPIO input status did reflect pin status even if the GPIO configured as functional pin.
[11]	**USRHDSEN**	**User Configurable USB Host Device Role Selection Enable Bit** 0 = USB host/device role selection decided by HDS pin. 1 = USB host/device role selection decided by USBID (SYS_PWRON[16]).
[10]	**Reserved**	Reserved.
[9]	**HDSPUEN**	**HDS Pin Internal Pull-up Enable Bit** 0 = HDS pin internal pull-up resister Disabled. 1 = HDS pin internal pull-up resister Enabled.
[8]	**WDTRSTEN**	**WatchDog Timer Reset Connection Enable Bit** This bit is used to enable the function that connect watch-dog timer reset to nRESET pin. If this bit is enabled, the watch-dog timer reset is connected to nRESET pin internally 0 = Watch-dog timer reset not connected to nRESET pin internally. 1 = Watch-dog timer reset connected to nRESET pin internally.
[7:0]	**Reserved**	Reserved.

Miscellaneous Interrupt Enable Register (SYS_MISCIER)

Register	Offset	R/W	Description	Reset Value
SYS_MISCIER	SYS_BA+0x040	R/W	Miscellaneous 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						**USBIDC_IEN**	**LVD_IEN**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**USBIDC_IEN**	**USB0_ID Pin Status Change Interrupt Enable Bit** 0 = HDS status change interrupt Disabled. 1 = HDS status change interrupt Enabled.
[0]	**LVD_IEN**	**Low Voltage Detect Interrupt Enable Bit** 0 = Low voltage detect interrupt Disabled. 1 = Low voltage detect interrupt Enabled.

Miscellaneous Interrupt Status Register (SYS_MISCISR)

Register	Offset	R/W	Description	Reset Value
SYS_MISCISR	SYS_BA+0x044	R/W	Miscellaneous Interrupt Status Register	0x0001_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						USB0_IDS	IBR_RUN_F
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						USBIDC_IS	LVD_IS

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**USB0_IDS**	**USB0_ID Status** 0 = USB port 0 used as a USB device port. 1 = USB port 0 used as a USB host port.
[16]	**IBR_RUN_F**	**IBR Run Flag** 0 = CPU didn't execute instruction in 0xFFFF_0000 yet. 1 = CPU executed instruction in 0xFFFF_0000.
[15:2]	**Reserved**	Reserved.
[1]	**USBIDC_IS**	**USB0_ID Pin State Change Interrupt Status** 0 = USB0_ID state didn't change. 1 = USB0_ID state changed from low to high or from high to low.
[0]	**LVD_IS**	**Low Voltage Detect Interrupt Status** 0 = No low voltage event. 1 = Low voltage event detected.

System Wakeup Source Enable Register 0 (SYS_WKUPSER0)

Register	Offset	R/W	Description	Reset Value
SYS_WKUPSER0	SYS_BA+0x050	R/W	System Wakeup Source Enable Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						UR9WKEN	UR8WKEN
23	22	21	20	19	18	17	16
UR7WKEN	UR6WKEN	UR5WKEN	UR4WKEN	UR3WKEN	UR2WKEN	UR1WKEN	UR0WKEN
15	14	13	12	11	10	9	8
Reserved		TMR5WKEN	TMR4WKEN	TMR3WKEN	TMR2WKEN	TMR1WKEN	TMR0WKEN
7	6	5	4	3	2	1	0
EINT3WKEN	EINT2WKEN	EINT1WKEN	EINT0WKEN	GPIOWKEN	Reserved	Reserved	WDTWKEN

Bits	Description	
[31:26]	Reserved	Reserved.
[25]	UR9WKEN	**UART 9 Wake System Up Enable Bit** 0 = UART 9 wake system up function Disabled. 1 = UART 9 wake system up function Enabled.
[24]	UR8WKEN	**UART 8 Wake System Up Enable Bit** 0 = UART 8 wake system up function Disabled. 1 = UART 8 wake system up function Enabled.
[23]	UR7WKEN	**UART 7 Wake System Up Enable Bit** 0 = UART 7 wake system up function Disabled. 1 = UART 7 wake system up function Enabled.
[22]	UR6WKEN	**UART 6 Wake System Up Enable Bit** 0 = UART 6 wake system up function Disabled. 1 = UART 6 wake system up function Enabled.
[21]	UR5WKEN	**UART 5 Wake System Up Enable Bit** 0 = UART 5 wake system up function Disabled. 1 = UART 5 wake system up function Enabled.
[20]	UR4WKEN	**UART 4 Wake System Up Enable Bit** 0 = UART 4 wake system up function Disabled. 1 = UART 4 wake system up function Enabled.
[19]	UR3WKEN	**UART 3 Wake System Up Enable Bit** 0 = UART 3 wake system up function Disabled. 1 = UART 3 wake system up function Enabled.

[18]	**UR2WKEN**	**UART 2 Wake System Up Enable Bit** 0 = UART 2 wake system up function Disabled. 1 = UART 2 wake system up function Enabled.
[17]	**UR1WKEN**	**UART 1 Wake System Up Enable Bit** 0 = UART 1 wake system up function Disabled. 1 = UART 1 wake system up function Enabled.
[16]	**UR0WKEN**	**UART 0 Wake System Up Enable Bit** 0 = UART 0 wake system up function Disabled. 1 = UART 0 wake system up function Enabled.
[15:14]	**Reserved**	Reserved.
[13]	**TMR5WKEN**	**TIMER 5 Wake System Up Enable Bit** 0 = TIMER 5 wake system up function Disabled. 1 = TIMER 5 wake system up function Enabled.
[12]	**TMR4WKEN**	**TIMER 4 Wake System Up Enable Bit** 0 = TIMER 4 wake system up function Disabled. 1 = TIMER 4 wake system up function Enabled.
[11]	**TMR3WKEN**	**TIMER 3 Wake System Up Enable Bit** 0 = TIMER 3 wake system up function Disabled. 1 = TIMER 3 wake system up function Enabled.
[10]	**TMR2WKEN**	**TIMER 2 Wake System Up Enable Bit** 0 = TIMER 2 wake system up function Disabled. 1 = TIMER 2 wake system up function Enabled.
[9]	**TMR1WKEN**	**TIMER 1 Wake System Up Enable Bit** 0 = TIMER 1 wake system up function Disabled. 1 = TIMER 1 wake system up function Enabled.
[8]	**TMR0WKEN**	**TIMER 0 Wake System Up Enable Bit** 0 = TIMER 0 wake system up function Disabled. 1 = TIMER 0 wake system up function Enabled.
[7]	**EINT3WKEN**	**External Interrupt 3 Wake System Up Enable Bit** 0 = External Interrupt 3 wake system up function Disabled. 1 = External Interrupt 3 wake system up function Enabled.
[6]	**EINT2WKEN**	**External Interrupt 2 Wake System Up Enable Bit** 0 = External Interrupt 2 wake system up function Disabled. 1 = External Interrupt 2 wake system up function Enabled.
[5]	**EINT1WKEN**	**External Interrupt 1 Wake System Up Enable Bit** 0 = External Interrupt 1 wake system up function Disabled. 1 = External Interrupt 1 wake system up function Enabled.
[4]	**EINT0WKEN**	**External Interrupt 0 Wake System Up Enable Bit** 0 = External Interrupt 0 wake system up function Disabled. 1 = External Interrupt 0 wake system up function Enabled.

[3]	**GPIOWKEN**	**GPIO Wake System Up Enable Bit** 0 = GPIO wake system up function Disabled. 1 = GPIO wake system up function Enabled.
[2:1]	**Reserved**	Reserved.
[0]	**WDTWKEN**	**WDT Wake System Up Enable Bit** 0 = WDT wake system up function Disabled. 1 = WDT wake system up function Enabled.

System Wakeup Source Enable Register 1 (SYS_WKUPSER1)

Register	Offset	R/W	Description	Reset Value
SYS_WKUPSER1	SYS_BA+0x054	R/W	System Wakeup Source Enable Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			SDHWKEN	USBDWKEN	USBHWKEN	EMAC1WKEN	EMAC0WKEN
15	14	13	12	11	10	9	8
LVDWKEN	Reserved			CAN3WKEN	CAN2WKEN	CAN1WKEN	CAN0WKEN
7	6	5	4	3	2	1	0
RTCWKEN	Reserved			I2C3WKEN	I2C2WKEN	I2C1WKEN	I2C0WKEN

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20]	**SDHWKEN**	**SDH Wake System Up Enable Bit** 0 = SDH wake system up function Disabled. 1 = SDH wake system up function Enabled.
[19]	**USBDWKEN**	**USB Device Wake System Up Enable Bit** 0 = USB device wake system up function Disabled. 1 = USB device wake system up function Enabled.
[18]	**USBHWKEN**	**USB Host Wake System Up Enable Bit** 0 = USB host wake system up function Disabled. 1 = USB host wake system up function Enabled.
[17]	**EMAC1WKEN**	**Ethernet MAC 1 Wake System Up Enable Bit** 0 = Ethernet MAC 1 wake system up function Disabled. 1 = Ethernet MAC 1 wake system up function Enabled.
[16]	**EMAC0WKEN**	**Ethernet MAC 0 Wake System Up Enable Bit** 0 = Ethernet MAC 0 wake system up function Disabled. 1 = Ethernet MAC 0 wake system up function Enabled.
[15]	**LVDWKEN**	**Low Voltage Detect Wake System Up Enable Bit** 0 = Low Voltage Detect wake system up function Disabled. 1 = Low Voltage Detect wake system up function Enabled.
[14:12]	**Reserved**	Reserved.
[11]	**CAN3WKEN**	**CAN 3 Wake System Up Enable Bit** 0 = CAN 3 wake system up function Disabled. 1 = CAN 3 wake system up function Enabled.

[10]	**CAN2WKEN**	**CAN 2 Wake System Up Enable Bit** 0 = CAN 2 wake system up function Disabled. 1 = CAN 2 wake system up function Enabled.
[9]	**CAN1WKEN**	**CAN 1 Wake System Up Enable Bit** 0 = CAN 1 wake system up function Disabled. 1 = CAN 1 wake system up function Enabled.
[8]	**CAN0WKEN**	**CAN 0 Wake System Up Enable Bit** 0 = CAN 0 wake system up function Disabled. 1 = CAN 0 wake system up function Enabled.
[7]	**RTCWKEN**	**RTC Wake System Up Enable Bit** 0 = RTC wake system up function Disabled. 1 = RTC wake system up function Enabled.
[6:4]	**Reserved**	Reserved.
[3]	**I2C3WKEN**	**I^{2}C 3 Wake System Up Enable Bit** 0 = I^{2}C 3 wake system up function Disabled. 1 = I^{2}C 3 wake system up function Enabled.
[2]	**I2C2WKEN**	**I^{2}C 2 Wake System Up Enable Bit** 0 = I^{2}C 2 wake system up function Disabled. 1 = I^{2}C 2 wake system up function Enabled.
[1]	**I2C1WKEN**	**I^{2}C 1 Wake System Up Enable Bit** 0 = I^{2}C 1 wake system up function Disabled. 1 = I^{2}C 1 wake system up function Enabled.
[0]	**I2C0WKEN**	**I^{2}C 0 Wake System Up Enable Bit** 0 = I^{2}C 0 wake system up function Disabled. 1 = I^{2}C 0 wake system up function Enabled.

System Wakeup Source Status Register 0 (SYS_WKUPSSR0)

Register	Offset	R/W	Description	Reset Value
SYS_WKUPSSR0	SYS_BA+0x058	R/W	System Wakeup Source Status Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						UR9WKST	UR8WKST
23	22	21	20	19	18	17	16
UR7WKST	UR6WKST	UR5WKST	UR4WKST	UR3WKST	UR2WKST	UR1WKST	UR0WKST
15	14	13	12	11	10	9	8
Reserved		TMR5WKST	TMR4WKST	TMR3WKST	TMR2WKST	TMR1WKST	TMR0WKST
7	6	5	4	3	2	1	0
EINT3WKST	EINT2WKST	EINT1WKST	EINT0WKST	GPIOWKST	Reserved	Reserved	WDTWKST

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25]	**UR9WKST**	**UART 9 Wake System Up Status** 0 = UART 9 didn't wake system up. 1 = UART 9 wake system up.
[24]	**UR8WKST**	**UART 8 Wake System Up Status** 0 = UART 8 didn't wake system up. 1 = UART 8 wake system up.
[23]	**UR7WKST**	**UART 7 Wake System Up Status** 0 = UART 7 didn't wake system up. 1 = UART 7 wake system up.
[22]	**UR6WKST**	**UART 6 Wake System Up Status** 0 = UART 6 didn't wake system up. 1 = UART 6 wake system up.
[21]	**UR5WKST**	**UART 5 Wake System Up Status** 0 = UART 5 didn't wake system up. 1 = UART 5 wake system up.
[20]	**UR4WKST**	**UART 4 Wake System Up Status** 0 = UART 4 didn't wake system up. 1 = UART 4 wake system up.
[19]	**UR3WKST**	**UART 3 Wake System Up Status** 0 = UART 3 didn't wake system up. 1 = UART 3 wake system up.

[18]	**UR2WKST**	**UART 2 Wake System Up Status** 0 = UART 2 didn’t wake system up. 1 = UART 2 wake system up.
[17]	**UR1WKST**	**UART 1 Wake System Up Status** 0 = UART 1 didn’t wake system up. 1 = UART 1 wake system up.
[16]	**UR0WKST**	**UART 0 Wake System Up Status** 0 = UART 0 didn’t wake system up. 1 = UART 0 wake system up.
[15:14]	**Reserved**	Reserved.
[13]	**TMR5WKST**	**TIMER 5 Wake System Up Status** 0 = TIMER 5 didn’t wake system up. 1 = TIMER 5 wake system up.
[12]	**TMR4WKST**	**TIMER 4 Wake System Up Status** 0 = TIMER 4 didn’t wake system up. 1 = TIMER 4 wake system up.
[11]	**TMR3WKST**	**TIMER 3 Wake System Up Status** 0 = TIMER 3 didn’t wake system up. 1 = TIMER 3 wake system up.
[10]	**TMR2WKST**	**TIMER 2 Wake System Up Status** 0 = TIMER 2 didn’t wake system up. 1 = TIMER 2 wake system up.
[9]	**TMR1WKST**	**TIMER 1 Wake System Up Status** 0 = TIMER 1 didn’t wake system up. 1 = TIMER 1 wake system up.
[8]	**TMR0WKST**	**TIMER 0 Wake System Up Status** 0 = TIMER 0 didn’t wake system up. 1 = TIMER 0 wake system up.
[7]	**EINT3WKST**	**External Interrupt 3 Wake System Up Status** 0 = External Interrupt 3 didn’t wake system up. 1 = External Interrupt 3 wake system up.
[6]	**EINT2WKST**	**External Interrupt 2 Wake System Up Status** 0 = External Interrupt 2 didn’t wake system up. 1 = External Interrupt 2 wake system up.
[5]	**EINT1WKST**	**External Interrupt 1 Wake System Up Status** 0 = External Interrupt 1 didn’t wake system up. 1 = External Interrupt 1 wake system up.
[4]	**EINT0WKST**	**External Interrupt 0 Wake System Up Status** 0 = External Interrupt 0 didn’t wake system up. 1 = External Interrupt 0 wake system up.

[3]	**GPIOWKST**	**GPIO Wake System Up Status** 0 = GPIO didn't wake system up. 1 = GPIO wake system up.
[2:1]	**Reserved**	Reserved.
[0]	**WDTWKST**	**WDT Wake System Up Status** 0 = WDT didn't wake system up. 1 = WDT wake system up.

System Wakeup Source Status Register 1 (SYS_WKUPSSR1)

Register	Offset	R/W	Description	Reset Value
SYS_WKUPSSR1	SYS_BA+0x05C	R/W	System Wakeup Source Status Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							ADCWKST
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved			SDHWKST	USBDWKST	USBHWKST	EMAC1WKST	EMAC0WKST
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
LVDWKST	Reserved			CAN3WKST	CAN2WKST	CAN1WKST	CAN0WKST
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RTCWKST	Reserved			I2C3WKST	I2C2WKST	I2C1WKST	I2C0WKST

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**ADCWKST**	**ADC Wake System Up Status** 0 = ADC didn't wake system up. 1 = ADC wake system up.
[23:21]	**Reserved**	Reserved.
[20]	**SDHWKST**	**SDH Wake System Up Status** 0 = SDH didn't wake system up. 1 = SDH wake system up.
[19]	**USBDWKST**	**USB Device Wake System Up Status** 0 = USB device didn't wake system up. 1 = USB device wake system up.
[18]	**USBHWKST**	**USB Host Wake System Up Status** 0 = USB host didn't wake system up. 1 = USB host wake system up.
[17]	**EMAC1WKST**	**Ethernet MAC 1 Wake System Up Status** 0 = Ethernet MAC 1 didn't wake system up. 1 = Ethernet MAC 1 wake system up.
[16]	**EMAC0WKST**	**Ethernet MAC 0 Wake System Up Status** 0 = Ethernet MAC 0 didn't wake system up. 1 = Ethernet MAC 0 wake system up.
[15]	**LVDWKST**	**Low Voltage Detect Wake System Up Status** 0 = Low Voltage Detect didn't wake system up. 1 = Low Voltage Detect wake system up.

[14:12]	**Reserved**	Reserved.
[11]	**CAN3WKST**	**CAN 3 Wake System Up Status** 0 = CAN 3 didn't wake system up. 1 = CAN 3 wake system up.
[10]	**CAN2WKST**	**CAN 2 Wake System Up Status** 0 = CAN 2 didn't wake system up. 1 = CAN 2 wake system up.
[9]	**CAN1WKST**	**CAN 1 Wake System Up Status** 0 = CAN 1 didn't wake system up. 1 = CAN 1 wake system up.
[8]	**CAN0WKST**	**CAN 0 Wake System Up Status** 0 = CAN 0 didn't wake system up. 1 = CAN 0 wake system up.
[7]	**RTCWKST**	**RTC Wake System Up Status** 0 = RTC didn't wake system up. 1 = RTC wake system up.
[6:4]	**Reserved**	Reserved.
[3]	**I2C3WKST**	**I^{2}C 3 Wake System Up Status** 0 = I^{2}C 3 didn't wake system up. 1 = I^{2}C 3 wake system up.
[2]	**I2C2WKST**	**I^{2}C 2 Wake System Up Status** 0 = I^{2}C 2 didn't wake system up. 1 = I^{2}C 2 wake system up.
[1]	**I2C1WKST**	**I^{2}C 1 Wake System Up Status** 0 = I^{2}C 1 didn't wake system up. 1 = I^{2}C 1 wake system up.
[0]	**I2C0WKST**	**I^{2}C 0 Wake System Up Status** 0 = I^{2}C 0 didn't wake system up. 1 = I^{2}C 0 wake system up.

AHB IP Reset Control Register (SYS_AHBIPRST)

Register	Offset	R/W	Description	Reset Value
SYS_AHBIPRST	SYS_BA+0x060	R/W	AHB IP Reset Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							SDHRST
23	22	21	20	19	18	17	16
CRYPTORST	Reserved		FMIRST	USBDRST	USBHRST	EMAC1RST	EMAC0RST
15	14	13	12	11	10	9	8
Reserved				VCAP1RST	VCAP0RST	Reserved	I2SRST
7	6	5	4	3	2	1	0
GPIORST	SDICRST	PDMA1RST	PDMA0RST	EBIRST	CPURST	Reserved	CHIPRST

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**SDHRST**	**SDIO Controller Reset Enable Bit** 0 = SDIO controller reset Disabled. 1 = SDIO controller reset Enabled.
[23]	**CRYPTORST**	**Cryptographic Accelerator Reset Enable Bit** 0 = Cryptographic Accelerator reset Disabled. 1 = Cryptographic Accelerator reset Enabled.
[22:21]	**Reserved**	Reserved.
[20]	**FMIRST**	**FMI Controller Reset Enable Bit** 0 = FMI controller reset Disabled. 1 = FMI controller reset Enabled.
[19]	**USBDRST**	**USB Device Controller Reset Enable Bit** 0 = USB device controller reset Disabled. 1 = USB device controller reset Enabled.
[18]	**USBHRST**	**USB Host Controller (EHCI/OHCI) Reset Enable Bit** 0 = USB host controller (EHCI/OHCI) reset Disabled. 1 = USB host controller (EHCI/OHCI) reset Enabled.
[17]	**EMAC1RST**	**Ethernet MAC 1 Reset Enable Bit** 0 = Ethernet MAC 1 reset Disabled. 1 = Ethernet MAC 1 reset Enabled.
[16]	**EMAC0RST**	**Ethernet MAC 0 Reset Enable Bit** 0 = Ethernet MAC 0 reset Disabled. 1 = Ethernet MAC 0 reset Enabled.

[15:12]	**Reserved**	Reserved.
[11]	**VCAP1RST**	**Capture Sensor Interface 1 Reset Enable Bit** 0 = Capture sensor interface 1 reset Disabled. 1 = Capture sensor interface 1 reset Enabled.
[10]	**VCAP0RST**	**Capture Sensor Interface 0 Reset Enable Bit** 0 = Capture sensor interface 0 reset Disabled. 1 = Capture sensor interface 0 reset Enabled.
[9]	**Reserved**	Reserved.
[8]	**I2S**	**I^{2}S Controller Reset Enable Bit** 0 = I^{2}S controller reset Disabled. 1 = I^{2}S controller reset Enabled.
[7]	**GPIORST**	**GPIO Reset Enable Bit** 0 = GPIO reset Disabled. 1 = GPIO reset Enabled.
[6]	**SDICRST**	**SDRAM Controller Reset Enable Bit** 0 = SDRAM controller reset Disabled. 1 = SDRAM Controller reset Enabled.
[5]	**PDMA1RST**	**PDMA1 Reset Enable Bit** 0 = PDMA1 reset Disabled. 1 = PDMA1 reset Enabled.
[4]	**PDMA0RST**	**PDMA0 Reset Enable Bit** 0 = PDMA0 reset Disabled. 1 = PDMA0 reset Enabled.
[2]	**CPURST**	**CPU Pulse Reset Enable Bit** This bit is used to generate a reset pulse to ARM926EJ-S™ CPU. When set this bit high, reset controller generates a 6 system clock long reset pulse to ARM926EJ-S™ CPU. After the reset completed, this bit will be clear to low automatically. 0 = CPU pulse reset Disabled. 1 = CPU pulse reset Enabled.
[1]	**Reserved**	Reserved.
[0]	**CHIP**	**Chip Reset Enable Bit** 0 = Chip reset Disabled. 1 = Chip reset Enabled.

APB IP Reset Control Register 0 (SYS_APBIPRST0)

Register	Offset	R/W	Description	Reset Value
SYS_APBIPRST0	SYS_BA+0x064	R/W	APB IP Reset Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					Reserved	UART9RST	UART8RST
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		TIMER5RST	TIMER4RST	TIMER3RST	TIMER2RST	TIMER1RST	TIMER0RST
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			AICRST	Reserved			

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25]	**UART9RST**	**UART 9 Reset Enable Bit** 0 = UART 9 reset Disabled. 1 = UART 9 reset Enabled.
[24]	**UART8RST**	**UART 8 Reset Enable Bit** 0 = UART 8 reset Disabled. 1 = UART 8 reset Enabled.
[23]	**UART7RST**	**UART 7 Reset Enable Bit** 0 = UART 7 reset Disabled. 1 = UART 7 reset Enabled.
[22]	**UART6RST**	**UART 6 Reset Enable Bit** 0 = UART 6 reset Disabled. 1 = UART 6 reset Enabled.
[21]	**UART5RST**	**UART 5 Reset Enable Bit** 0 = UART 5 reset Disabled. 1 = UART 5 reset Enabled.
[20]	**UART4RST**	**UART 4 Reset Enable Bit** 0 = UART 4 reset Disabled. 1 = UART 4 reset Enabled.
[19]	**UART3RST**	**UART 3 Reset Enable Bit** 0 = UART 3 reset Disabled. 1 = UART 3 reset Enabled.

[18]	**UART2RST**	**UART 2 Reset Enable Bit** 0 = UART 2 reset Disabled. 1 = UART 2 reset Enabled.
[17]	**UART1RST**	**UART 1 Reset Enable Bit** 0 = UART 1 reset Disabled. 1 = UART 1 reset Enabled.
[16]	**UART0RST**	**UART 0 Reset Enable Bit** 0 = UART 0 reset Disabled. 1 = UART 0 reset Enabled.
[15:14]	**Reserved**	Reserved.
[13]	**TIMER5RST**	**TIMER 5 Reset Enable Bit** 0 = TIMER 5 reset Disabled. 1 = TIMER 5 reset Enabled.
[12]	**TIMER4RST**	**TIMER 4 Reset Enable Bit** 0 = TIMER 4 reset Disabled. 1 = TIMER 4 reset Enabled.
[11]	**TIMER3RST**	**TIMER 3 Reset Enable Bit** 0 = TIMER 3 reset Disabled. 1 = TIMER 3 reset Enabled.
[10]	**TIMER2RST**	**TIMER 2 Reset Enable Bit** 0 = TIMER 2 reset Disabled. 1 = TIMER 2 reset Enabled.
[9]	**TIMER1RST**	**TIMER 1 Reset Enable Bit** 0 = TIMER 1 reset Disabled. 1 = TIMER 1 reset Enabled.
[8]	**TIMER0RST**	**TIMER 0 Reset Enable Bit** 0 = TIMER 0 reset Disabled. 1 = TIMER 0 reset Enabled.
[7:5]	**Reserved**	Reserved.
[4]	**AICRST**	**AIC Reset Enable Bit** 0 = AIC reset Disabled. 1 = AIC reset Enabled.
[3:0]	**Reserved**	Reserved.

APB IP Reset Control Register 1 (SYS_APBIPRST1)

Register	Offset	R/W	Description	Reset Value
SYS_APBIPRST1	SYS_BA+0x068	R/W	APB IP Reset Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				PWM1RST	PWM0RST	Reserved	ADCRST
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		SMC1RST	SMC0RST	CAN3RST	CAN2RST	CAN1RST	CAN0RST
7	6	5	4	3	2	1	0
Reserved	SPI1RST	SPI0RST	QSPI0RST	I2C3RST	I2C2RST	I2C1RST	I2C0RST

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27]	**PWM1RST**	**PWM1 Reset Enable Bit** 0 = PWM1 reset Disabled. 1 = PWM1 reset Enabled.
[26]	**PWM0RST**	**PWM0 Reset Enable Bit** 0 = PWM0 reset Disabled. 1 = PWM0 reset Enabled.
[25]	**Reserved**	Reserved.
[24]	**ADCRST**	**ADC Reset Enable Bit** 0 = ADC reset Disabled. 1 = ADC reset Enabled.
[23:14]	**Reserved**	Reserved.
[13]	**SMC1RST**	**SMC 1 Reset Enable Bit** 0 = SMC 1 reset Disabled. 1 = SMC 1 reset Enabled.
[12]	**SMC0RST**	**SMC 0 Reset Enable Bit** 0 = SMC 0 reset Disabled. 1 = SMC 0 reset Enabled.
[11]	**CAN3RST**	**CAN 3 Reset Enable Bit** 0 = CAN 3 reset Disabled. 1 = CAN 3 reset Enabled.

[10]	**CAN2RST**	**CAN 2 Reset Enable Bit** 0 = CAN 2 reset Disabled. 1 = CAN 2 reset Enabled.
[9]	**CAN1RST**	**CAN 1 Reset Enable Bit** 0 = CAN 1 reset Disabled. 1 = CAN 1 reset Enabled.
[8]	**CAN0RST**	**CAN 0 Reset Enable Bit** 0 = CAN 0 reset Disabled. 1 = CAN 0 reset Enabled.
[7]	**Reserved**	Reserved.
[6]	**SPI1RST**	**SPI 1 Reset Enable Bit** 0 = SPI 1 reset Disabled. 1 = SPI 1 reset Enabled.
[5]	**SPI0RST**	**SPI 1 Reset Enable Bit** 0 = SPI 0 reset Disabled. 1 = SPI 0 reset Enabled.
[4]	**QSPI0RST**	**QSPI 0 Reset Enable Bit** 0 = QSPI 0 reset Disabled. 1 = QSPI 0 reset Enabled.
[3]	**I2C3RST**	**I^{2}C 3 Reset Enable Bit** 0 = I^{2}C 3 reset Disabled. 1 = I^{2}C 3 reset Enabled.
[2]	**I2C2RST**	**I^{2}C 1 Reset Enable Bit** 0 = I^{2}C 2 reset Disabled. 1 = I^{2}C 2 reset Enabled.
[1]	**I2C1RST**	**I^{2}C 1 Reset Enable Bit** 0 = I^{2}C 1 reset Disabled. 1 = I^{2}C 1 reset Enabled.
[0]	**I2C0RST**	**I^{2}C 0 Reset Enable Bit** 0 = I^{2}C 0 reset Disabled. 1 = I^{2}C 0 reset Enabled.

Reset Source Active Status Register (SYS_RSTSTS)

Register	Offset	R/W	Description	Reset Value
SYS_RSTSTS	SYS_BA+0x06C	R/W	Reset Source Active Status 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**
Reserved		**WDTRSTS**	**CPURSTS**	**CHIPRSTS**	**LVRRSTS**	**PINRSTS**	**PORRSTS**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[5]	**WDTRSTS**	**Chip Reset by Watchdog Timer Status** 0 = No reset from watchdog timer. 1 = Watchdog timer had issued reset signal to reset the chip.
[4]	**CPURSTS**	**CPU Reset by CPU_LVL (AHBIPRST[1]) or CPU_PLS (AHBIPRST[2]) Status** 0 = No CPU reset from CPU_LVL (AHBIPRST[1]) or CPU_PLS (AHBIPRST[2]). 1 = CPU_LVL (AHBIPRST[1]) or CPU_PLS (AHBIPRST[2]) has been high to reset the CPU.
[3]	**CHIPRSTS**	**Chip Reset by CHIP (AHBIPRST[0]) Status** 0 = No reset from CHIP (AHBIPRST[0]). 1 = CHIP (AHBIPRST[0]) has been high to reset CPU.
[2]	**LVRRSTS**	**Chip Reset by LVRD Status** 0 = No reset from LVRD. 1 = LVRD had issued reset signal to reset the chip.
[1]	**PINRSTS**	**Chip Reset by NRESET Pin Status** 0 = No reset from nRESET pin. 1 = nRESET pin had issued reset signal to reset the chip.
[0]	**PORRSTS**	**Chip Reset by POR Status** 0 = No reset from POR. 1 = POR had issued reset signal to reset the chip.

GPIOA Low Byte Multiple Function Control Register (SYS_GPA_MFPL)

Register	Offset	R/W	Description	Reset Value
SYS_GPA_MFPL	SYS_BA+0x070	R/W	GPIOA Low Byte Multiple Function Control Register	0x0XXX_XX00

31	30	29	28	27	26	25	24
MFP_GPA7				MFP_GPA6			
23	22	21	20	19	18	17	16
MFP_GPA5				MFP_GPA4			
15	14	13	12	11	10	9	8
MFP_GPA3				MFP_GPA2			
7	6	5	4	3	2	1	0
MFP_GPA1				MFP_GPA0			

Bits	Description	
[31:28]	**MFP_GPA7**	**Pin PA.7 Multi-function Pin Selection**
[27:24]	**MFP_GPA6**	**Pin PA.6 Multi-function Pin Selection**
[23:20]	**MFP_GPA5**	**Pin PA.5 Multi-function Pin Selection**
[19:16]	**MFP_GPA4**	**Pin PA.4 Multi-function Pin Selection**
[15:12]	**MFP_GPA3**	**Pin PA.3 Multi-function Pin Selection**
[11:8]	**MFP_GPA2**	**Pin PA.2 Multi-function Pin Selection**
[7:4]	**MFP_GPA1**	**Pin PA.1 Multi-function Pin Selection**
[3:0]	**MFP_GPA0**	**Pin PA.0 Multi-function Pin Selection**

Note: Please refer to 4.2.2 and 4.2.3 for the multi-function pin summary.

GPIOA High Byte Multiple Function Control Register (SYS_GPA_MFPH)

Register	Offset	R/W	Description	Reset Value
SYS_GPA_MFPH	SYS_BA+0x074	R/W	GPIOA High Byte Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
MFP_GPA15				MFP_GPA14			
23	22	21	20	19	18	17	16
MFP_GPA13				MFP_GPA12			
15	14	13	12	11	10	9	8
MFP_GPA11				MFP_GPA10			
7	6	5	4	3	2	1	0
MFP_GPA9				MFP_GPA8			

Bits	Description	
[31:28]	**MFP_GPA15**	**Pin PA.15 Multi-function Pin Selection**
[27:24]	**MFP_GPA14**	**Pin PA.14 Multi-function Pin Selection**
[23:20]	**MFP_GPA13**	**Pin PA.13 Multi-function Pin Selection**
[19:16]	**MFP_GPA12**	**Pin PA.12 Multi-function Pin Selection**
[15:12]	**MFP_GPA11**	**Pin PA.11 Multi-function Pin Selection**
[11:8]	**MFP_GPA10**	**Pin PA.10 Multi-function Pin Selection**
[7:4]	**MFP_GPA9**	**Pin PA.9 Multi-function Pin Selection**
[3:0]	**MFP_GPA8**	**Pin PA.8 Multi-function Pin Selection**

Note: Please refer to 4.2.2 and 4.2.3 for the multi-function pin summary.

GPIOB Low Byte Multiple Function Control Register (SYS_GPB_MFPL)

Register	Offset	R/W	Description	Reset Value
SYS_GPB_MFPL	SYS_BA+0x078	R/W	GPIOB Low Byte Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
MFP_GPB7				MFP_GPB6			
23	22	21	20	19	18	17	16
MFP_GPB5				MFP_GPB4			
15	14	13	12	11	10	9	8
MFP_GPB3				MFP_GPB2			
7	6	5	4	3	2	1	0
MFP_GPB1				MFP_GPB0			

Bits	Description	
[31:28]	**MFP_GPB7**	**Pin PB.7 Multi-function Pin Selection**
[27:24]	**MFP_GPB6**	**Pin PB.6 Multi-function Pin Selection**
[23:20]	**MFP_GPB5**	**Pin PB.5 Multi-function Pin Selection**
[19:16]	**MFP_GPB4**	**Pin PB.4 Multi-function Pin Selection**
[15:12]	**MFP_GPB3**	**Pin PB.3 Multi-function Pin Selection**
[11:8]	**MFP_GPB2**	**Pin PB.2 Multi-function Pin Selection**
[7:4]	**MFP_GPB1**	**Pin PB.1 Multi-function Pin Selection**
[3:0]	**MFP_GPB0**	**Pin PB.0 Multi-function Pin Selection**

Note: Please refer to 4.2.2 and 4.2.3 for the multi-function pin summary.

GPIOB High Byte Multiple Function Control Register (SYS_GPB_MFPH)

Register	Offset	R/W	Description	Reset Value
SYS_GPB_MFPH	SYS_BA+0x07C	R/W	GPIOB High Byte Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
MFP_GPB13				**MFP_GPB12**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
MFP_GPB11				**MFP_GPB10**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MFP_GPB9				**MFP_GPB8**			

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:20]	**MFP_GPB13**	**Pin PB.13 Multi-function Pin Selection**
[19:16]	**MFP_GPB12**	**Pin PB.12 Multi-function Pin Selection**
[15:12]	**MFP_GPB11**	**Pin PB.11 Multi-function Pin Selection**
[11:8]	**MFP_GPB10**	**Pin PB.10 Multi-function Pin Selection**
[7:4]	**MFP_GPB9**	**Pin PB.9 Multi-function Pin Selection**
[3:0]	**MFP_GPB8**	**Pin PB.8 Multi-function Pin Selection**

Note: Please refer to 4.2.2 and 4.2.3 for the multi-function pin summary.

GPIOC Low Byte Multiple Function Control Register (SYS_GPC_MFPL)

Register	Offset	R/W	Description	Reset Value
SYS_GPC_MFPL	SYS_BA+0x080	R/W	GPIOC Low Byte Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
MFP_GPC7				MFP_GPC6			
23	22	21	20	19	18	17	16
MFP_GPC5				MFP_GPC4			
15	14	13	12	11	10	9	8
MFP_GPC3				MFP_GPC2			
7	6	5	4	3	2	1	0
MFP_GPC1				MFP_GPC0			

Bits	Description	
[31:28]	MFP_GPC7	Pin PC.7 Multi-function Pin Selection
[27:24]	MFP_GPC6	Pin PC.6 Multi-function Pin Selection
[23:20]	MFP_GPC5	Pin PC.5 Multi-function Pin Selection
[19:16]	MFP_GPC4	Pin PC.4 Multi-function Pin Selection
[15:12]	MFP_GPC3	Pin PC.3 Multi-function Pin Selection
[11:8]	MFP_GPC2	Pin PC.2 Multi-function Pin Selection
[7:4]	MFP_GPC1	Pin PC.1 Multi-function Pin Selection
[3:0]	MFP_GPC0	Pin PC.0 Multi-function Pin Selection

Note: Please refer to 4.2.2 and 4.2.3 for the multi-function pin summary.

GPIOC High Byte Multiple Function Control Register (SYS_GPC_MFPH)

Register	Offset	R/W	Description	Reset Value
SYS_GPC_MFPH	SYS_BA+0x084	R/W	GPIOC High Byte Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
MFP_GPC15				MFP_GPC14			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
MFP_GPC13				MFP_GPC12			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
MFP_GPC11				MFP_GPC10			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MFP_GPC9				MFP_GPC8			

Bits	Description	
[31:28]	**MFP_GPC15**	**Pin PC.15 Multi-function Pin Selection**
[27:24]	**MFP_GPC14**	**Pin PC.14 Multi-function Pin Selection**
[23:20]	**MFP_GPC13**	**Pin PC.13 Multi-function Pin Selection**
[19:16]	**MFP_GPC12**	**Pin PC.12 Multi-function Pin Selection**
[15:12]	**MFP_GPC11**	**Pin PC.11 Multi-function Pin Selection**
[11:8]	**MFP_GPC10**	**Pin PC.10 Multi-function Pin Selection**
[7:4]	**MFP_GPC9**	**Pin PC.9 Multi-function Pin Selection**
[3:0]	**MFP_GPC8**	**Pin PC.8 Multi-function Pin Selection**

Note: Please refer to 4.2.2 and 4.2.3 for the multi-function pin summary.

GPIOD Low Byte Multiple Function Control Register (SYS_GPD_MFPL)

Register	Offset	R/W	Description	Reset Value
SYS_GPD_MFPL	SYS_BA+0x088	R/W	GPIOD Low Byte Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
MFP_GPD7				MFP_GPD6			
23	22	21	20	19	18	17	16
MFP_GPD5				MFP_GPD4			
15	14	13	12	11	10	9	8
MFP_GPD3				MFP_GPD2			
7	6	5	4	3	2	1	0
MFP_GPD1				MFP_GPD0			

Bits	Description	
[31:28]	**MFP_GPD7**	**Pin PD.7 Multi-function Pin Selection**
[27:24]	**MFP_GPD6**	**Pin PD.6 Multi-function Pin Selection**
[23:20]	**MFP_GPD5**	**Pin PD.5 Multi-function Pin Selection**
[19:16]	**MFP_GPD4**	**Pin PD.4 Multi-function Pin Selection**
[15:12]	**MFP_GPD3**	**Pin PD.3 Multi-function Pin Selection**
[11:8]	**MFP_GPD2**	**Pin PD.2 Multi-function Pin Selection**
[7:4]	**MFP_GPD1**	**Pin PD.1 Multi-function Pin Selection**
[3:0]	**MFP_GPD0**	**Pin PD.0 Multi-function Pin Selection**

Note: Please refer to 4.2.2 and 4.2.3 for the multi-function pin summary.

GPIOD High Byte Multiple Function Control Register (SYS_GPD_MFPH)

Register	Offset	R/W	Description	Reset Value
SYS_GPD_MFPH	SYS_BA+0x08C	R/W	GPIOD High Byte Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
MFP_GPD15				MFP_GPD14			
23	22	21	20	19	18	17	16
MFP_GPD13				MFP_GPD12			
15	14	13	12	11	10	9	8
MFP_GPD11				MFP_GPD10			
7	6	5	4	3	2	1	0
MFP_GPD9				MFP_GPD8			

Bits	Description	
[31:28]	**MFP_GPD15**	**Pin PD.15 Multi-function Pin Selection**
[27:24]	**MFP_GPD14**	**Pin PD.14 Multi-function Pin Selection**
[23:20]	**MFP_GPD13**	**Pin PD.13 Multi-function Pin Selection**
[19:16]	**MFP_GPD12**	**Pin PD.12 Multi-function Pin Selection**
[15:12]	**MFP_GPD11**	**Pin PD.11 Multi-function Pin Selection**
[11:8]	**MFP_GPD10**	**Pin PD.10 Multi-function Pin Selection**
[7:4]	**MFP_GPD9**	**Pin PD.9 Multi-function Pin Selection**
[3:0]	**MFP_GPD8**	**Pin PD.8 Multi-function Pin Selection**

Note: Please refer to 4.2.2 and 4.2.3 for the multi-function pin summary.

GPIOE Low Byte Multiple Function Control Register (SYS_GPE_MFPL)

Register	Offset	R/W	Description	Reset Value
SYS_GPE_MFPL	SYS_BA+0x090	R/W	GPIOE Low Byte Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
MFP_GPE7				MFP_GPE6			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
MFP_GPE5				MFP_GPE4			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
MFP_GPE3				MFP_GPE2			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MFP_GPE1				MFP_GPE0			

Bits	Description	
[31:28]	**MFP_GPE7**	**Pin PE.7 Multi-function Pin Selection**
[27:24]	**MFP_GPE6**	**Pin PE.6 Multi-function Pin Selection**
[23:20]	**MFP_GPE5**	**Pin PE.5 Multi-function Pin Selection**
[19:16]	**MFP_GPE4**	**Pin PE.4 Multi-function Pin Selection**
[15:12]	**MFP_GPE3**	**Pin PE.3 Multi-function Pin Selection**
[11:8]	**MFP_GPE2**	**Pin PE.2 Multi-function Pin Selection**
[7:4]	**MFP_GPE1**	**Pin PE.1 Multi-function Pin Selection**
[3:0]	**MFP_GPE0**	**Pin PE.0 Multi-function Pin Selection**

Note: Please refer to 4.2.2 and 4.2.3 for the multi-function pin summary.

GPIOE High Byte Multiple Function Control Register (SYS_GPE_MFPH)

Register	Offset	R/W	Description	Reset Value
SYS_GPE_MFPH	SYS_BA+0x094	R/W	GPIOE High Byte Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved				MFP_GPE12			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
MFP_GPE11				MFP_GPE10			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MFP_GPE9				MFP_GPE8			

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19:16]	**MFP_GPE12**	**Pin PE.12 Multi-function Pin Selection**
[15:12]	**MFP_GPE11**	**Pin PE.11 Multi-function Pin Selection**
[11:8]	**MFP_GPE10**	**Pin PE.10 Multi-function Pin Selection**
[7:4]	**MFP_GPE9**	**Pin PE.9 Multi-function Pin Selection**
[3:0]	**MFP_GPE8**	**Pin PE.8 Multi-function Pin Selection**

Note: Please refer to 4.2.2 and 4.2.3 for the multi-function pin summary.

GPIOF Low Byte Multiple Function Control Register (SYS_GPF_MFPL)

Register	Offset	R/W	Description	Reset Value
SYS_GPF_MFPL	SYS_BA+0x098	R/W	GPIOF Low Byte Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
MFP_GPF7				MFP_GPF6			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
MFP_GPF5				MFP_GPF4			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
MFP_GPF3				MFP_GPF2			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MFP_GPF1				MFP_GPF0			

Bits	Description	
[31:28]	**MFP_GPF7**	**Pin PF.7 Multi-function Pin Selection**
[27:24]	**MFP_GPF6**	**Pin PF.6 Multi-function Pin Selection**
[23:20]	**MFP_GPF5**	**Pin PF.5 Multi-function Pin Selection**
[19:16]	**MFP_GPF4**	**Pin PF.4 Multi-function Pin Selection**
[15:12]	**MFP_GPF3**	**Pin PF.3 Multi-function Pin Selection**
[11:8]	**MFP_GPF2**	**Pin PF.2 Multi-function Pin Selection**
[7:4]	**MFP_GPF1**	**Pin PF.1 Multi-function Pin Selection**
[3:0]	**MFP_GPF0**	**Pin PF.0 Multi-function Pin Selection**

Note: Please refer to 4.2.2 and 4.2.3 for the multi-function pin summary.

GPIOF High Byte Multiple Function Control Register (SYS_GPF_MFPH)

Register	Offset	R/W	Description	Reset Value
SYS_GPF_MFPH	SYS_BA+0x09C	R/W	GPIOF High Byte Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved				**MFP_GPF12**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
MFP_GPF11				**MFP_GPF10**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MFP_GPF9				**MFP_GPF8**			

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19:16]	**MFP_GPF12**	**Pin PF.12 Multi-function Pin Selection**
[15:12]	**MFP_GPF11**	**Pin PF.11 Multi-function Pin Selection**
[11:8]	**MFP_GPF10**	**Pin PF.10 Multi-function Pin Selection**
[7:4]	**MFP_GPF9**	**Pin PF.9 Multi-function Pin Selection**
[3:0]	**MFP_GPF8**	**Pin PF.8 Multi-function Pin Selection**

Note: Please refer to 4.2.2 and 4.2.3 for the multi-function pin summary.

GPIOG Low Byte Multiple Function Control Register (SYS_GPG_MFPL)

Register	Offset	R/W	Description	Reset Value
SYS_GPG_MFPL	SYS_BA+0x0A0	R/W	GPIOG Low Byte Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
MFP_GPG7				MFP_GPG6			
23	22	21	20	19	18	17	16
MFP_GPG5				MFP_GPG4			
15	14	13	12	11	10	9	8
MFP_GPG3				MFP_GPG2			
7	6	5	4	3	2	1	0
MFP_GPG1				MFP_GPG0			

Bits	Description	
[31:28]	MFP_GPG7	Pin PG.7 Multi-function Pin Selection
[27:24]	MFP_GPG6	Pin PG.6 Multi-function Pin Selection
[23:20]	MFP_GPG5	Pin PG.5 Multi-function Pin Selection
[19:16]	MFP_GPG4	Pin PG.4 Multi-function Pin Selection
[15:12]	MFP_GPG3	Pin PG.3 Multi-function Pin Selection
[11:8]	MFP_GPG2	Pin PG.2 Multi-function Pin Selection
[7:4]	MFP_GPG1	Pin PG.1 Multi-function Pin Selection
[3:0]	MFP_GPG0	Pin PG.0 Multi-function Pin Selection

Note: Please refer to 4.2.2 and 4.2.3 for the multi-function pin summary.

GPIOG High Byte Multiple Function Control Register (SYS_GPG_MFPH)

Register	Offset	R/W	Description	Reset Value
SYS_GPG_MFPH	SYS_BA+0x0A4	R/W	GPIOG High Byte Multiple Function Control Register	0xXXXX_X000

31	30	29	28	27	26	25	24
MFP_GPG15				MFP_GPG14			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
MFP_GPG13				MFP_GPG12			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
MFP_GPG11				MFP_GPG10			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MFP_GPG9				MFP_GPG8			

Bits	Description	
[31:28]	**MFP_GPG15**	**Pin PG.15 Multi-function Pin Selection**
[27:24]	**MFP_GPG14**	**Pin PG.14 Multi-function Pin Selection**
[23:20]	**MFP_GPG13**	**Pin PG.13 Multi-function Pin Selection**
[19:16]	**MFP_GPG12**	**Pin PG.12 Multi-function Pin Selection**
[15:12]	**MFP_GPG11**	**Pin PG.11 Multi-function Pin Selection**
[11:8]	**MFP_GPG10**	**Pin PG.10 Multi-function Pin Selection**
[7:4]	**MFP_GPG9**	**Pin PG.9 Multi-function Pin Selection**
[3:0]	**MFP_GPG8**	**Pin PG.8 Multi-function Pin Selection**

Note: Please refer to 4.2.2 and 4.2.3 for the multi-function pin summary.

DDR I/O Driving Strength Control Register (SYS_DDR_DSCTL)

Register	Offset	R/W	Description	Reset Value
SYS_DDR_DSCTL	SYS_BA+0x0F0	R/W	DDR I/O Driving Strength 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
DATA_DS		ADDR_DS		CTRL_DS		CLK_DS	

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:6]	**DATA_DS**	**DDR Data I/O Driving Strength Selection** This bit controls the driving strength for DDR I/O used as data. 00 = Reserved. 01 = Reduced Strength. 10 = Reserved. 11 = Full Strength.
[5:4]	**ADDR_DS**	**DDR Address I/O Driving Strength Selection** This bit controls the driving strength for DDR I/O used as address. 00 = Reserved. 01 = Reduced Strength. 10 = Reserved. 11 = Full Strength.
[3:2]	**CTRL_DS**	**DDR Control I/O Driving Strength Selection** This bit controls the driving strength for DDR I/O used as control signals. 00 = Reserved. 01 = Reduced Strength. 10 = Reserved. 11 = Full Strength.
[1:0]	**CLK_DS**	**DDR Clock I/O Driving Strength Selection** This bit controls the driving strength for DDR I/O used as clock. 00 = Reserved. 01 = Reduced Strength. 10 = Reserved. 11 = Full Strength.

GPIOB Low Byte Driving Strength Control Register (SYS_GPBL_DSCTL)

Register	Offset	R/W	Description	Reset Value
SYS_GPBL_DSCTL	SYS_BA+0x0F4	R/W	GPIOB Low Byte Driving Strength Control Register	0x4444_4444

31	30	29	28	27	26	25	24
DS_GPB7				DS_GPB6			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DS_GPB5				DS_GPB4			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DS_GPB3				DS_GPB2			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DS_GPB1				DS_GPB0			

Bits	Description	
[31:28]	**DS_GPB7**	**Pin PB.7 Driving Strength Selection** This field controls the pin PB.7 driving strength 000 = Pin PB.7 driving strength is 2.2mA. 001 = Pin PB.7 driving strength is 6.5mA. 010 = Pin PB.7 driving strength is 8.7mA. 011 = Pin PB.7 driving strength is 13.0mA. 100 = Pin PB.7 driving strength is 15.2mA. 101 = Pin PB.7 driving strength is 19.5mA. 110 = Pin PB.7 driving strength is 21.7mA. 111 = Pin PB.7 driving strength is 26.1mA. Others = Reserved.
[27:24]	**DS_GPB6**	**Pin PB.6 Driving Strength Selection** This field controls the pin PB.6 driving strength 000 = Pin PB.6 driving strength is 2.2mA. 001 = Pin PB.6 driving strength is 6.5mA. 010 = Pin PB.6 driving strength is 8.7mA. 011 = Pin PB.6 driving strength is 13.0mA. 100 = Pin PB.6 driving strength is 15.2mA. 101 = Pin PB.6 driving strength is 19.5mA. 110 = Pin PB.6 driving strength is 21.7mA. 111 = Pin PB.6 driving strength is 26.1mA. Others = Reserved.

[23:20]	**DS_GPB5**	**Pin PB.5 Driving Strength Selection** This field controls the pin PB.5 driving strength 000 = Pin PB.5 driving strength is 2.2mA. 001 = Pin PB.5 driving strength is 6.5mA. 010 = Pin PB.5 driving strength is 8.7mA. 011 = Pin PB.5 driving strength is 13.0mA. 100 = Pin PB.5 driving strength is 15.2mA. 101 = Pin PB.5 driving strength is 19.5mA. 110 = Pin PB.5 driving strength is 21.7mA. 111 = Pin PB.5 driving strength is 26.1mA. Others = Reserved.
[19:16]	**DS_GPB4**	**Pin PB.4 Driving Strength Selection** This field controls the pin PB.4 driving strength 000 = Pin PB.4 driving strength is 2.2mA. 001 = Pin PB.4 driving strength is 6.5mA. 010 = Pin PB.4 driving strength is 8.7mA. 011 = Pin PB.4 driving strength is 13.0mA. 100 = Pin PB.4 driving strength is 15.2mA. 101 = Pin PB.4 driving strength is 19.5mA. 110 = Pin PB.4 driving strength is 21.7mA. 111 = Pin PB.4 driving strength is 26.1mA. Others = Reserved.
[15:12]	**DS_GPB3**	**Pin PB.3 Driving Strength Selection** This field controls the pin PB.3 driving strength 000 = Pin PB.3 driving strength is 2.2mA. 001 = Pin PB.3 driving strength is 6.5mA. 010 = Pin PB.3 driving strength is 8.7mA. 011 = Pin PB.3 driving strength is 13.0mA. 100 = Pin PB.3 driving strength is 15.2mA. 101 = Pin PB.3 driving strength is 19.5mA. 110 = Pin PB.3 driving strength is 21.7mA. 111 = Pin PB.3 driving strength is 26.1mA. Others = Reserved.

[11:8]	**DS_GPB2**	**Pin PB.2 Driving Strength Selection** This field controls the pin PB.2 driving strength 000 = Pin PB.2 driving strength is 2.2mA. 001 = Pin PB.2 driving strength is 6.5mA. 010 = Pin PB.2 driving strength is 8.7mA. 011 = Pin PB.2 driving strength is 13.0mA. 100 = Pin PB.2 driving strength is 15.2mA. 101 = Pin PB.2 driving strength is 19.5mA. 110 = Pin PB.2 driving strength is 21.7mA. 111 = Pin PB.2 driving strength is 26.1mA. Others = Reserved.
[7:4]	**DS_GPB1**	**Pin PB.1 Driving Strength Selection** This field controls the pin PB.1 driving strength 000 = Pin PB.1 driving strength is 2.2mA. 001 = Pin PB.1 driving strength is 6.5mA. 010 = Pin PB.1 driving strength is 8.7mA. 011 = Pin PB.1 driving strength is 13.0mA. 100 = Pin PB.1 driving strength is 15.2mA. 101 = Pin PB.1 driving strength is 19.5mA. 110 = Pin PB.1 driving strength is 21.7mA. 111 = Pin PB.1 driving strength is 26.1mA. Others = Reserved.
[3:0]	**DS_GPB0**	**Pin PB.0 Driving Strength Selection** This field controls the pin PB.0 driving strength 000 = Pin PB.0 driving strength is 2.2mA. 001 = Pin PB.0 driving strength is 6.5mA. 010 = Pin PB.0 driving strength is 8.7mA. 011 = Pin PB.0 driving strength is 13.0mA. 100 = Pin PB.0 driving strength is 15.2mA. 101 = Pin PB.0 driving strength is 19.5mA. 110 = Pin PB.0 driving strength is 21.7mA. 111 = Pin PB.0 driving strength is 26.1mA. Others = Reserved.

Power-On-reset Disable Control Register (SYS_PORDISCR)

Register	Offset	R/W	Description	Reset Value
SYS_PORDISCR	SYS_BA+0x100	R/W	Power-On-reset Disable Control 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
POR_DIS_CODE							
7	6	5	4	3	2	1	0
POR_DIS_CODE							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**POR_DIS_CODE**	**Power-on-reset Disable Code (Write-protection Bits)** When powered on, the Power-On-Reset (POR) circuit generates a reset signal to reset whole chip function. However, after power is ready, the POR circuit would consume a few power. To minimize the POR circuit power consumption, user to disable POR circuit by writing 0x5AA5 to this field. The POR circuit will become active again when this field is set to other value or chip is reset by other reset source, including /RESET pin, Watchdog, LVR reset and the software chip reset function. This field is protected. It means that before programming it, user has to write "59h", "16h" and "88h" to address 0xB000_01FC continuously to disable the register protection. Refer to the register REGWRPROT at address SYS_BA+0x1FC for detail.

Reset Pin De-bounce Control Register (SYS_RSTDEBCTL)

Register	Offset	R/W	Description	Reset Value
SYS_RSTDEBCTL	SYS_BA+0x10C	R/W	Reset Pin De-bounce Control Register	0x0000_04B0

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

Bits	Description	
[31]	**RSTDEBEN**	**Reset Pin De-bounce Enable Bit** 0 = Reset pin de-bounce Disabled. (Default) 1 = Reset pin de-bounce Enabled.
[31:16]	**Reserved**	Reserved.
[15:0]	**DEBCNT**	**Power-on-reset Disable Code (Write-protection Bits)** This 16-bit external RESET De-bounce Counter can specify the external RESET de-bounce time up to around 5.46ms (0xFFFF) @ XIN=12 MHz. The default external RESET de-bounce time is 0.1ms (0x04B0) @ XIN = 12 MHz.

Register Write-protection Control Register (SYS_REGWPCTL)

Some of the system control registers need to be protected to avoid inadvertent write and disturb the chip operation. These system control registers are protected after the power-on reset till user to disable register protection. For user to program these protected registers, a register protection disable sequence needs to be followed by a special programming. The register protection disable sequence is writing the data “59h”, “16h” “88h” to the register REGWRPROT address at 0xB000_01FC 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 0xB000_01FC 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 “0xB000_01FC” to enable register protection.

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

Register	Offset	R/W	Description	Reset Value
SYS_REGWPCTL	SYS_BA+0x1FC	R/W	Register Write-protection Control Register	0x0000_0000

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

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

6.3 Clock Controller (CLK_CTL)

6.3.1 Overview

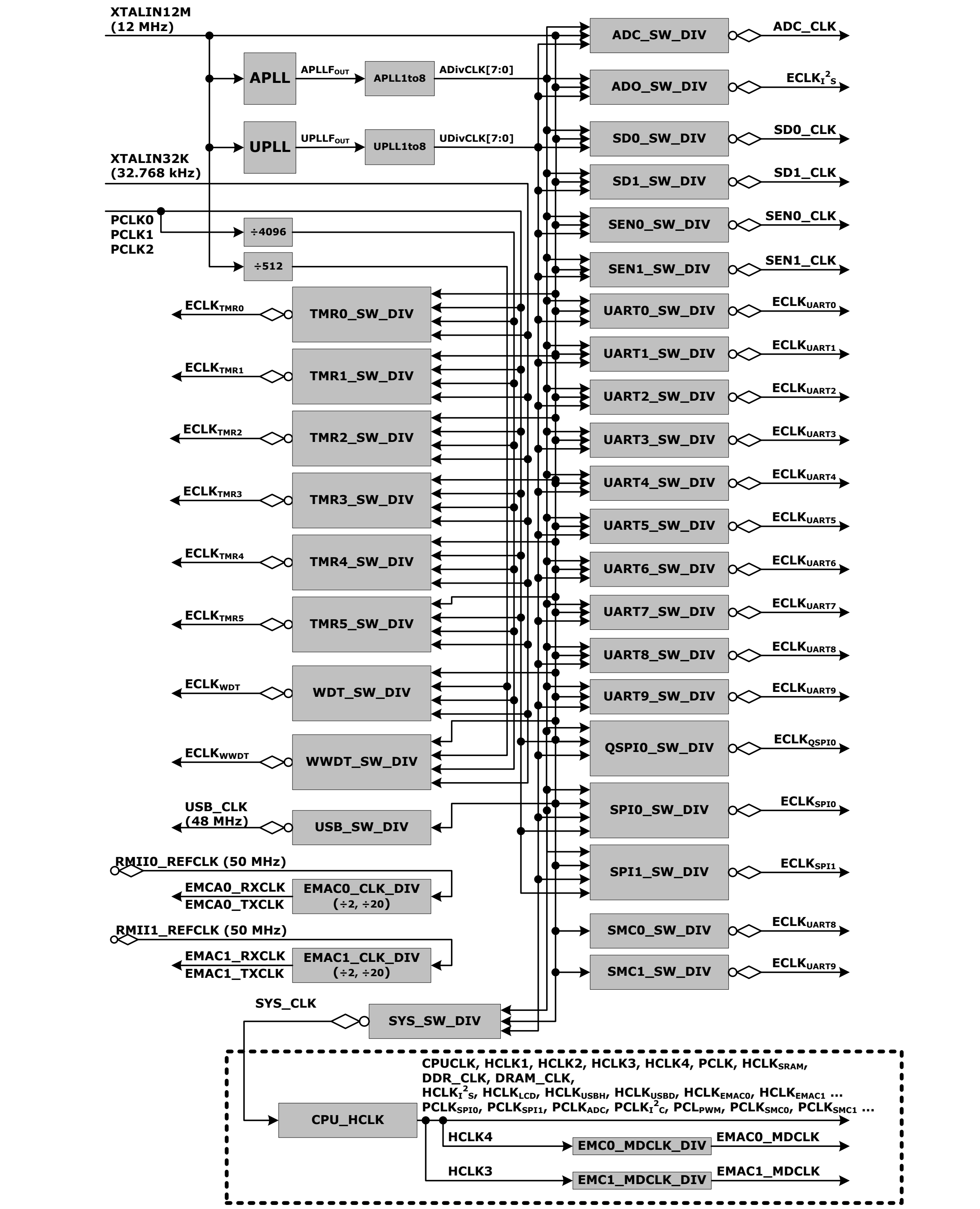
The clock controller generates all clocks for Video, Audio, CPU, system bus and all functionalities. This chip includes two PLL modules. The clock source for each functionality comes from the PLL, or from the external crystal input directly. For each clock there is a bit on the CLKEN register to control the clock ON or OFF individually, and the divider setting is in the CLK_DIVCTL register. The register can also be used to control the clock enable or disable for power control.

6.3.2 Features

- Supports two PLLs, up to 500 MHz, for high performance system operation
- External 12 MHz high speed crystal input for precise timing operation
- External 32.768 kHz low speed crystal input for RTC function and low speed clock source

6.3.3 Block Diagram

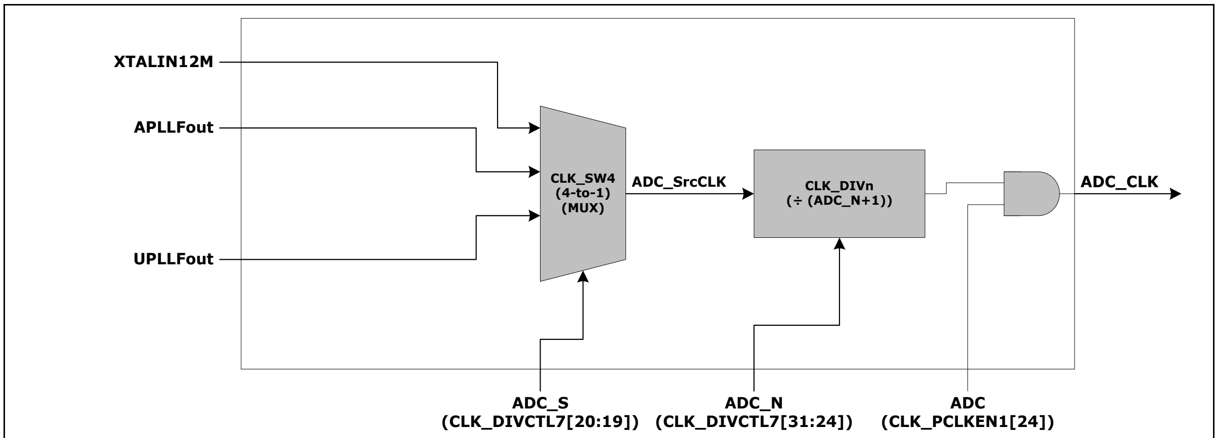
6.3.3.1 Clock Controller Top View



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

Figure 6.3-1 Clock Controller Block Diagram

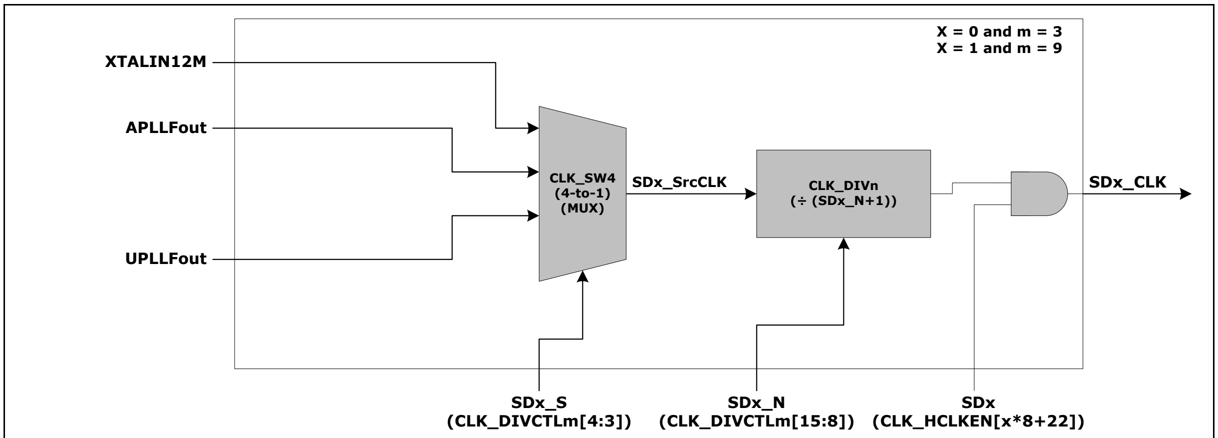
6.3.3.2 ADC Controller Clock Divider



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

Figure 6.3-2 ADC Controller Clock Divider Block Diagram

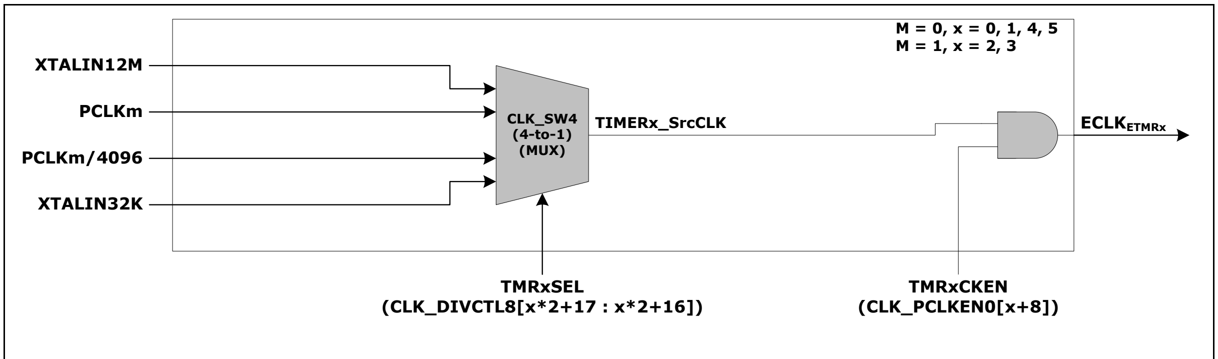
6.3.3.3 SD Card Host Controller Clock Divider



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

Figure 6.3-3 SD Card Host Controller Clock Divider Block Diagram

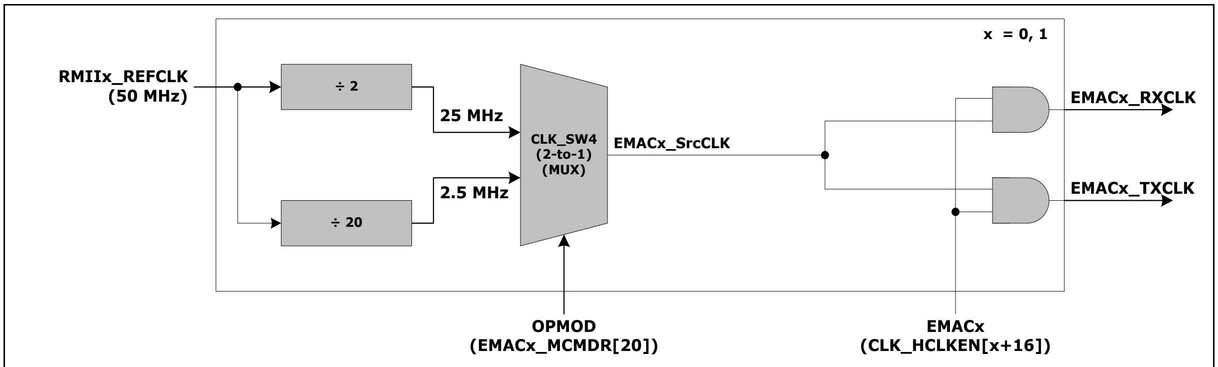
6.3.3.4 Timer Clock Divider



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

Figure 6.3-4 Timer Clock Divider Clock Diagram

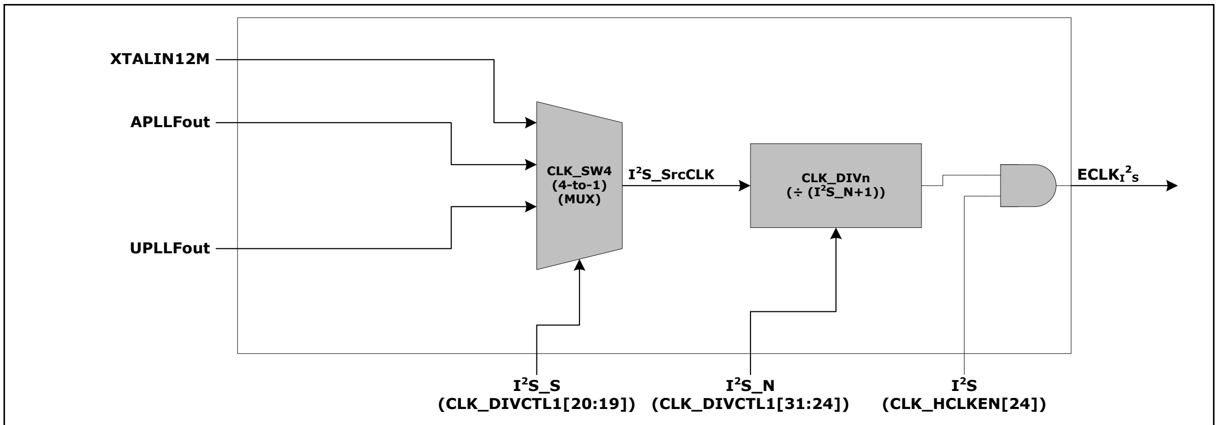
6.3.3.5 Ethernet MAC Controller Clock Divider



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

Figure 6.3-5 Ethernet MAC Controller Clock Divider Block Diagram

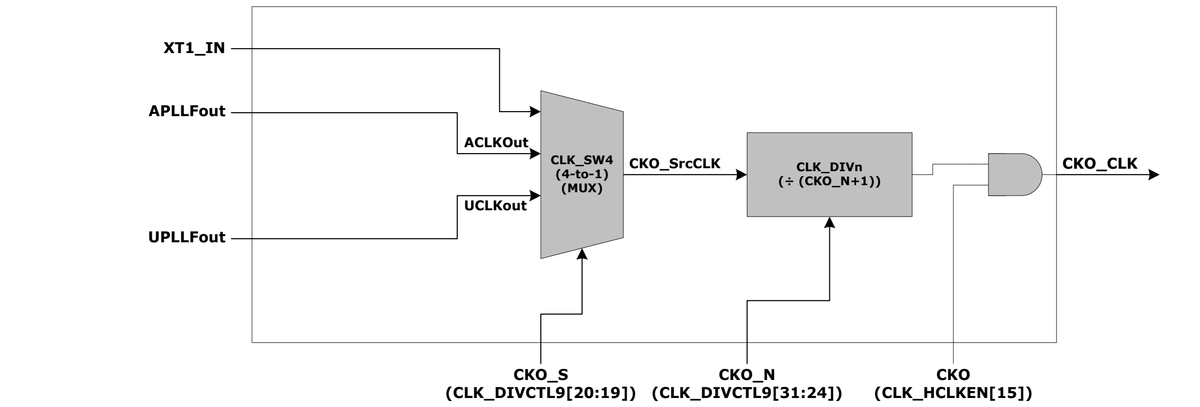
6.3.3.6 I^{2}S Controller Clock Divider



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

Figure 6.3-6 I^{2}S Controller Clock Divider Block Diagram

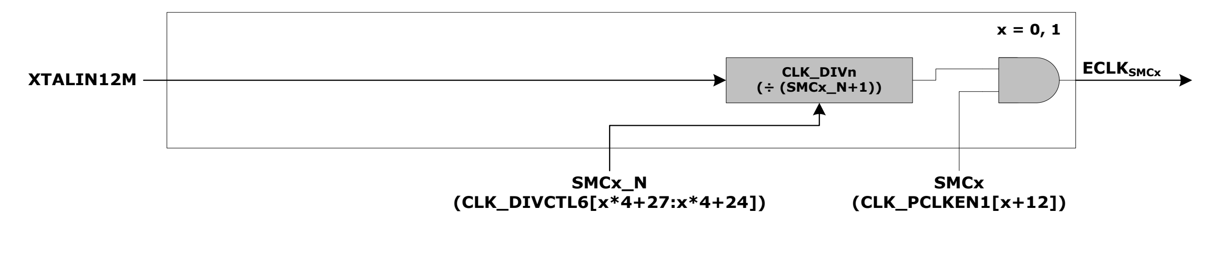
6.3.3.7 Reference Clock Output Divider



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

Figure 6.3-7 Reference Clock Output Divider Block Diagram

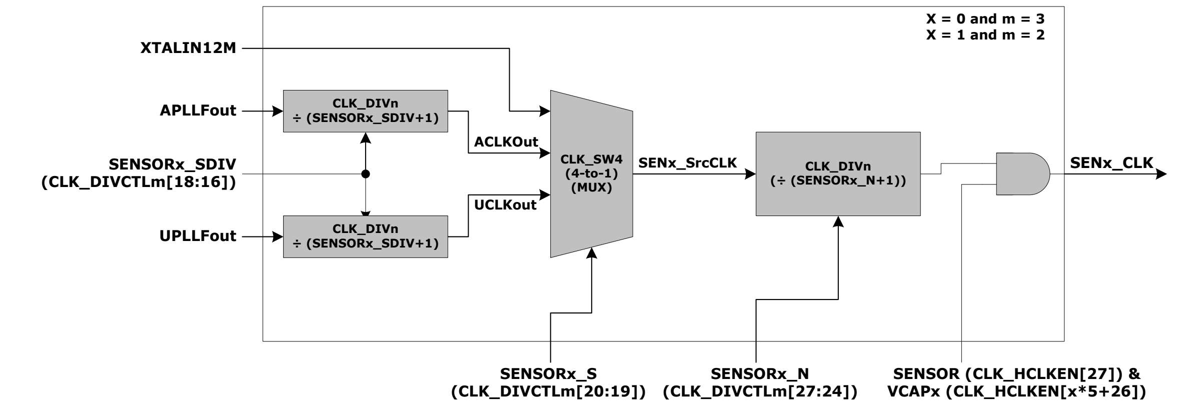
6.3.3.8 Smart Card Host Controller Clock Divider



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

Figure 6.3-8 Smart Card Host Controller Clock Divider Block Diagram

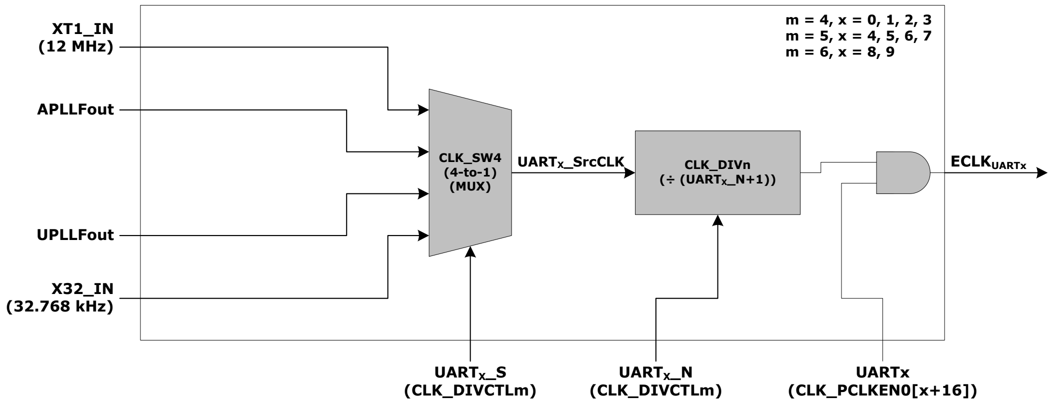
6.3.3.9 CMOS Sensor Clock Divider



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

Figure 6.3-9 CMOS Sensor Controller Divider Block Diagram

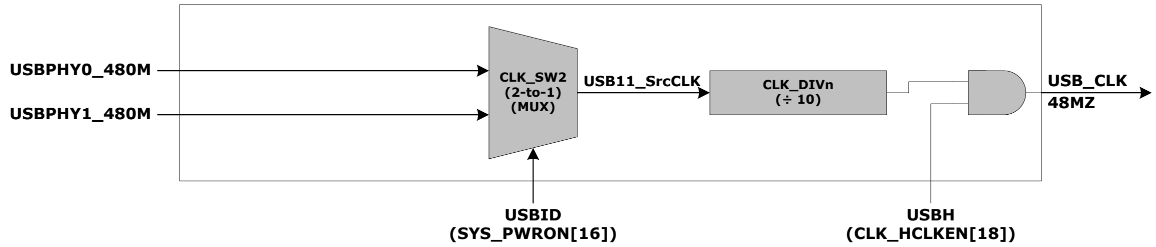
6.3.3.10 UART Clock Divider



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

Figure 6.3-10 UART Clock Divider Block Diagram

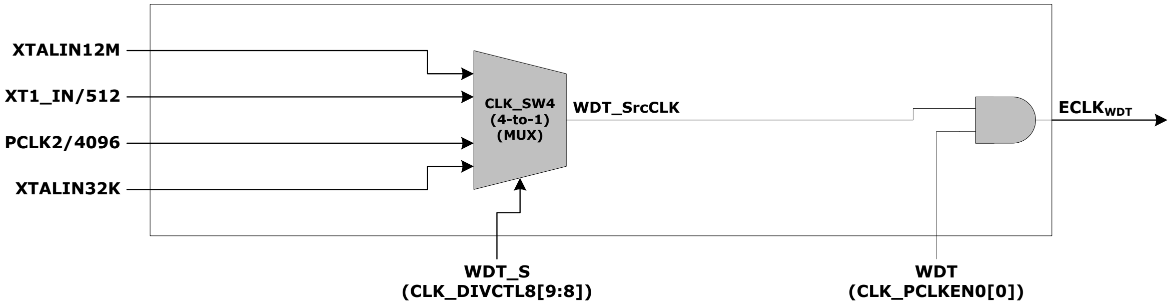
6.3.3.11 USB 1.1 Host 48 MHz Clock Divider



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

Figure 6.3-11 USB 1.1 Host Controller 48 MHz Clock Divider Block Diagram

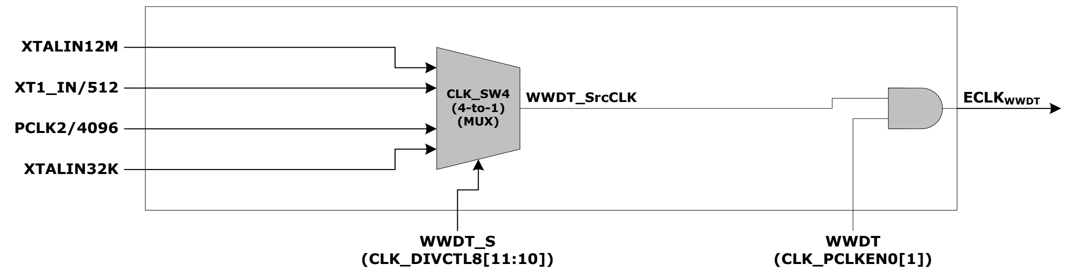
6.3.3.12 Watchdog Timer Clock Divider



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

Figure 6.3-12 Watchdog Timer Clock Divider Block Diagram

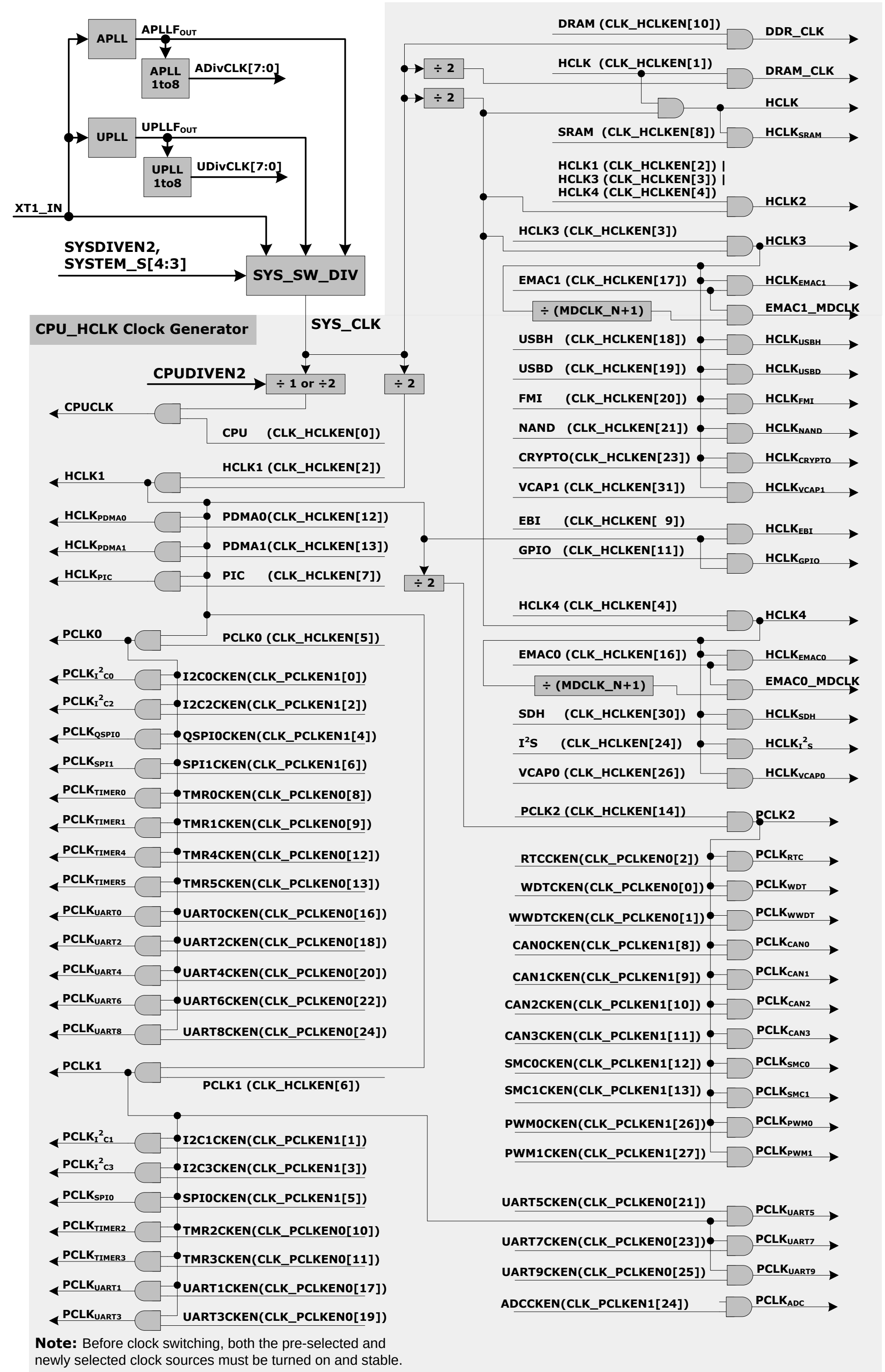
6.3.3.13 Windowed Watchdog Timer Clock Divider



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

Figure 6.3-13 Windowed Watchdog Timer Clock Divider Block Diagram

6.3.3.14 CPU_HCLK Clock Generator



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

Figure 6.3-14 CPU_HCLK Clock Generator Block Diagram

6.3.4 Functional Description

6.3.4.1 Power Management

This chip provides four power management scenarios, including Power-down, Idle and Normal Operating modes, to manage the power consumption. The peripheral clocks can be Enabled / Disabled individually by controlling the corresponding bit in CLKSEL control register. User can turn-off the unused modules' clock for power saving.

6.3.4.2 Normal Operating Mode

In this mode, CPU runs normally and clocks of all functionalities are on. The clock frequency of CPU, DRAM, AHB peripherals and APB peripherals are 300 MHz, 150 MHz, 150 MHz and 75 MHz, respectively.

6.3.4.3 Idle Mode

When CPU is not busy, user can put ARM926EJ-S™ processor into a low-power state by the wait for interrupt instruction:

MCR p15, 0, <Rd>, c7, c0, 4

This instruction switches the ARM926EJ-S™ processor into a low-power state until either an interrupt (IRQ or FIQ) or a debug request occurs.

In this mode, the clocks of all functionalities are on. The clock frequency of DRAM, AHB peripherals and APB peripherals are 150 MHz, 150 MHz and 75 MHz.

6.3.4.4 Power-down Mode

To reduce power consumption further, user could put the chip into Power-down mode by clearing XTAL_EN (CLK_PMCON[0]) to 0 before waiting for interrupt instruction:

MCR p15, 0, <Rd>, c7, c0, 4

In this mode, all clocks (clocks for all functionalities, CPU and the HXT (Ext. Crystall Osc. 12 MHz) stop, except LXT (Ext. Crystal Osc. 32.768 kHz), with SRAM retention.

The mechanisms shown below could wake chip up from Power-down mode:

- EINT0, EINT1, EINT2 or EINT3 (External Interrupt) pin toggled.
- GPIO pin toggled.
- Timer 0/1/2/3/4/5 timeout or capture interrupt is active.
- WDT time-out interrupt is active.
- RTC alarm or relative alarm interrupt is active.
- UART 0/1/2/3/4/5/6/7/8/9
  - UARTx_nCTS pin toggleed (x is 0, 1, 2, 3, 4, 5, 6, 7, 8 or 9).
  - UARTx_RXD pin goes low level (x is 0, 1, 2, 3, 4, 5, 6, 7, 8 or 9).
  - Received data FIFO reached threshold.
  - Received data FIFO threshold time-out.
  - RS-485 address match (AAD Mode).
- I^{2}C slave mode address match.
- EMAC 0/1 received a Magic Packet.
- HSUSBD detected a VBUS change event or USB bus RESET/RESUME event.
- USB 1.1 host controller detected a connect/dis-connect/remote-wakeup event.

- CANx_RXD pin goes low level (x is 0, 1, 2 or 3).
- SDH detected card pulg/un-plug event or SDIO card interrupt.

6.3.5 Registers 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 = 0xB000_0200**				
CLK_PMCON	CLK_BA+0x000	R/W	Power Management Control Register	0xFFFF_FF03
CLK_HCLKEN	CLK_BA+0x010	R/W	AHB Devices Clock Enable Control Register	0x0000_4527
CLK_PCLKEN0	CLK_BA+0x018	R/W	APB Devices Clock Enable Control Register 0	0x0000_000X
CLK_PCLKEN1	CLK_BA+0x01C	R/W	APB Devices Clock Enable Control Register 1	0x0000_0000
CLK_DIVCTL0	CLK_BA+0x020	R/W	Clock Divider Control Register 0	0x0000_00XX
CLK_DIVCTL1	CLK_BA+0x024	R/W	Clock Divider Control Register 1	0x0000_0000
CLK_DIVCTL2	CLK_BA+0x028	R/W	Clock Divider Control Register 2	0x0000_1500
CLK_DIVCTL3	CLK_BA+0x02C	R/W	Clock Divider Control Register 3	0x0000_0000
CLK_DIVCTL4	CLK_BA+0x030	R/W	Clock Divider Control Register 4	0x0000_0000
CLK_DIVCTL5	CLK_BA+0x034	R/W	Clock Divider Control Register 5	0x0000_0000
CLK_DIVCTL6	CLK_BA+0x038	R/W	Clock Divider Control Register 6	0x0000_0000
CLK_DIVCTL7	CLK_BA+0x03C	R/W	Clock Divider Control Register 7	0x0000_0000
CLK_DIVCTL8	CLK_BA+0x040	R/W	Clock Divider Control Register 8	0x0000_0500
CLK_DIVCTL9	CLK_BA+0x044	R/W	Clock Divider Control Register 9	0x0000_0000
CLK_APLLCON	CLK_BA+0x060	R/W	APLL Control Register	0x1000_0018
CLK_UPLLCON	CLK_BA+0x064	R/W	UPLL Control Register	0xX000_0018
CLK_PLLSTBCNTR	CLK_BA+0x080	R/W	PLL Stable Counter and Test Clock Control Register	0x0000_1800

6.3.6 Register Description

Power Management Control Register (CLK_PMCON)

The chip clock source is from an external crystal. The crystal oscillator can be control on/off by the register XTAL_EN. When turn off the crystal, the chip into power down state. To avoid outputting an unstable clock to system, clock controller implements a pre-scalar counter. After the clock counter count pre-scalar x 256 crystal cycle, the clock controller starts to output the clock to system.

Register	Offset	R/W	Description	Reset Value
CLK_PMCON	CLK_BA+0x000	R/W	Power Management Control Register	0xFFFF_FF03

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PRESCALE							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PRESCALE							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**SEN1_OFF_ST**	**SEN0_OFF_ST**	**Reserved**		**XIN_CTL**	**XTAL_EN**

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:8]	**PRESCALE**	**Pre-scalar Counter** Assume the crystal is stable after the Pre-Scalar x 256 crystal cycles. Clock controller wouldn't output clock to system before the counter reaching (pre-scalar x 256).
[7:6]	**Reserved**	Reserved.
[5]	**SEN1_OFF_ST**	**Sensor 1 Clock Level on Clock Off State** 0 = Sensor 1 clock keep on low level. 1 = Sensor 1 clock keep on high level.
[4]	**SEN0_OFF_ST**	**Sensor Clock Level on Clock Off State** 0 = Sensor 0 clock keep on low level. 1 = Sensor 0 clock keep on high level.
[3:2]	**Reserved**	Reserved.
[1]	**XIN_CTL**	**Pre-scalar Counter Enable Bit** Crystal pre-divide control for Wake-up from power down mode The chip will delay 256 x pre-scalar cycles after the reset signal to wait the Crystal to stable 0 = The pre-scalar counter Disabled (assume the crystal is stable). 1 = The pre-scalar counter Enabled.

[0]	**XTAL_EN**	**Crystal (Power-down) Control** 0 = Crystal off (Power-down mode). 1 = Crystal on (Normal operating mode).

AHB Devices Clock Enable Control Register (CLK_HCLKEN)

Register	Offset	R/W	Description	Reset Value
CLK_HCLKEN	CLK_BA+0x010	R/W	AHB Devices Clock Enable Control Register	0x0000_4527

31	30	29	28	27	26	25	24
VCAP1	SD1	Reserved	Reserved	SENSOR	VCAP0	Reserved	I2S
23	22	21	20	19	18	17	16
CRYPTO	SD0	NAND	FMI	USBD	USBH	EMAC1	EMAC0
15	14	13	12	11	10	9	8
CKO	PCLK2	PDMA1	PDMA0	GPIO	SDIC	EBI	SRAM
7	6	5	4	3	2	1	0
TIC	PCLK1	PCLK0	HCLK4	HCLK3	HCLK1	HCLK	CPU

Bits	Description	
[31]	**VCAP1**	**CMOS Sensor Interface Controller 1 Clock Enable Bit** 0 = CMOS sensor interface controller 1 clock Disabled. 1 = CMOS sensor interface controller 1 clock Enabled.
[30]	**SD1**	**SD Card Controller 1 Clock Enable Bit** 0 = SD card controller 1 clock Disabled. 1 = SD card controller 1 clock Enabled.
[29:28]	**Reserved**	Reserved.
[27]	**SENSOR**	**CMOS Sensor Reference Clock Output Enable Bit** 0 = CMOS sensor reference clock output Disabled. 1 = CMOS sensor reference clock output Enabled. **Note1:** The reference clock output for CMOS sensor interface 0 only Enabled when both VCAP0 and SENSOR Enabled. **Note2:** The reference clock output for CMOS sensor interface 1 only Enabled when both VCAP1 and SENSOR Enabled.
[26]	**VCAP0**	**CMOS Sensor Interface Controller 0 Clock Enable Bit** 0 = CMOS sensor interface controller 0 clock Disabled. 1 = CMOS sensor interface controller 0 clock Enabled.
[25]	**Reserved**	Reserved.
[24]	**I2S**	**I^{2}S Controller Clock Enable Bit** 0 = I^{2}S controller clock Disabled. 1 = I^{2}S controller clock Enabled.
[23]	**CRYPTO**	**Crypto Engine Clock Enable Bit** 0 = Crypto engine clock Disabled. 1 = Crypto engine clock Enabled.

[22]	**SD0**	**SD Card Controller 0 Clock Enable Bit** 0 = SD card controller 0 clock Disabled. 1 = SD card controller 0 clock Enabled.
[21]	**NAND**	**NAND Engine Clock Enable Bit** 0 = NAND controller clock Disabled. 1 = NAND controller clock Enabled.
[20]	**FMI**	**FMI Controller Clock Enable Bit** 0 = FMI controller clock Disabled. 1 = FMI controller clock Enabled.
[19]	**USBD**	**USB Device Controller Clock Enable Bit** 0 = USB device controller clock Disabled. 1 = USB device controller clock Enabled.
[18]	**USBH**	**USB Host Controller Clock Enable Bit** 0 = USB host controller clock Disabled. 1 = USB host controller clock Enabled.
[17]	**EMAC1**	**Ethernet MAC Controller 1 Clock Enable Bit** 0 = Ethernet MAC controller 1 clock Disabled. 1 = Ethernet MAC controller 1 clock Enabled.
[16]	**EMAC0**	**Ethernet MAC Controller 0 Clock Enable Bit** 0 = Ethernet MAC controller 0 clock Disabled. 1 = Ethernet MAC controller 0 clock Enabled.
[15]	**CKO**	**Reference Clock Output Enable Bit** 0 = Reference clock output Disabled. 1 = Reference clock output Enabled.
[14]	**PCLK2**	**Internal APB-2 Bus Clock Enable Bit** 0 = Internal APB-2 bus clock Disabled. 1 = Internal APB-2 bus clock Enabled.
[13]	**PDMA1**	**PDMA 1 Clock Enable Bit** 0 = PDMA 1 clock Disabled. 1 = PDMA 1 clock Enabled.
[12]	**PDMA0**	**PDMA 0 Clock Enable Bit** 0 = PDMA 0 clock Disabled. 1 = PDMA 0 clock Enabled.
[11]	**GPIO**	**GPIO Clock Enable Bit** 0 = GPIO clock Disabled. 1 = GPIO clock Enabled.
[10]	**SDIC**	**SDIC Clock Enable Bit** 0 = DDR clock Disabled. 1 = DDR clock Enabled.
[9]	**EBI**	**EBI Controller Clock Enable Bit** 0 = EBI controller clock Disabled. 1 = EBI controller clock Enabled.

[8]	**SRAM**	**SRAM Controller Clock Enable Bit** 0 = SRAM controller clock Disabled. 1 = SRAM controller clock Enabled.
[7]	**TIC**	**TIC Clock Enable Bit** 0 = TIC clock Disabled. 1 = TIC clock Enabled.
[6]	**PCLK1**	**Internal APB-1 Bus Clock Enable Bit** 0 = Internal APB-1 bus clock Disabled. 1 = Internal APB-1 bus clock Enabled.
[5]	**PCLK0**	**Internal APB-0 Bus Clock Enable Bit** 0 = Internal APB-1 bus clock Disabled. 1 = Internal APB-1 bus clock Enabled.
[4]	**HCLK4**	**Internal AHB-4 Bus Clock Enable Bit** 0 = Internal AHB-4 bus clock Disabled. 1 = Internal AHB-4 bus clock Enabled.
[3]	**HCLK3**	**Internal AHB-3 Bus Clock Enable Bit** 0 = Internal AHB-3 bus clock Disabled. 1 = Internal AHB-3 bus clock Enabled.
[2]	**HCLK1**	**Internal AHB-1 Bus Clock Enable Bit** 0 = Internal AHB-1 bus clock Disabled. 1 = Internal AHB-1 bus clock Enabled.
[1]	**HCLK**	**Internal AHB Bus Clock Enable Bit** 0 = Internal AHB bus clock Disabled. 1 = Internal AHB bus clock Enabled.
[0]	**CPU**	**ARM926EJ-S™ CPU Clock Enable Bit** 0 = ARM926EJ-S™ CPU clock Disabled. 1 = ARM926EJ-S™ CPU clock Enabled.

APB Devices Clock Enable Control Register 0 (CLK_PCLKEN0)

Register	Offset	R/W	Description	Reset Value
CLK_PCLKEN0	CLK_BA+0x018	R/W	APB Devices Clock Enable Control Register 0	0x0000_000X

31	30	29	28	27	26	25	24
Reserved						UART9CKEN	UART8CKEN
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		TMR5CKEN	TMR4CKEN	TMR3CKEN	TMR2CKEN	TMR1CKEN	TMR0CKEN
7	6	5	4	3	2	1	0
Reserved				Reserved	RTCCKEN	WWDTCKEN	WDTCKEN

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25]	**UART9CKEN**	**UART 9 Clock Enable Bit** 0 = UART 9 clock Disabled. 1 = UART 9 clock Enabled.
[24]	**UART8CKEN**	**UART 8 Clock Enable Bit** 0 = UART 8 clock Disabled. 1 = UART 8 clock Enabled.
[23]	**UART7CKEN**	**UART 7 Clock Enable Bit** 0 = UART 7 clock Disabled. 1 = UART 7 clock Enabled.
[22]	**UART6CKEN**	**UART 6 Clock Enable Bit** 0 = UART 6 clock Disabled. 1 = UART 6 clock Enabled.
[21]	**UART5CKEN**	**UART 5 Clock Enable Bit** 0 = UART 5 clock Disabled. 1 = UART 5 clock Enabled.
[20]	**UART4CKEN**	**UART 4 Clock Enable Bit** 0 = UART 4 clock Disabled. 1 = UART 4 clock Enabled.
[19]	**UART3CKEN**	**UART 3 Clock Enable Bit** 0 = UART 3 clock Disabled. 1 = UART 3 clock Enabled.

[18]	**UART2CKEN**	**UART 2 Clock Enable Bit** 0 = UART 2 clock Disabled. 1 = UART 2 clock Enabled.
[17]	**UART1CKEN**	**UART 1 Clock Enable Bit** 0 = UART 1 clock Disabled. 1 = UART 1 clock Enabled.
[16]	**UART0CKEN**	**UART 0 Clock Enable Bit** 0 = UART 0 clock Disabled. 1 = UART 0 clock Enabled.
[15:14]	**Reserved**	Reserved.
[13]	**TMR5CKEN**	**Timer 5 Clock Enable Bit** 0 = Timer 5 clock Disabled. 1 = Timer 5 clock Enabled.
[12]	**TMR4CKEN**	**Timer 4 Clock Enable Bit** 0 = Timer 4 clock Disabled. 1 = Timer 4 clock Enabled.
[11]	**TMR3CKEN**	**Timer 3 Clock Enable Bit** 0 = Timer 3 clock Disabled. 1 = Timer 3 clock Enabled.
[10]	**TMR2CKEN**	**Timer 2 Clock Enable Bit** 0 = Timer 2 clock Disabled. 1 = Timer 2 clock Enabled.
[9]	**TMR1CKEN**	**Timer 1 Clock Enable Bit** 0 = Timer 1 clock Disabled. 1 = Timer 1 clock Enabled.
[8]	**TMR0CKEN**	**Timer 0 Clock Enable Bit** 0 = Timer 0 clock Disabled. 1 = Timer 0 clock Enabled.
[7:3]	**Reserved**	Reserved.
[2]	**RTCCKEN**	**RTC Clock Enable Bit** 0 = RTC clock Disabled. 1 =RTC clock Enabled.
[1]	**WWDTCKEN**	**Windowed Watch-dog Clock Enable Bit** 0 = Windowed Watch-dog clock Disabled. 1 = Windowed Watch-dog clock Enabled.
[0]	**WDTCKEN**	**Watch-dog Clock Enable Bit** 0 = Watch-dog clock Disabled. 1 = Watch-dog clock Enabled. **Note:** If WDT default Enabled (WDTON(SYS_PWRON[3])=1), this bit is read-only and read back value is always 1.

APB Devices Clock Enable Control Register 1 (CLK_PCLKEN1)

Register	Offset	R/W	Description	Reset Value
CLK_PCLKEN1	CLK_BA+0x01C	R/W	APB Devices Clock Enable Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				PWM1CKEN	PWM0CKEN	Reserved	ADCCKEN
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved		SMC1CKEN	SMC0CKEN	CAN3CKEN	CAN2CKEN	CAN1CKEN	CAN0CKEN
7	6	5	4	3	2	1	0
Reserved	SPI1CKEN	SPI0CKEN	QSPI0CKEN	I2C3CKEN	I2C2CKEN	I2C1CKEN	I2C0CKEN

Bits	Description	
[31:26]	**Reserved**	Reserved.
[27]	**PWM1CKEN**	**PWM 1 Clock Enable Bit** 0 = PWM 1 clock Disabled. 1 = PWM 1 clock Enabled.
[26]	**PWM0CKEN**	**PWM 0 Clock Enable Bit** 0 = PWM 0 clock Disabled. 1 = PWM 0 clock Enabled.
[25]	**Reserved**	Reserved.
[24]	**ADCCKEN**	**ADC Controller Clock Enable Bit** 0 = ADC controller clock Disabled. 1 = ADC controller clock Enabled.
[23:14]	**Reserved**	Reserved.
[13]	**SMC1CKEN**	**Smart Card Interface 1 Clock Enable Bit** 0 = Smart Card interface 1 clock Disabled. 1 = Smart Card interface 1 clock Enabled.
[12]	**SMC0CKEN**	**Smart Card Interface 0 Clock Enable Bit** 0 = Smart Card interface 0 clock Disabled. 1 = Smart Card interface 0 clock Enabled.
[11]	**CAN3CKEN**	**CAN 3 Clock Enable Bit** 0 = CAN 3 clock Disabled. 1 = CAN 3 clock Enabled.
[10]	**CAN2CKEN**	**CAN 2 Clock Enable Bit** 0 = CAN 2 clock Disabled. 1 = CAN 2 clock Enabled.

[9]	**CAN1CKEN**	**CAN 1 Clock Enable Bit** 0 = CAN 1 clock Disabled. 1 = CAN 1 clock Enabled.
[8]	**CAN0CKEN**	**CAN 0 Clock Enable Bit** 0 = CAN 0 clock Disabled. 1 = CAN 0 clock Enabled.
[7]	**Reserved**	Reserved.
[6]	**SPI1CKEN**	**SPI 1 Clock Enable Bit** 0 = SPI 1 clock Disabled. 1 = SPI 1 clock Enabled.
[5]	**SPI0CKEN**	**SPI 0 Clock Enable Bit** 0 = SPI 0 clock Disabled. 1 = SPI 0 clock Enabled.
[4]	**QSPI0CKEN**	**QSPI 0 Clock Enable Bit** 0 = QSPI 0 clock Disabled. 1 = QSPI 0 clock Enabled.
[3]	**I2C3CKEN**	**I^2C 3 Clock Enable Bit** 0 = I^2C 3 clock Disabled. 1 = I^2C 3 clock Enabled.
[2]	**I2C2CKEN**	**I^2C 2 Clock Enable Bit** 0 = I^2C 2 clock Disabled. 1 = I^2C 2 clock Enabled.
[1]	**I2C1CKEN**	**I^2C 1 Clock Enable Bit** 0 = I^2C 1 clock Disabled. 1 = I^2C 1 clock Enabled.
[0]	**I2C0CKEN**	**I^2C 0 Clock Enable Bit** 0 = I^2C 0 clock Disabled. 1 = I^2C 0 clock Enabled.

Clock Divider Control Register 0 (CLK_DIVCTL0)

Register	Offset	R/W	Description	Reset Value
CLK_DIVCTL0	CLK_BA+0x020	R/W	Clock Divider Control Register 0	0x0000_00XX

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							CPUDIV2EN
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							SYSDIV2EN
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			SYSTEM_S		Reserved		

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**CPUDIV2EN**	**CPU Clock Divided by 2 Enable Bit** This field defines if CPUCLK for ARM926EJ-S™ CPU is SYS_CLK devided by 2 or not. 0 = The frequency of CPUCLK is equal to SYS_CLK. 1 = The frequency of CPUCLK is SYS_CLK devided by 2.
[15:9]	**Reserved**	Reserved.
[8]	**SYSDIV2EN**	**System Clock Divided by 2 Enable Bit** This field defines if SYS_CLK is SYSTEM_SrcCLK devided by 2 or not. 0 = The frequency of SYS_CLK is equal to SYSTEM_SrcCLK. 1 = The frequency of SYS_CLK is SYSTEM_SrcCLK devided by 2.
[7:5]	**Reserved**	Reserved.
[4:3]	**SYSTEM_S**	**System Clock Source Selection** This field selects which clock is used to be the source of system clock SYS_CLK. 00 = SYSTEM_SrcCLK is from XIN. 01 = Reserved. 10 = SYSTEM_SrcCLK is from APLLFout. 11 = SYSTEM_SrcCLK is from UPLLFout.
[2:0]	**Reserved**	Reserved.

Clock Divider Control Register 1 (CLK_DIVCTL1)

Register	Offset	R/W	Description	Reset Value
CLK_DIVCTL1	CLK_BA+0x024	R/W	Clock Divider Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
I2S_N							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved			**I2S_S**		**Reserved**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:24]	**I2S_N**	**I^2S Controller Clock Divider** This field defines the clock divide number for clock divider to generate the engine clock for I^2S controller. The actual clock divide number is (I2S_N + 1). So, ECLKi2s = I2S_SrcCLK / (I2S_N + 1).
[23:21]	**Reserved**	Reserved.
[20:19]	**I2S_S**	**I^2S Controller Clock Source Selection** This field selects which clock is used to be the source of engine clock for I^2S controller. 00 = I2S_SrcCLK is from XIN. 01 = Reserved. 10 = I2S_SrcCLK is from ACLKOut. 11 = I2S_SrcCLK is from UCLKOut.
[18:0]	**Reserved**	Reserved.

Clock Divider Control Register 2 (CLK_DIVCTL2)

Register	Offset	R/W	Description	Reset Value
CLK_DIVCTL2	CLK_BA+0x028	R/W	Clock Divider Control Register 2	0x0000_1500

31	30	29	28	27	26	25	24
Reserved				SENSOR1_N			
23	22	21	20	19	18	17	16
Reserved			SENSOR1_S		SENSOR1_SDIV		
15	14	13	12	11	10	9	8
Reserved		SPI1_S		SPI0_S		QSPI0_S	
7	6	5	4	3	2	1	0
Reserved			USB_S		Reserved		

Bits	Description	
[31:28]	Reserved	Reserved.
[27:24]	SENSOR1_N	**Sensor 1 Clock Divider** This field defines the clock divide number for clock divider to generate the sensor 1 clock. The actual clock divide number is (SENSOR1_N + 1). So, SEN1_CLK = SEN1_SrcCLK / (SENSOR1_N + 1).
[23:21]	Reserved	Reserved.
[20:19]	SENSOR1_S	**Sensor 1 Clock Source Selection** This field selects which clock is used to be the source of sensor 1 clock. 00 = SEN1_SrcCLK is from XIN. 01 = Reserved. 10 = SEN1_SrcCLK is from ACLKOut. 11 = SEN1_SrcCLK is from UCLKOut.
[18:16]	SENSOR1_SDIV	**Sensor 1 Source Clock Divider** This field defines the source clock divide number for clock divider of APLL and UPLL output. This field only takes effect while the SENSOR1_S (CLK_DIVCTL3[20:19]) is 2’b10 (APLL) or 2’b11 (UPLL). If SENSOR1_S (CLK_DIVCTL3[20:19]) is 2’b10, ACLKOut = APLLFout ÷ (SENSOR1_SDIV + 1). If SENSOR1_S (CLK_DIVCTL3[20:19]) is 2’b11, UCLKOut = UPLLFout ÷ (SENSOR1_SDIV + 1).
[15:14]	Reserved	Reserved.

[13:12]	**SPI1_S**	**SPI 1 Engine Clock Source Selection** This field selects which clock is used to be the source of engine clock for SPI 1. 00 = SPI1_SrcCLK is from XIN. 01 = SPI1_SrcCLK is from PCLK0. 10 = SPI1_SrcCLK is from ACLKOut. 11 = SPI1_SrcCLK is from UCLKOut.
[11:10]	**SPI0_S**	**SPI 0 Engine Clock Source Selection** This field selects which clock is used to be the source of engine clock for SPI 0. 00 = SPI0_SrcCLK is from XIN. 01 = SPI0_SrcCLK is from PCLK1. 10 = SPI0_SrcCLK is from ACLKOut. 11 = SPI0_SrcCLK is from UCLKOut.
[9:8]	**QSPI0_S**	**QSPI 0 Engine Clock Source Selection** This field selects which clock is used to be the source of engine clock for QSPI 0. 00 = QSPI0_SrcCLK is from XIN. 01 = QSPI0_SrcCLK is from PCLK0. 10 = QSPI0_SrcCLK is from ACLKOut. 11 = QSPI0_SrcCLK is from UCLKOut.
[7:5]	**Reserved**	Reserved.
[4:3]	**USB_S**	**USB 1.1 Engine Clock Source Selection** This field selects which clock is used to be the source of 48 MHz clock for USB 1.1 host controller. 00 = Reserved. 01 = Reserved. 10 = USB11_SrcCLK is from 480 MHz outputted by USB PHY 0. 11 = USB11_SrcCLK is from 480 MHz outputted by USB PHY 1.
[2:0]	**Reserved**	Reserved.

Clock Divider Control Register 3 (CLK_DIVCTL3)

Register	Offset	R/W	Description	Reset Value
CLK_DIVCTL3	CLK_BA+0x02C	R/W	Clock Divider Control Register 3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				**SENSOR0_N**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved			**SENSOR0_S**		**SENSOR0_SDIV**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SD0_N							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**SD0_S**		**Reserved**		

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:24]	**SENSOR0_N**	**Sensor 0 Clock Divider** This field defines the clock divide number for clock divider to generate the sensor 0 clock. The actual clock divide number is (SENSOR0_N + 1). So, SEN0_CLK = SEN0_SrcCLK / (SENSOR0_N + 1).
[33:21]	**Reserved**	Reserved.
[20:19]	**SENSOR0_S**	**Sensor 0 Clock Source Selection** This field selects which clock is used to be the source of sensor 0 clock. 00 = SEN0_SrcCLK is from XIN. 01 = Reserved. 10 = SEN0_SrcCLK is from ACLKOut. 11 = SEN0_SrcCLK is from UCLKOut.
[18:16]	**SENSOR0_SDIV**	**Sensor 0 Source Clock Divider** This field defines the source clock divide number for clock divider of APLL and UPLL output. This field only takes effect while the SENSOR0_S (CLK_DIVCTL3[20:19]) is 2'b10 (APLL) or 2'b11 (UPLL). If SENSOR0_S (CLK_DIVCTL3[20:19]) is 2'b10, ACLKOut = APLLFout ÷ (SENSOR0_SDIV + 1). If SENSOR0_S (CLK_DIVCTL3[20:19]) is 2'b11, UCLKOut = UPLLFout ÷ (SENSOR0_SDIV + 1).
[15:8]	**SD0_N**	**SD Card Controller 0 Engine Clock Divider** This field defines the clock divide number for clock divider to generate the engine clock for SD card controller 0. The actual clock divide number is (SD0_N + 1). So, SD0_CLK = SD0_SrcCLK / (SD0_N + 1).

[7:5]	**Reserved**	Reserved.
[4:3]	**SD0_S**	**SD Card Controller 0 Engine Clock Source Selection** This field selects which clock is used to be the source of engine clock for SD card controller 0. 00: SD0_SrcCLK = XIN. 01: SD0_SrcCLK = Reserved. 10: SD0_SrcCLK = ACLKOut. 11: SD0_SrcCLK = UCLKOut.
[2:0]	**Reserved**	Reserved.

Clock Divider Control Register 4 (CLK_DIVCTL4)

Register	Offset	R/W	Description	Reset Value
CLK_DIVCTL4	CLK_BA+0x030	R/W	Clock Divider Control Register 4	0x0000_0000

31	30	29	28	27	26	25	24
UART3_N			UART3_S		Reserved		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
UART2_N			UART2_S		Reserved		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
UART1_N			UART1_S		Reserved		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UART0_N			UART0_S		Reserved		

Bits	Description	
[31:29]	**UART3_N**	**UART3 Engine Clock Divider** This field defines the clock divide number for clock divider to generate the engine clock for UART3. The actual clock divide number is (UART3_N + 1). So, ECLKuart3 = UART3_SrcCLK / (UART3_N + 1).
[28:27]	**UART3_S**	**UART3 Engine Clock Source Selection** This field selects which clock is used to be the source of engine clock for UART3 controller. 00 = UART3_SrcCLK is from XIN. 01 = UART3_SrcCLK is from LXT. 10 = UART3_SrcCLK is from ACLKOut. 11 = UART3_SrcCLK is from UCLKOut.
[26:24]	**Reserved**	Reserved.
[23:21]	**UART2_N**	**UART2 Engine Clock Divider** This field defines the clock divide number for clock divider to generate the engine clock for UART2. The actual clock divide number is (UART2_N + 1). So, ECLKuart2 = UART2_SrcCLK / (UART2_N + 1).
[20:19]	**UART2_S**	**UART2 Engine Clock Source Selection** This field selects which clock is used to be the source of engine clock for UART2 controller. 00 = UART2_SrcCLK is from XIN. 01 = UART2_SrcCLK is from LXT. 10 = UART2_SrcCLK is from ACLKOut. 11 = UART2_SrcCLK is from UCLKOut.
[18:16]	**Reserved**	Reserved.

[15:13]	**UART1_N**	**UART1 Engine Clock Divider** This field defines the clock divide number for clock divider to generate the engine clock for UART1. The actual clock divide number is (UART1_N + 1). So, ECLKuart1 = UART1_SrcCLK / (UART1_N + 1).
[12:11]	**UART1_S**	**UART1 Engine Clock Source Selection** This field selects which clock is used to be the source of engine clock for UART1 controller. 00 = UART1_SrcCLK is from XIN. 01 = UART1_SrcCLK is from LXT. 10 = UART1_SrcCLK is from ACLKOut. 11 = UART1_SrcCLK is from UCLKOut.
[10:8]	**Reserved**	Rserved
[7:5]	**UART0_N**	**UART0 Engine Clock Divider** This field defines the clock divide number for clock divider to generate the engine clock for UART0. The actual clock divide number is (UART0_N + 1). So, ECLKuart0 = UART0_SrcCLK / (UART0_N + 1).
[4:3]	**UART0_S**	**UART0 Engine Clock Source Selection** This field selects which clock is used to be the source of engine clock for UART0 controller. 00 = UART0_SrcCLK is from XIN. 01 = UART0_SrcCLK is from LXT. 10 = UART0_SrcCLK is from ACLKOut. 11 = UART0_SrcCLK is from UCLKOut.
[2:0]	**Reserved**	Reserved.

Clock Divider Control Register 5 (CLK_DIVCTL5)

Register	Offset	R/W	Description	Reset Value
CLK_DIVCTL5	CLK_BA+0x034	R/W	Clock Divider Control Register 5	0x0000_0000

31	30	29	28	27	26	25	24
UART7_N			UART7_S		Reserved		
23	22	21	20	19	18	17	16
UART6_N			UART6_S		Reserved		
15	14	13	12	11	10	9	8
UART5_N			UART5_S		Reserved		
7	6	5	4	3	2	1	0
UART4_N			UART4_S		Reserved		

Bits	Description	
[31:29]	UART7_N	**UART7 Engine Clock Divider** This field defines the clock divide number for clock divider to generate the engine clock for UART7. The actual clock divide number is (UART7_N + 1). So, ECLKuart7 = UART7_SrcCLK / (UART7_N + 1).
[28:27]	UART7_S	**UART7 Engine Clock Source Selection** This field selects which clock is used to be the source of engine clock for UART7 controller. 00 = UART7_SrcCLK is from XIN. 01 = UART7_SrcCLK is from LXT. 10 = UART7_SrcCLK is from ACLKOut. 11 = UART7_SrcCLK is from UCLKOut.
[26:24]	Reserved	Reserved.
[23:21]	UART6_N	**UART6 Engine Clock Divider** This field defines the clock divide number for clock divider to generate the engine clock for UART6. The actual clock divide number is (UART6_N + 1). So, ECLKuart6 = UART6_SrcCLK / (UART6_N + 1).
[20:19]	UART6_S	**UART6 Engine Clock Source Selection** This field selects which clock is used to be the source of engine clock for UART6 controller. 00 = UART6_SrcCLK is from XIN. 01 = UART6_SrcCLK is from LXT. 10 = UART6_SrcCLK is from ACLKOut. 11 = UART6_SrcCLK is from UCLKOut.
[18:16]	Reserved	Reserved.

[15:13]	**UART5_N**	**UART5 Engine Clock Divider** This field defines the clock divide number for clock divider to generate the engine clock for UART5. The actual clock divide number is (UART5_N + 1). So, ECLKuart5 = UART5_SrcCLK / (UART5_N + 1).
[12:11]	**UART5_S**	**UART5 Engine Clock Source Selection** This field selects which clock is used to be the source of engine clock for UART5 controller. 00 = UART5_SrcCLK is from XIN. 01 = UART5_SrcCLK is from LXT. 10 = UART5_SrcCLK is from ACLKOut. 11 = UART5_SrcCLK is from UCLKOut.
[10:8]	**Reserved**	Reserved.
[7:5]	**UART4_N**	**UART4 Engine Clock Divider** This field defines the clock divide number for clock divider to generate the engine clock for UART4. The actual clock divide number is (UART4_N + 1). So, ECLKuart4 = UART4_SrcCLK / (UART4_N + 1).
[4:3]	**UART4_S**	**UART4 Engine Clock Source Selection** This field selects which clock is used to be the source of engine clock for UART4 controller. 00 = UART4_SrcCLK is from XIN. 01 = UART4_SrcCLK is from LXT. 10 = UART4_SrcCLK is from ACLKOut. 11 = UART4_SrcCLK is from UCLKOut.
[2:0]	**Reserved**	Reserved.

Clock Divider Control Register 6 (CLK_DIVCTL6)

Register	Offset	R/W	Description	Reset Value
CLK_DIVCTL6	CLK_BA+0x038	R/W	Clock Divider Control Register 6	0x0000_0000

31	30	29	28	27	26	25	24
SMC1_N				**SMC0_N**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
UART9_N			**UART9_S**		**Reserved**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UART8_N			**UART8_S**		**Reserved**		

Bits	Description	
[30:28]	**SMC1_N**	**Smart Card 1 Engine Clock Divider** This field defines the clock divide number for clock divider to generate the engine clock for Smart card controller. The actual clock divide number is (SMC1_N + 1). So, ECLKsmc1 = XIN12M / (SMC1_N + 1).
[27:24]	**SMC0_N**	**Smart Card 0 Engine Clock Divider** This field defines the clock divide number for clock divider to generate the engine clock for Smart card controller. The actual clock divide number is (SMC0_N + 1). So, ECLKsmc0 = XIN12M / (SMC0_N + 1).
[23:16]	**Reserved**	Reserved.
[15:13]	**UART9_N**	**UART9 Engine Clock Divider** This field defines the clock divide number for clock divider to generate the engine clock for UART9. The actual clock divide number is (UART9_N + 1). So, ECLKuart9 = UART9_SrcCLK / (UART9_N + 1).
[12:11]	**UART9_S**	**UART9 Engine Clock Source Selection** This field selects which clock is used to be the source of engine clock for UART9 controller. 00 = UART9_SrcCLK is from XIN. 01 = UART9_SrcCLK is from LXT. 10 = UART9_SrcCLK is from ACLKOut. 11 = UART9_SrcCLK is from UCLKOut.
[10:8]	**Reserved**	Reserved.

[7:5]	**UART8_N**	**UART8 Engine Clock Divider** This field defines the clock divide number for clock divider to generate the engine clock for UART8. The actual clock divide number is (UART8_N + 1). So, ECLKuart8 = UART8_SrcCLK / (UART8_N + 1).
[4:3]	**UART8_S**	**UART8 Engine Clock Source Selection** This field selects which clock is used to be the source of engine clock for UART8 controller. 00 = UART8_SrcCLK is from XIN. 01 = UART8_SrcCLK is from LXT. 10 = UART8_SrcCLK is from ACLKOut. 11 = UART8_SrcCLK is from UCLKOut.
[2:0]	**Reserved**	Reserved.

Clock Divider Control Register 7 (CLK_DIVCTL7)

Register	Offset	R/W	Description	Reset Value
CLK_DIVCTL7	CLK_BA+0x03C	R/W	Clock Divider Control Register 7	0x0000_0000

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

Bits	Description	
[31:24]	**ADC_N**	**ADC Engine Clock Divider** This field defines the clock divide number for clock divider to generate the engine clock for ADC. The actual clock divide number is (ADC_N + 1). So, ADC_CLK = ADC_SrcCLK / (ADC_N + 1).
[23:21]	**Reserved**	Reserved.
[20:19]	**ADC_S**	**ADC Engine Clock Source Selection** This field selects which clock is used to be the source of engine clock for ADC controller. 00 = ADC_SrcCLK is from XIN. 01 = Reserved. 10 = ADC_SrcCLK is from APLLFOut. 11 = ADC_SrcCLK is from UPLLFOut.
[18:0]	**Reserved**	Reserved.

Clock Divider Control Register 8 (CLK_DIVCTL8)

Register	Offset	R/W	Description	Reset Value
CLK_DIVCTL8	CLK_BA+0x040	R/W	Clock Divider Control Register 8	0x0000_0500

31	30	29	28	27	26	25	24
Reserved				TMR5SEL		TMR4SEL	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TMR3SEL		TMR2SEL		TMR1SEL		TMR0SEL	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				WWDTSEL		WDTSEL	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MDCLKDIV							

Bits	Description	
[27:26]	**TMR5SEL**	**Timer 5 Engine Clock Source Selection** This field selects which clock is used to be the source of engine clock for Timer 5 controller. 00: TIMER5_SrcCLK = XIN. 01: TIMER5_SrcCLK = PCLK0. 10: TIMER5_SrcCLK = PCLK0/4096. 11: TIMER5_SrcCLK = 32.768 kHz.
[25:24]	**TMR4SEL**	**Timer 4 Engine Clock Source Selection** This field selects which clock is used to be the source of engine clock for Timer 4 controller. 00: TIMER4_SrcCLK = XIN. 01: TIMER4_SrcCLK = PCLK0. 10: TIMER4_SrcCLK = PCLK0/4096. 11: TIMER4_SrcCLK = 32.768 kHz.
[23:22]	**TMR3SEL**	**Timer 3 Engine Clock Source Selection** This field selects which clock is used to be the source of engine clock for Timer 3 controller. 00: TIMER3_SrcCLK = XIN. 01: TIMER3_SrcCLK = PCLK1. 10: TIMER3_SrcCLK = PCLK1/4096. 11: TIMER3_SrcCLK = 32.768 kHz.
[21:20]	**TMR2SEL**	**Timer 2 Engine Clock Source Selection** This field selects which clock is used to be the source of engine clock for Timer 2 controller. 00: TIMER2_SrcCLK = XIN. 01: TIMER2_SrcCLK = PCLK1. 10: TIMER2_SrcCLK = PCLK1/4096. 11: TIMER2_SrcCLK = 32.768 kHz.

[19:18]	**TMR1SEL**	**Timer 1 Engine Clock Source Selection** This field selects which clock is used to be the source of engine clock for Timer 1 controller. 00: TIMER1_SrcCLK = XIN. 01: TIMER1_SrcCLK = PCLK0. 10: TIMER1_SrcCLK = PCLK0/4096. 11: TIMER1_SrcCLK = 32.768 kHz.
[17:16]	**TMR0SEL**	**Timer 0 Engine Clock Source Selection** This field selects which clock is used to be the source of engine clock for Timer 0 controller. 00: TIMER0_SrcCLK = XIN. 01: TIMER0_SrcCLK = PCLK0. 10: TIMER0_SrcCLK = PCLK0/4096. 11: TIMER0_SrcCLK = 32.768 kHz.
[11:10]	**WWDTSEL**	**WWDT Engine Clock Source Selection** This field selects which clock is used to be the source of engine clock for WWDT controller. 00: WWDT_SrcCLK = XIN. 01: WWDT_SrcCLK = XIN/512. 10: WWDT_SrcCLK = PCLK2/4096. 11: WWDT_SrcCLK = 32.768 kHz.
[9:8]	**WDTSEL**	**WDT Engine Clock Source Selection** This field selects which clock is used to be the source of engine clock for WDT controller. 00: WDT_SrcCLK = XIN. 01: WDT_SrcCLK = XIN/512. 10: WDT_SrcCLK = PCLK2/4096. 11: WDT_SrcCLK = 32.768 kHz.
[7:0]	**MDCLKDIV**	**MII Management Interface Clock** This field defines the clock divide number for clock divider to generate the clock for MII management interface. The actual clock divide number is (MDCLK_N + 1). So, MDCLK = HCLK / (MDCLK_N + 1).

Clock Divider Control Register 9 (CLK_DIVCTL9)

Register	Offset	R/W	Description	Reset Value
CLK_DIVCTL9	CLK_BA+0x044	R/W	Clock Divider Control Register 9	0x0000_0000

31	30	29	28	27	26	25	24
CKO_N							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved			CKO_S		Reserved		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SD1_N							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			SD1_S		Reserved		

Bits	Description	
[31:24]	**CKO_N**	**Reference Clock Out Divide** This field defines the clock divide number for clock divider to generate the reference clock output The actual clock divide number is (CKO_N + 1). So, CKO_CLK = CKO_SrcCLK / (CKO_N + 1).
[23:21]	**Reserved**	Reserved.
[20:19]	**CKO_S**	**Reference Clock Out Source Selection** This field selects which clock is used to be the source of reference clock output. 00 = CKO_SrcCLK is from XIN. 01 = CKO_SrcCLK is from LXT. 10 = CKO_SrcCLK is from ACLKOut. 11 = CKO_SrcCLK is from UCLKOut.
[18:16]	**Reserved**	Reserved.
[15:8]	**SD1_N**	**SD Card Controller 1 Engine Clock Divider** This field defines the clock divide number for clock divider to generate the engine clock for SD card controller 1. The actual clock divide number is (SD1_N + 1). So, SD1_CLK = SD1_SrcCLK / (SD1_N + 1).
[7:5]	**Reserved**	Reserved.
[4:3]	**SD1_S**	**SD Card Controller 1 Engine Clock Source Selection** This field selects which clock is used to be the source of engine clock for SD card controller 1. 00 = SD1_SrcCLK is from XIN. 01 = Reserved. 10 = SD1_SrcCLK is from ACLKOut. 11 = SD1_SrcCLK is from UCLKOut.

[2:0]	**Reserved**	Reserved.

APLL Control Register (CLK_APLLCON), UPLL Control Register (CLK_UPLLCON)

Register	Offset	R/W	Description	Reset Value
CLK_APLLCON	CLK_BA+0x060	R/W	APLL Control Register	0x1000_0018
CLK_UPLLCON	CLK_BA+0x064	R/W	UPLL Control Register	0xX000_0018

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

Bits	Description	
[31]	**PLL_STB**	**PLL Stable Flag** 0 = PLL is not stable. 1 = PLL is stable (500us after PLL setting changed).
[30]	**RESETN**	**Reset Mode Enable Bit** 0 = PLL is in reset mode. 1 = PLL is in normal operation mode (Default).
[29]	**BYPASS**	**Bypass Mode Enable Bit** 0 = PLL is in normal operation mode (Default). 1 = PLL is in bypass mode.
[28]	**PD**	**Power Down Mode Enable Bit** 0 = PLL is in normal operation mode. 1 = PLL is in power down mode (Default).
[27:16]	**FRAC**	**PLL VCO Output Clock Feedback Divider Fraction Part** Set the fraction part (X) of feedback divider factor. Write a non-zero value to this field enables the fraction mode automatically. Please keep this field in 0x0 if don’t want to use the PLL fraction mode. The X = FRAC[11:0] / 2^{12}.
[15:13]	**OUT_DV**	**PLL Output Divider** Set the output divider factor (P) from 1 to 8. The P = OUT_DV[2:0] + 1.
[12:7]	**IN_DV**	**Reference Input Divider** Set the reference divider factor (M) from 1 to 64. The M = IN_DV[5:0] + 1.

[6:0]	**FB_DV**	**PLL VCO Output Clock Feedback Divider Integer Part** Set the feedback divider factor (N) from 1 to 128. The N = FB_DV[6:0] + 1.

The formula to calculate the PLL output frequency shown below:

$$F_{pllout} = 12\text{ MHz} \times \frac{N}{M \times P}$$

$$F_{vco} = 12\text{ MHz} \times \frac{N}{M}$$

$$200\text{ MHz} < F_{vco} < 500\ MHz$$

$$F_{pfd} = \frac{12\ MHz}{M} = \frac{F_{vco}}{N}$$

N	F_{pfd} Range
1	$11.0 \le F_{pfd} \le 80$
2	$7.0 \le F_{pfd} \le 80$
3	$5.0 \le F_{pfd} \le 80$
4	$4.0 \le F_{pfd} \le 80$
5	$3.5 \le F_{pfd} \le 80$
6	$3.0 \le F_{pfd} \le 80$
7 ~ 8	$2.5 \le F_{pfd} \le 80$
9 ~ 10	$3.5 \le F_{pfd} \le 80$
11 ~ 40	$3.0 \le F_{pfd} \le 80$
41 ~ 128	$2.5 \le F_{pfd} \le 80$

Table 6.3-1 The Mapping of N and F_{pfd} Range

PLL Stable Counter and Test Clock Control Register (CLK_PLLSTBCNTR)

Register	Offset	R/W	Description	Reset Value
CLK_PLLSTBCNTR	CLK_BA+0x080	R/W	PLL Stable Counter and Test Clock Control Register	0x0000_1800

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[15:0]	**PLLSTBCNT**	PLL Stable Counter

6.4 Advanced Interrupt Controller (AIC)

6.4.1 Overview

An interrupt can temporarily change the sequence of program execution to react to some specific events, such as power failure, watchdog timer timeout, transmit/receive requests from Ethernet MAC Controller, and so on. There are two interrupt types the CPU can process. The first type is the Fast Interrupt Request (FIQ) for servicing timing-critical events, and the second type is the Interrupt Request (IRQ) for servicing other general-purpose events. An FIQ interrupt occurrs when the signal *nFIQ* to the CPU is asserted, and a IRQ interrupt occurrs when the signal *nIRQ* to the CPU is asserted.

A FIQ interrupt has higher priority than an IRQ interrupt to be processed by CPU. An IRQ service routine in-process can be interrupted by a new coming FIQ interrupt; however, a FIQ service routine in-process cannot be interrupted by a new coming IRQ interrupt.

The Advanced Interrupt Controller (AIC) can process up to 64 interrupt sources. Currently, 62 interrupt sources are supported in the system. AIC assigns every interrupt source a unique source number. For example, the watchdog timer interrupt is assigned to source number 1, and window WDT interrupt is assigned to source number 2.

Every interrupt source can be configured to have one of eight priority levels, numbered from 0 to 7. Interrupt sources with priority level 0 have the highest priority, and interrupt sources with priority level 7 have the lowest priority. For those interrupt sources with the same priority levels, an interrupt source with a lower source number will have higher priority.

An interrupt request generated by an interrupt source with priority level 0 will become a FIQ interrupt to the CPU. An interrupt request generated by an interrupt source with priority levels from 1 to 7 will become a IRQ interrupt to the CPU.

Each interrupt source can be configured as disabled or enabled. An interrupt request from a disabled interrupt source is always ignored by AIC, no matter what its source number and priority level are.

AIC supports four trigger types for every interrupt source: high-level trigger, low-level trigger, rising-edge trigger, and falling-edge trigger.

6.4.2 Features

- AMBA APB interface
- 62 interrupt sources
- Configurable 8 priority levels for each interrupt source
- Configurable 4 trigger types for each interrupt source
- Configurable disabled/enabled status for each interrupt source
- Readable on the current logic value of each interrupt source
- Arbitration of interrupt requests from two or more interrupt sources
- Easy programming of interrupt service routines

6.4.3 Block Diagram

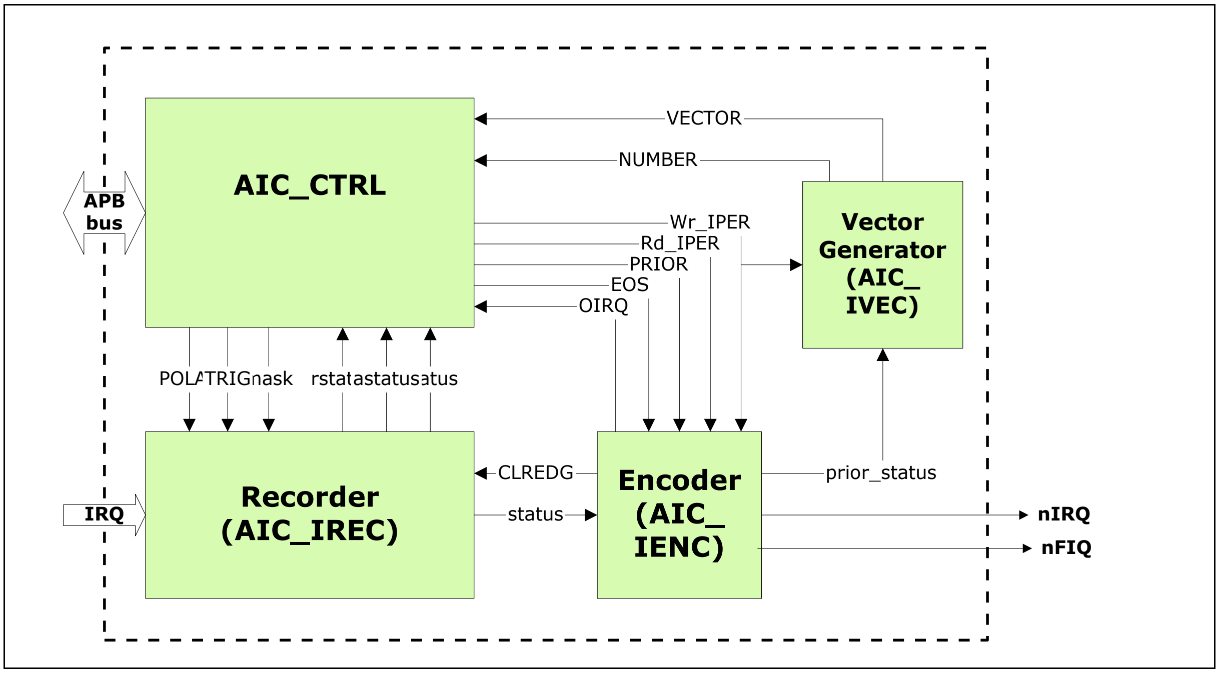


Figure 6.4-1 Advanced Interrupt Controller (AIC) Block Diagram

6.4.4 Basic Configuration

- Interrupt Disable/Enable Configuration
  - Disable or Enable interrupt sources on AIC_IE0 and AIC_IE1.
  - Enable interrupt sources on AIC_IEN0 and AIC_IEN1.
  - Disable interrupt sources on AIC_IDIS0 and AIC_IDIS1.
- Priority Level Configuration
  - Set priority level for every interrupt source on AIC_SRC00 ~ AIC_SRC15 (PL0[2:0], PL1[10:8], PL2[18:16], PL3[26:24]).
- Trigger Type Configuration
  - Set trigger type for every interrupt source on AIC_SRC00 ~ AIC_SRC15 (TT0[7:6], TT1[15:14], TT2[23:22], TT3[31:30]).

6.4.5 Functional Description

6.4.5.1 Hardware Interrupt Arbitration

A FIQ interrupt is activated by only one interrupt request at a time. If two or more interrupt requests simultaneously occur from various interrupt sources with priority level 0, AIC will recognize an interrupt source with the highest priority, and use it to activate a FIQ interrupt. All other requests are pending.

Similarly, an IRQ interrupt is activated by only one interrupt request at a time. If two or more interrupt requests simultaneously occur from various interrupt sources with priority levels from 1 to 7, AIC will recognize an interrupt source with the highest priority, and use it to activate an IRQ interrupt. All other requests are pending.

6.4.5.2 Priority Determination

Priority is determined by two characteristics of an interrupt source: the priority level and the source number. The priority level is configurable, but the source number is unique and fixedly assigned by the AIC hardware.

The following lists the priority determination rule:

1. An interrupt source with lower priority level has higher priority.
2. For two or more interrupt sources with the same priority level, the interrupt source with lower source number has higher priority.

6.4.5.3 Arbitration of FIQ Interrupt Requests

When two or more interrupt requests from various interrupt sources with priority level 0 occur at the same time, AIC recognizes the one with the highest priority, writes its source number into the FIQ Source Number Register (AIC_FIQNUM), and asserts the signal *nFIQ* (i.e., set *nFIQ* = 0) to the CPU.

6.4.5.4 Arbitration of IRQ Interrupt Requests

When two or more interrupt requests from various interrupt sources with priority levels from 1 to 7 occur at the same time, AIC recognizes the one with the highest priority, writes its source number into the IRQ Source Number Register (AIC_IRQNUM), and asserts the signal *nIRQ* (i.e., set *nIRQ* = 0) to the CPU.

6.4.5.5 Re-Arbitration of FIQ Interrupt Requests

Once AIC writes the register AIC_FIQNUM and asserts the signal *nFIQ*, it will keep this state unchanged, even if the original interrupt request activating this FIQ interrupt is released, or one or more new interrupt requests from other interrupt sources with higher priority are issued.

The only way to break this state is to write any value into the FIQ Reset Register (AIC_FIQRST).

When a value is written into the register AIC_FIQRST, the signal *nFIQ* is de-asserted (i.e., becomes 1). AIC then reviews all interrupt requests from the interrupt sources with priority level 0. If at least one interrupt request exists, AIC recognizes the one with the highest priority, writes its source number into the register AIC_FIQNUM, and asserts the signal *nFIQ* again.

If no such an interrupt request exists, the register AIC_FIQNUM keeps unchanged, the signal *nFIQ* stays de-asserted, and AIC waits for new interrupt requests occuring.

6.4.5.6 Re-Arbitration of IRQ Interrupt Requests

Once AIC writes the register AIC_IRQNUM and asserts the signal *nIRQ*, it will keep this state unchanged, even if the original interrupt request activating this IRQ interrupt is released, or one or more new interrupt requests from other interrupt sources with higher priority are issued.

The only way to break this state is to write any value into the IRQ Reset Register (AIC_IRQRST).

When a value is written into the register AIC_IRQRST, the signal *nIRQ* is de-asserted (i.e., becomes 1). AIC then reviews all interrupt requests from the interrupt sources with priority levels from 1 to 7. If at least one interrupt request exists, AIC recognizes the one with the highest priority, writes its source number into the register AIC_IRQNUM, and asserts the signal *nIRQ* again.

If no such an interrupt request exists, the register AIC_IRQNUM keeps unchanged, the signel *nIRQ* stays de-asserted, and AIC waits for new interrupt requests occuring.

6.4.5.7 Disabling and Enabling of Interrupt Requests

AIC ingnores those interrupt requests from *disabled* interrupt sources. That is, these interrupt requests cannot participate in AIC’s arbitration.

If an interrupt source is enabled and generates an interrupt request to AIC, it is called ***active***.

If an interrupt source is disabled, or is enabled but doesn't gerenate any interrupt request to AIC, it is called ***inactive***.

All interrupt sources are disabled at power-on. AIC provides three registers for users to easily configure the disables/enabled status of every interrupt source:

1. Interrupt Enable Register (AIC_IE)
2. Interrupt Enabling-Only Register (AIC_IEN)
3. Interrupt Disabling-Only Register (AIC_IDIS)

Each register is of 64 bits, and divided into two 32-bit APB-accessable registers. Each bit corresponds to an interrupt source. Their programming functions are listed in Table 6.4-1.

Register	Readable	Write (Each Individual Bit)
AIC_IE	Yes	0 = Disable an interrupt source. 1 = Enable an interrupt source.
AIC_IEN	No	0 = No effect. 1 = Enable an interrupt source.
AIC_IDIS	No	0 = No effect. 1 = Disable an interrupt source.

Table 6.4-1 Disabling/Enabling Register Programming Functions

6.4.6 Programming Guide

An example procedure of FIQ interrupt handling is listed below for user's reference.

1. Configure the priority level 0 and trigger type for every interrupt source.
2. Configure the disable/enable status for every interrupt source.
3. Wait for a FIQ interrupt activated.
4. When a FIQ interrupt is activated, the CPU enters the FIQ interrupt service routine.
5. Read the register AIC_FIQNUM to find out which interrupt source activating the FIQ.
6. (Interrupt Handling...)
7. Relase (Delete) the interrupt request from the interrupt source.
8. Write any value to the register AIC_FIQRST to reset AIC.
9. Go to Step 3.

For Step 3, please refer to section 6.4.5.3 for how AIC arbitrates two or more interrupt requests and activates a FIQ interrup to the CPU.

Steps 5-8 represent a basic interrupt service routine.

An similar example procedure of IRQ interrupt handling is listed below for user's reference.

1. Configure the priority levels from 1 to 7 and trigger type for every interrupt source.
2. Configure the disable/enable status for every interrupt source.
3. Wait for a IRQ interrupt activated.

4. When a IRQ interrupt is activated, the CPU enters the IRQ interrupt service routine.
5. Read the register AIC_IRQNUM to find out which interrupt source activating the IRQ.
6. (Interrupt Handling...)
7. Relase (Delete) the interrupt request from the interrupt source.
8. Write any value to the register AIC_IRQRST to reset AIC.
9. Go to Step 3.

For Step 3, please refer to section 6.4.5.4 for how AIC arbitrates two or more interrupt requests and activates an IRQ interrup to the CPU.

In Step 4, if a FIQ interrupt is being serviced, the IRQ interrupt should wait.

Steps 5-8 represent a basic interrupt service routine.

6.4.7 Interrupt Source List

The AIC can support up to 64 interrupt sources, numbered from 0 to 63. There are now 62 interrupt sources supported in the system, as listed in Table 6.4-2.

Source Number	Interrupt Name	Interrupt Source
0	(Reserved)	(Reserved)
1	WDT_INT	Watchdog Timer Interrupt
2	WWDT_INT	Windowed WDT Interrupt
3	LVD_INT	Low Voltage Detect Interrupt
4	EXT_INT0	External Interrupt 0
5	EXT_INT1	External Interrupt 1
6	EXT_INT2	External Interrupt 2
7	EXT_INT3	External Interrupt 3
8	GPA_INT	GPIO A Interrupt
9	GPB_INT	GPIO B Interrupt
10	GPC_INT	GPIO C Interrupt
11	GPD_INT	GPIO D Interrupt
12	I2S_INT	I^{2}S Interrupt
13	(Reserved)	(Reserved)
14	VCAP0_INT	VCAP 0 Interrupt
15	RTC_INT	RTC Interrupt
16	TIMER0_INT	Timer 0 Interrupt
17	TIMER1_INT	Timer 1 Interrupt
18	ADC_INT	ADC Interrupt

19	EMC0_RX_INT	EMC 0 RX Interrupt
20	EMC1_RX_INT	EMC 1 RX Interrupt
21	EMC0_TX_INT	EMC 0 TX Interrupt
22	EMC1_TX_INT	EMC 1 TX Interrupt
23	EHCI_INT	USB 2.0 Host Controller Interrupt
24	OHCI_INT	USB 1.1 Host Controller Interrupt
25	PDMA0_INT	PDMA Channel 0 Interrupt
26	PDMA1_INT	PDMA Channel 1 Interrupt
27	SDH_INT	SD/SDIO Host Interrupt
28	FMI_INT	FMI Interrupt
29	UDC_INT	USB Device Controller Interrupt
30	TIMER2_INT	Timer 2 Interrupt
31	TIMER3_INT	Timer 3 Interrupt
32	TIMER4_INT	Timer 4 Interrupt
33	VCAP1_INT	VCAP 1 Interrupt
34	TIMER5_INT	Timer 5 Interrupt
35	CRYPTO_INT	CRYPTO Engine Interrupt
36	UART0_INT	UART 0 Interrupt
37	UART1_INT	UART 1 Interrupt
38	UART2_INT	UART 2 Interrupt
39	UART4_INT	UART 4 Interrupt
40	UART6_INT	UART 6 Interrupt
41	UART8_INT	UART 8 Interrupt
42	CAN3_INT	CAN 3 Interrupt
43	UART3_INT	UART 3 Interrupt
44	UART5_INT	UART 5 Interrupt
45	UART7_INT	UART 7 Interrupt
46	UART9_INT	UART 9 Interrupt
47	I2C2_INT	I^{2}C 2 Interrupt
48	I2C3_INT	I^{2}C 3 Interrupt
49	GPE_INT	GPIO E Interrupt
50	SPI1_INT	SPI 1 Interrupt
51	QSPI0_INT	QSPI 0 Interrupt
52	SPI0_INT	SPI 0 Interrupt

53	I2C0_INT	I^{2}C 0 Interrupt
54	I2C1_INT	I^{2}C 1 Interrupt
55	SMC0_INT	SmartCard 0 Interrupt
56	SMC1_INT	SmartCard 1 Interrupt
57	GPF_INT	GPIO F Interrupt
58	CAN0_INT	CAN 0 Interrupt
59	CAN1_INT	CAN 1 Interrupt
60	PWM0_INT	PWM 0 Interrupt
61	PWM1_INT	PWM 1 interrupt
62	CAN2_INT	CAN 2 Interrupt
63	GPG_INT	GPIO G Interrupt

Table 6.4-2 Interrupt Source List

6.4.8 Register Map

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

Register	Offset	R/W	Description	Reset Value
AIC Base Address: **AIC_BA = 0xB004_2000**				
AIC_SRC00	AIC_BA+0x000	R/W	AIC Source Configuration Register 0	0x4747_4747
AIC_SRC01	AIC_BA+0x004	R/W	AIC Source Configuration Register 1	0x4747_4747
AIC_SRC02	AIC_BA+0x008	R/W	AIC Source Configuration Register 2	0x4747_4747
AIC_SRC03	AIC_BA+0x00C	R/W	AIC Source Configuration Register 3	0x4747_4747
AIC_SRC04	AIC_BA+0x010	R/W	AIC Source Configuration Register 4	0x4747_4747
AIC_SRC05	AIC_BA+0x014	R/W	AIC Source Configuration Register 5	0x4747_4747
AIC_SRC06	AIC_BA+0x018	R/W	AIC Source Configuration Register 6	0x4747_4747
AIC_SRC07	AIC_BA+0x01C	R/W	AIC Source Configuration Register 7	0x4747_4747
AIC_SRC08	AIC_BA+0x020	R/W	AIC Source Configuration Register 8	0x4747_4747
AIC_SRC09	AIC_BA+0x024	R/W	AIC Source Configuration Register 9	0x4747_4747
AIC_SRC10	AIC_BA+0x028	R/W	AIC Source Configuration Register 10	0x4747_4747
AIC_SRC11	AIC_BA+0x02C	R/W	AIC Source Configuration Register 11	0x4747_4747
AIC_SRC12	AIC_BA+0x030	R/W	AIC Source Configuration Register 12	0x4747_4747
AIC_SRC13	AIC_BA+0x034	R/W	AIC Source Configuration Register 13	0x4747_4747
AIC_SRC14	AIC_BA+0x038	R/W	AIC Source Configuration Register 14	0x4747_4747
AIC_SRC15	AIC_BA+0x03C	R/W	AIC Source Configuration Register 15	0x4747_4747
AIC_RAW0	AIC_BA+0x100	R	AIC Source Raw Value Register 0	Undefined
AIC_RAW1	AIC_BA+0x104	R	AIC Source Raw Value Register 1	Undefined
AIC_IS0	AIC_BA+0x110	R	AIC Interrupt Status Register 0	0x0000_0000
AIC_IS1	AIC_BA+0x114	R	AIC Interrupt Status Register 1	0x0000_0000
AIC_IRQ	AIC_BA+0x120	R	AIC IRQ Source Number Register	0x0000_0000
AIC_FIQ	AIC_BA+0x124	R	AIC FIQ Source Number Register	0x0000_0000
AIC_IE0	AIC_BA+0x128	R/W	AIC Interrupt Enable Register 0	0x0000_0000
AIC_IE1	AIC_BA+0x12C	R/W	AIC Interrupt Enable Register 1	0x0000_0000
AIC_IEN0	AIC_BA+0x130	W	AIC Interrupt Enabling-only Register 0	Undefined
AIC_IEN1	AIC_BA+0x134	W	AIC Interrupt Enabling-only Register 1	Undefined
AIC_IDIS0	AIC_BA+0x138	W	AIC Interrupt Disabling-only Register 0	Undefined
AIC_IDIS1	AIC_BA+0x13C	W	AIC Interrupt Disabling-only Register 1	Undefined
AIC_IRQRST	AIC_BA+0x150	W	AIC IRQ Reset Register	Undefined

AIC_FIQRST	AIC_BA+0x154	W	AIC FIQ Reset Register	Undefined

6.4.9 Register Description

AIC Source Configuration Register (AIC_SRC00 ~ AIC_SRC15)

Register	Offset	R/W	Description	Reset Value
AIC_SRC00	AIC_BA+0x000	R/W	AIC Source Configuration Register 0	0x4747_4747
AIC_SRC01	AIC_BA+0x004	R/W	AIC Source Configuration Register 1	0x4747_4747
AIC_SRC02	AIC_BA+0x008	R/W	AIC Source Configuration Register 2	0x4747_4747
AIC_SRC03	AIC_BA+0x00C	R/W	AIC Source Configuration Register 3	0x4747_4747
AIC_SRC04	AIC_BA+0x010	R/W	AIC Source Configuration Register 4	0x4747_4747
AIC_SRC05	AIC_BA+0x014	R/W	AIC Source Configuration Register 5	0x4747_4747
AIC_SRC06	AIC_BA+0x018	R/W	AIC Source Configuration Register 6	0x4747_4747
AIC_SRC07	AIC_BA+0x01C	R/W	AIC Source Configuration Register 7	0x4747_4747
AIC_SRC08	AIC_BA+0x020	R/W	AIC Source Configuration Register 8	0x4747_4747
AIC_SRC09	AIC_BA+0x024	R/W	AIC Source Configuration Register 9	0x4747_4747
AIC_SRC10	AIC_BA+0x028	R/W	AIC Source Configuration Register 10	0x4747_4747
AIC_SRC11	AIC_BA+0x02C	R/W	AIC Source Configuration Register 11	0x4747_4747
AIC_SRC12	AIC_BA+0x030	R/W	AIC Source Configuration Register 12	0x4747_4747
AIC_SRC13	AIC_BA+0x034	R/W	AIC Source Configuration Register 13	0x4747_4747
AIC_SRC14	AIC_BA+0x038	R/W	AIC Source Configuration Register 14	0x4747_4747
AIC_SRC15	AIC_BA+0x03C	R/W	AIC Source Configuration Register 15	0x4747_4747

31	30	29	28	27	26	25	24
TT3		**Reserved**			**PL3**		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TT2		**Reserved**			**PL2**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TT1		**Reserved**			**PL1**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TT0		**Reserved**			**PL0**		

Bits	Description	
[31:30] [23:22]	**TT3** **TT2**	**Trigger Type** 00 = low-level trigger.

[15:14] [7:6]	**TT1** **TT0**	01 = high-level trigger. (Default) 10 = falling-edge trigger. 11 = rising-edge trigger. For the register AIC_SRC*m*, TT*n* is applied to the interrupt source (*m* x 4 + *n*).
[5:3]	**Reserved**	Reserved.
[26:24] [18:16] [10:8] [2:0]	**PL3** **PL2** **PL1** **PL0**	**Priority Level** 000 = Priority Level 0 (the highest priority). 001 = Priority Level 1. 010 = Priority Level 2. 011 = Priority Level 3. 100 = Priority Level 4. 101 = Priority Level 5. 110 = Priority Level 6. 111 = Priority Level 7 (the lowest priority, Default). For the register AIC_SRC*m*, PL*n* is applied to the interrupt source (*m* x 4 + *n*). An interrupt request with priority level 0 will activate a FIQ interrupt. An interrupt request with other priority levels will activate an IRQ interrupt.

AIC Source Raw Value Register 0 (AIC_RAW0)

Register	Offset	R/W	Description	Reset Value
AIC_RAW0	AIC_BA+0x100	R	AIC Source Raw Value Register 0	Undefined

31	30	29	28	27	26	25	24
RV31	RV30	RV29	RV28	RV27	RV26	RV25	RV24
23	22	21	20	19	18	17	16
RV23	RV22	RV21	RV20	RV19	RV18	RV17	RV16
15	14	13	12	11	10	9	8
RV15	RV14	RV13	RV12	RV11	RV10	RV09	RV08
7	6	5	4	3	2	1	0
RV07	RV06	RV05	RV04	RV03	RV02	RV01	RV00

Bits	Description	
[31:0]	**RV*x*,** ***x* = 00-31**	**Interrupt Source Raw Value** Each bit indicates the raw logic value of interrupt source *x*. 0 = Logic value of interrupt source *x* is 0. 1 = Logic value of interrupt source *x* is 1.

AIC Source Raw Value Register 1 (AIC_RAW1)

Register	Offset	R/W	Description	Reset Value
AIC_RAW1	AIC_BA+0x104	R	AIC Source Raw Value Register 1	Undefined

31	30	29	28	27	26	25	24
RV63	RV62	RV61	RV60	RV59	RV58	RV57	RV56
23	22	21	20	19	18	17	16
RV55	RV54	RV53	RV52	RV51	RV50	RV49	RV48
15	14	13	12	11	10	9	8
RV47	RV46	RV45	RV44	RV43	RV42	RV41	RV40
7	6	5	4	3	2	1	0
RV39	RV38	RV37	RV36	RV35	RV34	RV33	RV32

Bits	Description	
[31:0]	**RV*x*,** ***x* = 32-63**	**Interrupt Source Raw Value** Each bit indicates the raw logic value of interrupt source *x*. 0 = Logic value of interrupt source *x* is 0. 1 = Logic value of interrupt source *x* is 1.

AIC Interrupt Status Register 0 (AIC_IS0)

Register	Offset	R/W	Description	Reset Value
AIC_IS0	AIC_BA+0x110	R	AIC Interrupt Status Register 0	0x0000_0000

31	30	29	28	27	26	25	24
IS31	IS30	IS29	IS28	IS27	IS26	IS25	IS24
23	22	21	20	19	18	17	16
IS23	IS22	IS21	IS20	IS19	IS18	IS17	IS16
15	14	13	12	11	10	9	8
IS15	IS14	IS13	IS12	IS11	IS10	IS09	IS08
7	6	5	4	3	2	1	0
IS07	IS06	IS05	IS04	IS03	IS02	IS01	IS00

Bits	Description	
[31:0]	**IS*x*,** ***x* = 00-31**	**Interrupt Status** Each bit indicates the inactive/active status of an interrupt source. 0 = Interrupt source *x* is inactive. 1 = Interrupt source *x* is active. An interrupt source is active if it is enabled and generates an interrupt request to AIC. An interrupt source is inactive if it is disabled, or it is enabled but does not generate an interrupt request to AIC.

AIC Interrupt Status Register 1 (AIC_IS1)

Register	Offset	R/W	Description	Reset Value
AIC_IS1	AIC_BA+0x114	R	AIC Interrupt Status Register 1	0x0000_0000

31	30	29	28	27	26	25	24
IS63	IS62	IS61	IS60	IS59	IS58	IS57	IS56
23	22	21	20	19	18	17	16
IS55	IS54	IS53	IS52	IS51	IS50	IS49	IS48
15	14	13	12	11	10	9	8
IS47	IS46	IS45	IS44	IS43	IS42	IS41	IS40
7	6	5	4	3	2	1	0
IS39	IS38	IS37	IS36	IS35	IS34	IS33	IS32

Bits	Description	
[31:0]	**IS*x*,** ***x* = 32-63**	**Interrupt Status** Each bit indicates the inactive/active status of an interrupt source *x*. 0 = Interrupt source *x* is inactive. 1 = Interrupt source *x* is active. An interrupt source is active if it is enabled and generates an interrupt request to AIC. An interrupt source is inactive if it is disabled, or it is enabled but does not generate an interrupt request to AIC.

AIC IRQ Source Number Register (AIC_IRQ)

Register	Offset	R/W	Description	Reset Value
AIC_IRQ	AIC_BA+0x120	R	AIC IRQ Source 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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**NUM**					

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**NUM**	**Interrup Source Number** The number indicates an interrupt source that activates an IRQ interrupt currently.

AIC FIQ Source Number Register (AIC_FIQ)

Register	Offset	R/W	Description	Reset Value
AIC_FIQ	AIC_BA+0x124	R	AIC FIQ Source 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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**NUM**					

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**NUM**	**Interrupt Source Number** The number indicates an interrupt source that activates an FIQ interrupt currently.

AIC Interrupt Enable Register 0 (AIC_IE0)

Register	Offset	R/W	Description	Reset Value
AIC_IE0	AIC_BA+0x128	R/W	AIC Interrupt Enable Register 0	0x0000_0000

31	30	29	28	27	26	25	24
IE31	**IE30**	**IE29**	**IE28**	**IE27**	**IES26**	**IE25**	**IE24**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
IE23	**IE22**	**IE21**	**IE20**	**IE19**	**IES18**	**IE17**	**IE16**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
IE15	**IE14**	**IE13**	**IE12**	**IE11**	**IE10**	**IE09**	**IE08**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IE07	**IE06**	**IE05**	**IE04**	**IE03**	**IE02**	**IE01**	**IE00**

Bits	Description	
[31:0]	**IE*x*,** ***x* = 00-31**	**Interrupt Enable Status** Each bit indicates the disabled/enabled status of an interrupt source. 0 = Interrupt source x Disabled. 1 = Interrupt source x Enabled. **Note:** The interrupt request from a disabled interrupt source is always ignored by AIC.

AIC Interrupt Enable Register 1 (AIC_IE1)

Register	Offset	R/W	Description	Reset Value
AIC_IE1	AIC_BA+0x12C	R/W	AIC Interrupt Enable Register 1	0x0000_0000

31	30	29	28	27	26	25	24
IE63	IE62	IE61	IE60	IE59	IE58	IE57	IE56
23	22	21	20	19	18	17	16
IE55	IE54	IE53	IE52	IE51	IE50	IE49	IE48
15	14	13	12	11	10	9	8
IE47	IE46	IE45	IE44	IE43	IE42	IE41	IE40
7	6	5	4	3	2	1	0
IE39	IE38	IE37	IE36	IE35	IE34	IE33	IE32

Bits	Description	
[31:0]	**IE*x*,** ***x* = 32-63**	**Interrupt Enable Status** Each bit indicates the disabled/enabled status of an interrupt source. 0 = Interrupt source *x* Disabled. 1 = Interrupt source *x* Enabled. **Note:** The interrupt request from a disabled interrupt source is always ignored by AIC.

AIC Interrupt Enabling-only Register 0 (AIC_IEN0)

Register	Offset	R/W	Description	Reset Value
AIC_IEN0	AIC_BA+0x130	W	AIC Interrupt Enabling-only Register 0	Undefined

31	30	29	28	27	26	25	24
IEN31	IEN30	IEN29	IEN28	IEN27	IEN26	IEN25	IEN24
23	22	21	20	19	18	17	16
IEN23	IEN22	IEN21	IEN20	IEN19	IEN18	IEN17	IEN16
15	14	13	12	11	10	9	8
IEN15	IEN14	IEN13	IEN12	IEN11	IEN10	IEN09	IEN08
7	6	5	4	3	2	1	0
IEN07	IEN06	IEN05	IEN04	IEN03	IEN02	IEN01	IEN00

Bits	Description	
[31:0]	**IEN*x*,** ***x* = 00-31**	**Interrupt Enabling** Writing 1 to a bit enables the corresponding interrupt source. 0 = No effect. 1 = Enable interrupt source *x*. **Note:** Writing 0 to a given bit of AIC_IEN0 cannot change the current disabled/enabled status of the corresponding interrupt source. Users should write 0 to the the same bit of AIC_IE0, or write 1 to the same bit of AIC_IDIS0, to disable the interrupt source.

AIC Interrupt Enabling-only Register 1 (AIC_IEN1)

Register	Offset	R/W	Description	Reset Value
AIC_IEN1	AIC_BA+0x134	W	AIC Interrupt Enabling-only Register 1	Undefined

31	30	29	28	27	26	25	24
IEN63	**IEN62**	**IEN61**	**IEN60**	**IEN59**	**IEN58**	**IEN57**	**IEN56**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
IEN55	**IEN54**	**IEN53**	**IEN52**	**IEN51**	**IEN50**	**IEN49**	**IEN48**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
IEN47	**IEN46**	**IEN45**	**IEN44**	**IEN43**	**IEN42**	**IEN41**	**IEN40**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IEN39	**IEN38**	**IEN37**	**IEN36**	**IEN35**	**IEN34**	**IEN33**	**IEN32**

Bits	Description	
[31:0]	**IEN*x*,** ***x* = 32-63**	**Interrupt Enabling** Writing 1 to a bit enables the corresponding interrupt source 0 = No effect. 1 = Enable interrupt source *x*. **Note:** Writing 0 to a given bit of AIC_IEN1 cannot change the current disabled/enabled status of the corresponding interrupt source. Users should write 0 to the the same bit of AIC_IE1, or write 1 to the same bit of AIC_IDIS1, to disable the interrupt source.

AIC Interrupt Disabling-only Register 0 (AIC_IDIS0)

Register	Offset	R/W	Description	Reset Value
AIC_IDIS0	AIC_BA+0x138	W	AIC Interrupt Disabling-only Register 0	Undefined

31	30	29	28	27	26	25	24
IDIS31	IDIS30	IDIS29	IDIS28	IDIS27	IDIS26	IDIS25	IDIS24
23	22	21	20	19	18	17	16
IDIS23	IDIS22	IDIS21	IDIS20	IDIS19	IDIS18	IDIS17	IDIS16
15	14	13	12	11	10	9	8
IDIS15	IDIS14	IDIS13	IDIS12	IDIS11	IDIS10	IDIS09	IDIS08
7	6	5	4	3	2	1	0
IDIS07	IDIS06	IDIS05	IDIS04	IDIS03	IDIS02	IDIS01	IDIS00

Bits	Description	
[31:0]	**IDIS*x*,** ***x* = 00-31**	**Interrupt Disabling** Writing 1 to a bit disables the corresponding interrupt source. 0 = No effect. 1 = Disable interrupt source *x*.

AIC Interrupt Disabling-only Register 1 (AIC_IDIS1)

Register	Offset	R/W	Description	Reset Value
AIC_IDIS1	AIC_BA+0x13C	W	AIC Interrupt Disabling-only Register 1	Undefined

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**
IDIS63	**IDIS62**	**IDIS61**	**IDIS60**	**IDIS59**	**IDIS58**	**IDIS57**	**IDIS56**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
IDIS55	**IDIS54**	**IDIS53**	**IDIS52**	**IDIS51**	**IDIS50**	**IDIS49**	**IDIS48**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
IDIS47	**IDIS46**	**IDIS45**	**IDIS44**	**IDIS43**	**IDIS42**	**IDIS41**	**IDIS40**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IDIS39	**IDIS38**	**IDIS37**	**IDIS36**	**IDIS35**	**IDIS34**	**IDIS33**	**IDIS32**

Bits	Description	
[31:0]	**IDIS*x*,** ***x* = 32-63**	**Interrupt Disabling** Writing 1 to a bit disables the corresponding interrupt source. 0 = No effect. 1 = Disable interrupt source *x*.

NUC980 SERIES TECHNICAL REFERENCE MANUAL

AIC IRQ Reset Register (AIC_IRQRST)

Register	Offset	R/W	Description	Reset Value
AIC_IRQRST	AIC_BA+0x150	W	AIC IRQ Reset Register	Undefined

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

Bits	Description	
[31:0]	**Null**	Writing any value to this register will reset and restart AIC's IRQ processing.

AIC FIQ Reset Register (AIC_FIQRST)

Register	Offset	R/W	Description	Reset Value
AIC_FIQRST	AIC_BA+0x154	W	AIC FIQ Reset Register	Undefined

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

Bits	Description	
[31:0]	**Null**	Writing any value to this register will reset and restart AIC’s FIQ processing.

6.5 SDRAM Interface Controller (SDIC)

6.5.1 Overview

The SDRAM Controller supports SDR, DDR, Low-Power DDR and DDR2 type SDRAM. The memory device size type can be from 16M bit and up to 1G bits. Only 16-bit data bus width is supported. The total system memory size can be from 2M bytes and up to 256M bytes for different SDRAM configuration.

The SDRAM controller interface to three isolated AHB. All these AHB masters can access the memory independent. Except the memory access, the masters of AHB also could access the SDRAM control registers.

For performance and function issue, the SDRAM controller also supports the proprietary Enhanced-AHB (EAHB). The EAHB adds the down-count address mode, byte-enable signal and explicit burst access number. The explicit access number function is reached by modifing the HBURST signal to EHBURST and it represents the access number. The maximum EAHB access number is 16. The SDRAM controller also builds a BIST module to test the external memory device.

An internal arbiter is used to schedule the access from the masters and the BIST request, the BIST request with the highest priority and then the AHB3 master, AHB2 master and AHB1 master.

The SDRAM controller uses 3 pipe queues to improve the SDRAM command and data bus efficiency. The request in queue0 is the SDRAM active data access request. Simultaneously, the requests in queue1 can request the controller to issue the ACTIVE or PRECHARGE command to reduce the access latency for the later command. The queue1 can also issue the READ or WRITE command to close the SDRAM command when advance pipe queue.

The SDRAM refresh rate is programmable. The Refresh and Power-on control module generates the refresh request signal and SDRAM power on sequence. The SDRAM controller also supports software reset, SDRAM self-refresh and auto power down function.

6.5.2 Features

- Supports DDR, DDR2 and LPDDR SDRAM
- Clock speed up to 150 MHz
- Supports 16-bit data bus width
- Supports two chip selects
- Supports total memory size up to 256M bytes (each chip select for 128M bytes)

6.5.3 Block Diagram

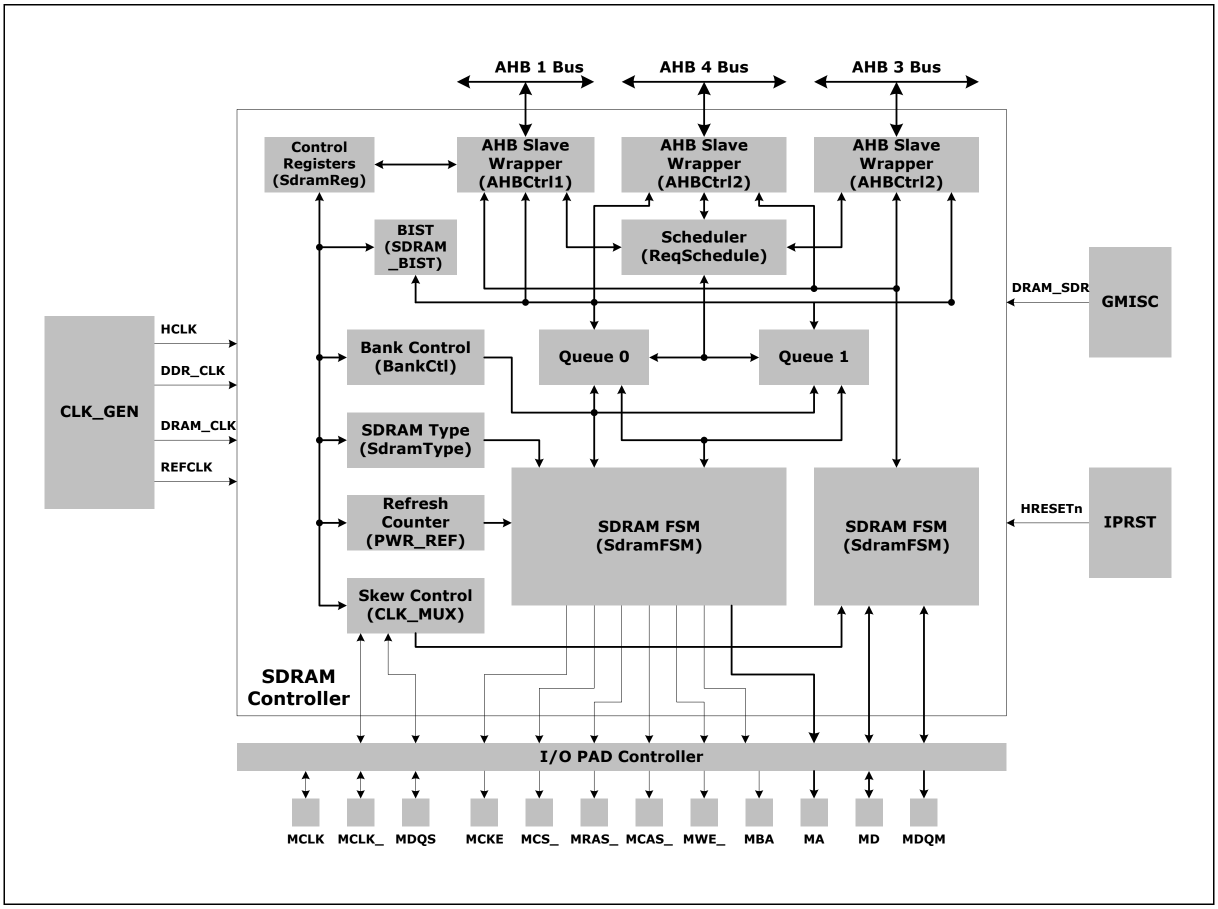


Figure 6.5-1 SDRAM Controller Block Diagram

6.5.4 Basic Configuration

- Clock source Configuration
  - Enable SDRAM controller clock in SDIC (CLK_HCLKEN[10]).
- Reset Configuration
  - Reset SDRAM controller in SDICRST (SYS_AHBIPRST[6]).
- SDRAM Initialization
  - Refer to SDRAM Power-Up Sequence section to initialize SDRAM.

6.5.5 Functional Description

6.5.5.1 SDRAM Control Timing

The SDIC supports programmable CAS Latency and Refresh Rate control. It also can control the SDRAM to enter self-refresh mode to reduce the power consumption in power-down mode.

The SDIC provides the fixed sequential burst type and burst length is 4. In addition, SDIC implements some programmable controls for the SDRAM operations:

- Configurable SDRAM Type to support DDR, DDR2 and LPDDR SDRAM.

- Configurable SDRAM Size to support 16Mbits, 64Mbits, 128Mbits, 256Mbits, 512Mbits and 1Gbits SDRAM with 16 bits data width.
- Configurable SDRAM Timing to adjust tWR, tRP, tRCD, tRAS, tRFC, tXSR, tRC, tRRD and tWTR timings.
- Configurable SDRAM Read Latency from 2 to 4 clocks.
- Configurable SDRAM Refresh timing for Auto Refresh or power save mode Self Refresh.

6.5.5.2 SDRAM Power-Up Sequence

Before the SDRAM can be accessed after power on, or when exiting Deep -Power-down mode, an SDRAM device must be initialized by software to progress an initialization sequence.

Because the DDR, DDR2 and LPDDR SDRAM require different initialization sequences and different parameters, the sequence is driven by software manually by using the registers SDIC_CMD, SDIC_MR, SDIC_EMR, SDIC_EMR2 and SDIC_EMR3.

DDR initialization sequence

1. Wait for 200us after power up.
2. Set the SDRAM type to DDR. This is accomplished by writing 10 to SD_TYPE (SDIC_OPMCTL[6:5]).
3. Set the SDRAM controller in initialization state. This is accomplished by writing 1 to InitState (SDIC_CMD[0]).
4. Set the CKE_H (SDIC_CMD[1]) to be 1 to force the CKE at high state.
5. Apply a PRECHARGE ALL command. This is accomplished by writing 1 to PALL_CMD (SDIC_CMD[2]). The PALL_CMD bit will auto clear after the PRECHARGE command completed.
6. Apply a MRS (Mode Register Set) command to EMR (Extended Mode Register) to enable DLL. This is accomplished by writing appropriate value to the register SDIC_EMR (SDRAM EXTEND MODE Register).
7. Apply a MRS (Mode Register Set) command to MR (Mode Register) with A8 high to set DDR SDRAM in normal operation with resetting the DLL. This is accomplished by writing appropriate value with bit [8] high to the register SDIC_MR (SDRAM MODE Register).
8. Apply a PRECHARGE ALL command. This is accomplished by writing 1 to PALL_CMD (SDIC_CMD[2]). The PALL_CMD bit will auto clear after the PRECHARGE command completed.
9. Apply two or more AUTOREFRESH commands. This is accomplished by writing 1 to REF_CMD (SDIC_CMD[3]) twice or more. The REF_CMD is auto cleared after SDRAM controller completes each CAS-BEFORE-RAS refresh command.
10. Apply a MRS (Mode Register Set) command to MR (Mode Register) with A8 low to set DDR SDRAM in normal operation without resetting the DLL. This is accomplished by writing appropriate value with bit [8] low to the register SDIC_MR (SDRAM MODE Register).
11. Apply 200 dummy clocks to meet minimum latency delay between MRS and normal operation command (ACTIVE, READ, WRITE …). This is accomplished by inserting a period of delay.
12. SDRAM initialization sequence completed and SDRAM controller exit initialization state and enter normal operating mode. This is accomplished by writing 0 to both InitState (SDIC_CMD[0]) and CKE_H (SDIC_CMD[1]).

DDR2 initialization sequence

1. Wait for 200us after power up.
2. Apply NOP or DESELECT commands for a minimum 400 ns.
3. Set the SDRAM type is DDR. This is accomplished by writing 11 to SD_TYPE (SDIC_OPMCTL[6:5]).
4. Set the SDRAM controller in initialization state. This is accomplished by writing 1 to InitState (SDIC_CMD[0]).
5. Set the CKE_H (SDIC_CMD[1]) to be 1 to force the CKE at high state.
6. Apply a PRECHARGE ALL command. This is accomplished by writing 1 to PALL_CMD (SDIC_CMD[2]). The PALL_CMD bit will auto clear after the PRECHARGE command completed.
7. Apply a MRS (Mode Register Set) command to EMR2 (Extended Mode Register 2). This is accomplished by writing appropriate value to the register SDIC_EMR2 (SDRAM EXTEND MODE Register 2).
8. Apply a MRS (Mode Register Set) command to EMR3 (Extended Mode Register 3). This is accomplished by writing appropriate value to the register SDIC_EMR3 (SDRAM EXTEND MODE Register 3).
9. Apply a MRS (Mode Register Set) command to EMR (Extended Mode Register) to enable DLL. This is accomplished by writing appropriate value to the register SDIC_EMR (SDRAM EXTEND MODE Register).
10. Apply a MRS (Mode Register Set) command to MR (Mode Register) with A8 high to set DDR SDRAM in normal operation with resetting the DLL. This is accomplished by writing appropriate value with bit [8] high to the register SDIC_MR (SDRAM MODE Register).
11. Apply a PRECHARGE ALL command. This is accomplished by writing 1 to PALL_CMD (SDIC_CMD[2]). The PALL_CMD bit will auto clear after the PRECHARGE command completed.
12. Apply two or more AUTOREFRESH commands. This is accomplished by writing 1 to REF_CMD (SDIC_CMD[3]) twice or more. The REF_CMD is auto cleared after SDRAM controller completes each CAS-BEFORE-RAS refresh command.
13. Apply a MRS (Mode Register Set) command to MR (Mode Register) with A8 low to set DDR SDRAM in normal operation without resetting the DLL. This is accomplished by writing appropriate value with bit [8] low to the register SDIC_MR (SDRAM MODE Register).
14. Apply a MRS (Mode Register Set) command to EMR (Extended Mode Register) to enable OCD default state. This is accomplished by writing appropriate value with 3is accomplished by to the register SDIC_EMR (SDRAM EXTEND MODE Register).
15. Apply a MRS (Mode Register Set) command to EMR (Extended Mode Register) to enable OCD exit state. This is accomplished by writing appropriate value with 3is accomplished by to the register SDIC_EMR (SDRAM EXTEND MODE Register).
16. Apply 200 dummy clocks to meet minimum latency delay between MRS and normal operation command (ACTIVE, READ, WRITE…). This is accomplished by inserting a period of delay..
17. SDRAM initialization sequence completed and SDRAM controller exit initialization state and enter normal operating mode. This is accomplished by writing 0 to both InitState (SDIC_CMD[0]) and CKE_H (SDIC_CMD[1]).

6.5.5.3 System Memory Address and SDRAM Address Mapping

Table 6.5-1 and Table 6.5-2 indicate how the 32-bit system memory address be mapped to SDRAM address. All the SDRAM devices listed below are 16-bit data bus width.

Type	R X C	R/C	BA1	BA0	A13	A12	A11	A10	A09	A08	A07	A06	A05	A04	A03	A02	A01	A00
128M 8Mx16	12x9	R	11	10			23	12	13	22	21	20	19	18	17	16	15	14
		C						AP		9	8	7	6	5	4	3	2	1
256M 16Mx16	13x9	R	11	10		24	23	12	13	22	21	20	19	18	17	16	15	14
		C						AP		9	8	7	6	5	4	3	2	1
512M 32Mx16	13x10	R	12	11		25	23	24	13	22	21	20	19	18	17	16	15	14
		C						AP	10	9	8	7	6	5	4	3	2	1
1G 64Mx16	14x10	R	12	11	26	25	23	24	13	22	21	20	19	18	17	16	15	14
		C						AP	10	9	8	7	6	5	4	3	2	1

Table 6.5-1 DDR SDRAM System Memory Address and SDRAM Address Mapping Table

Type	R X C	R/C	BA2	BA1	BA0	A12	A11	A10	A09	A08	A07	A06	A05	A04	A03	A02	A01	A00
128M 8Mx16	12x9	R		11	10		23	12	13	22	21	20	19	18	17	16	15	14
		C						AP		9	8	7	6	5	4	3	2	1
256M 16Mx16	13x9	R		11	10	24	23	12	13	22	21	20	19	18	17	16	15	14
		C						AP		9	8	7	6	5	4	3	2	1
512M 32Mx16	13x10	R		12	11	25	23	24	13	22	21	20	19	18	17	16	15	14
		C						AP	10	9	8	7	6	5	4	3	2	1
1G 64Mx16	13x10	R	13	12	11	26	23	25	24	22	21	20	19	18	17	16	15	14
		C						AP	10	9	8	7	6	5	4	3	2	1

Table 6.5-2 DDR2 SDRAM System Memory Address and SDRAM Address Mapping Table

Note: The AHB bus address HADDR prefixes have been omitted in Table 6.5-1 and Table 6.5-2.

A13 ~ A00 are the Address pins of the SDRAM interface.

BA2, BA1 and BA0 are the Bank Selected Signal of SDRAM.

6.5.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
SDIC Base Address: **SDIC_BA = 0xB000_2000**				
SDIC_OPMCTL	SDIC_BA + 0x000	R/W	SDRAM Controller Operation Mode Control Register	0x0003_04x6
SDIC_CMD	SDIC_BA + 0x004	R/W	SDRAM Command Register	0x0000_0021
SDIC_REFCTL	SDIC_BA + 0x008	R/W	SDRAM Controller Refresh Control Register	0x0000_80FF
SDIC_SIZE0	SDIC_BA + 0x010	R/W	SDRAM 0 Size Register	0x0000_000X
SDIC_SIZE1	SDIC_BA + 0x014	R/W	SDRAM 1 Size Register	0x1000_0000
SDIC_MR	SDIC_BA + 0x018	R/W	SDRAM Mode Register	0x0000_0032
SDIC_EMR	SDIC_BA + 0x01C	R/W	SDRAM Extended Mode Register	0x0000_4000
SDIC_EMR2	SDIC_BA + 0x020	R/W	SDRAM Extended Mode Register 2	0x0000_8000
SDIC_EMR3	SDIC_BA + 0x024	R/W	SDRAM Extended Mode Register 3	0x0000_C000
SDIC_TIME	SDIC_BA + 0x028	R/W	SDRAM Timing Control Register	0x2BDE_9649
SDIC_DQSODS	SDIC_BA + 0x030	R/W	DQS Output Delay Selection Register	0x0000_1010
SDIC_CKDQSDS	SDIC_BA + 0x034	R/W	Clock and DQS Delay Selection Register	0x0044_4400
SDIC_DAENSEL	SDIC_BA + 0x038	R/W	Data Latch Enable Selection Register	0x0000_0000

6.5.7 Register Description

SDRAM Controller Operation Mode Control Register (SDIC_OPMCTL)

Register	Offset	R/W	Description	Reset Value
SDIC_OPMCTL	SDIC_BA + 0x000	R/W	SDRAM Controller Operation Mode Control Register	0x0003_04x6

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved			**RD2WR_CTL**	**OEDelay**	**LowFreq**	**PreActBnk**	**AutoPDn**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**RDBUFTH**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**SD_TYPE**		**PchMode**	**OPMode**	**MCLKMode**	**SDRAM_EN**	**Reserved**

Bits	Description	
[31:19]	**Reserved**	Reserved.
[20]	**RD2WR_CTL**	**Read-to-write Turn Around Control** This bit is to insert one more turn around cycle between memory read and memory write access to SDRA'M device. 0 = Default turn-around cycle is used. 1 = One more turn-around cycle is inserted between memory read and write access.
[19]	**OEDelay**	**Output Enable Delay Half MCLK** This bit controls the data output enable signal. If set high, the data output enable will be turned off half MCLK earlier. 0 = Default data output enable timing. 1 = Turn off data output enable half MCLK earlier.
[18]	**LowFreq**	**Low Frequency Mode** For low power DDR (LPDDR) SDRAM, the valid read data outputted by LPDDR SDRAM is not ready at clock edge. If this bit is enabled, the SDRAM controller will sampled read data based on the following timing: If CL is 2, the read data output latency will be 2*tCK+tAC. If CL is 3, the read data output latency will be tCK+tAC. CL = CAS Latency. tCK = Clock cycle time for LPDDR SDRAM. tAC = Data output latency from clock for LPDDR SDRAM. This bit only takes effect when the SD_TYPE is selected in DDR or DDR2 SDRAM. 0 = SDRAM controller sampled read data based on the DDR/DDR2 standard. (Default) 1 = SDRAM controller sampled read data based on the LPDDR standard.

[17]	**PreActBnk**	**Pre-active Bank** If this bit is enabled, the SDRAM controller will open request bank early to get better performance. It means maybe more than one bank active and consumes more power. There are several bus requests in this chip and the SDRAM controller checks all these requests simultaneous. If request in queue access bank is different with current bank, the SDRAM controller will open the new bank early to reduce the access latency to get better performance. The mode takes effect for Close-Page mode (OPMode is 0) only. In Open-Page mode, SDRAM controller always opens bank early. 0 = Pre-Active-Bank mode Disabled. 1 = Pre-Active-Bank mode Enabled. (Default)
[16]	**AutoPDn**	**Auto Power Down Mode** If this bit is enabled, the SDRAM controller will make SDRAM to enter power down mode (CKE low) automatically while the memory request is stop. Otherwise, the SDRAM is in IDLE state (CKE = high). 0 = Auto power down mode Disabled. 1 = Auto power down mode Enabled. (Default)
[15:11]	**Reserved**	Reserved.
[10:8]	**RDBUFTH**	**The AHB Read SDRAM Read Buffer Threshold Control** Due to the SDRAM working clock may be higher than the AHB BUS clock, the SDRAM controller contains a read buffer for each AHB interface and they are used for data pre-read. The controller read the data to buffer full or till AHB read request end. When the data buffer full, the controller stop the read request and to service another AHB request. The RDBUFTH is used to control the buffer threshold level for the SDRAM re-start the memory request. This function can minimize the redundant memory read. 000 = Reserved. Others = Re-start memory read when the data buffer remain data level is equal to RDBUFTH.
[7]	**Reserved**	Reserved.
[6:5]	**SD_TYPE**	**SDRAM Type** This file indicates which type of SDRAM is used. The reset value is decided by chip’s system power-on setting. 00 = SDRAM type is SDR (Single Data Rate SDRAM). 01 = SDRAM type is LPDDR. 10 = SDRAM type is DDR SDRAM (Double Data Rate SDRAM). 11 = SDRAM type is DDR2 SDRAM.
[4]	**PchMode**	**Auto Pre-charge Mode** This bit controls if SDRAM controller will pre-charge all active banks while there is no new memory request. The SDRAM power consumption increases with the active bank number. If no new memory access request, the active bank can be pre-charge to save power, but the SDRAM controller may lose some performance. 0 = The SDRAM controller keeps bank active. 1 = Pre-charge all bank if there is no new memory request. (Default) **Note:** This bit only take effect while OPMode is high.

[3]	**OPMode**	**Open Page Mode** This bit controls if the SDRAM controller will send pre-charge command to close the active bank page after SDRAM access. If this bit doesn’t be enabled, the SDRAM controller will pre-charge bank after each burst read or write cycle. This could make the SDRAM consume less power. If set this bit high, the state machine will keep on the bank-active state until a page missed read/write request or at a period refresh request. This makes SDRAM controller to get better performance, but SDRAM will consume more power. 0 = Pre-charge after each read/write request. (Default) 1 = No auto pre-charge. The bank page keeps in active state after read/write.
[2]	**MCLKMode**	**MCLK Mode** This bit controls the SDRAM clock (MCLK) is always enabled, or is enabled and disabled by SDRAM controller automatically. 0 = The MCLK is enabled and disabled by SDRAM controller automatically. The MCLK will keep low when the SDRAM is in the power down state. 1 = MCLK is always enabled. (Default)
[1]	**SDRAM_EN**	**SDRAM Controller Enable Bit** Set this bit 0 will disable the SDRAM controller function. 0 = SDRAM controller Disabled. 1 = SDRAM controller Enabled. (Default)
[0]	**Reserved**	Reserved.

SDRAM Command Register (SDIC_CMD)

Register	Offset	R/W	Description	Reset Value
SDIC_CMD	SDIC_BA + 0x004	R/W	SDRAM Command Register	0x0000_0021

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

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5]	**AutoExSelfRef**	**Auto Exit Self-refresh** This controls if the SDRAM will exit self refresh mode automatically while the system interrupt occurred. 0 = Auto exit self-refresh function Disabled. The SDRAM keep in self-refresh state when the interrupt occur. 1 = Auto exit self-refresh function Enabled. The SDRAM will exit self-refresh state when the interrupt occurred. (Default)
[4]	**SELF_REF**	**Self-refresh Command** Set this bit high, the SDRAM controller will make SDRAM to enter self-refresh mode. SDRAM controller will not have response to any read, write or refresh request until this bit is cleared. If the bit 5 (AutoExSelfRef) is set high, this bit will be cleared automatically when the system interrupt occurred. 0 = SDRAM in normal operation mode. (Default) 1 = Set the SDRAM enter the Self Refresh power saving state.
[3]	**REF_CMD**	**Auto Refresh Command** Set this bit high the SDRAM controller will issue an auto refresh command to SDRAM. This bit will be cleared by SDRAM controller automatically after the auto refresh command is end. 0 = No operation. (Default) 1 = Issue an auto refresh command to the SDRAM.
[2]	**PALL_CMD**	**Pre-charge All Bank Command** Set this bit high, the SDRAM controller will issue a pre-charge all bank command to the SDRAM. This bit will be cleared by SDRAM controller automatically after the pre-chare all bank command is end 0 = No operation. (Default) 1 = Issue a pre-charge all bank command to the SDRAM.

[1]	**CKE_H**	**CKE High** This bit indicates the CKE is controlled by SDRAM controller state machine or always keeps high. 0 = Set the CKE signal in normal state and controlled by the SDRAM controller state machine. (Default) 1 = Set the CKE signal keep in llerateed by.
[0]	**InitState**	**Initial State** This bit indicates if the SDRAM is in the Initialize State. When the SDRAM is in the initialize state, SDRAM controller will not accept any SDRAM read or write request. The logical state of the internal circuit of the SDRAM is undefined after power on. The SDRAM must be initialized to set the SDRAM into the right operation. The SDR SDRAM, LP SDRAM, DDR SDRAM and DDR2 SDRAM have different initialization sequence, and the users must set the right sequence to initialize the SDRAM. This bit is default high and means the SDRAM is not initialized yet. After the initialization, user must set this bit low to set the SDRAM controller in correct mode. 0 = The SDRAM is in normal state. 1 = The SDRAM is in initialization state, the SDRAM initialization doesn't complete yet. (Default)

SDRAM Controller Refresh Control Register (SDIC_REFCTL)

Register	Offset	R/W	Description	Reset Value
SDIC_REFCTL	SDIC_BA + 0x008	R/W	SDRAM Controller Refresh Control Register	0x0000_80FF

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[15]	**REF_EN**	**Refresh Period Counter Enable Bit** This bit controls if the refresh period counter is enabled. If refresh period counter is disabled, the SDRAM controller would never issue auto-refresh command to SDRAM automatically. However, if refresh period counter is enabled, the SDRAM controller will issue auto-refresh command to SDRAM automatically once the refresh period counter is equal to REFRATE. 0 = Refresh period counter Disabled. 1 = Refresh period counter Enabled. SDRAM controller issues auto-refresh command to SDRAM periodically. (Default)
[14:0]	**REFRATE**	**Refresh Count Value** This field defines the period for SDRAM controller to generate the auto-refresh command to SDRAM. The SDRAM controller will issue an auto-refresh cycle to SDRAM automatically for every period programmed in the REFRAT field when the REF_EN bit of is set. The refresh period is calculated as Period = REFRAT / fSCLK. The fSCLK is the frequency of external crystal for chip.

SDRAM Size Register (SDIC_SIZE)

Register	Offset	R/W	Description	Reset Value
SDIC_SIZE0	SDIC_BA + 0x010	R/W	SDRAM 0 Size Register	0x0000_000X
SDIC_SIZE1	SDIC_BA + 0x014	R/W	SDRAM 1 Size Register	0x1000_0000

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

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:21]	**BASADDR**	**Base Address** This field defines the memory space where the SDRAM is mapped. In this chip, the SDRAM could be mapped to address 0x0000_0000 ~ 0x1fff_ffff of system memory, and shadow address on 0x8000_0000 ~ 0x9fff_ffff of system memory. The minimum supported SDRAM size is 2M bytes. Based on the above criteria, the bit [28:21] is used to define the base address. For example, if [28:21] is set as 0x01, the address 0 of SDRAM memory will be mapped to 0x00200000 of system memory.
[20:4]	**Reserved**	Reserved.
[3]	**BUSWD**	**SDRAM Data Bus Width** This bit defines if the data bus width of SDRAM is 16 bit or 32 bit. The DDR and DDR2 type SDRAM only support 16 bit data bus width and this bit will be 1'b0. In this chip, SDRAM controller only supports 16 bit SDRAM. So, this bit will be fixed at 1ype SDRAM only support 16 bit data bus width and t0 = 16bits SDRAM data BUS width (Default). 1 = Reserved. **Note:** In register SDIC_SIZE1, this field is reserved.

[2:0]	**DRAMSIZE**	**Size of SDRAM Device** This field indicates the size of SDRAM device. The default memory size is 2MB or 16MB depend on power on setting value. If the power on setting value indicates the SDRAM type is DDR/DDR2, the default size is 16MB (8Mx16). Otherwise, the default size is 2MB (1Mx16). 000 = SDRAM controller Disabled. 001 = 2M Bytes. 010 = 4M Bytes. 011 = 8M Bytes. 100 = 16M Bytes. 101 = 32M Bytes. 110 = 64M Bytes. 111 = 128M Bytes. **Note:** In register SDIC_SIZE1, this field is reserved.

SDRAM SIZE	**DDR SDRAM**	**DDR2 SDRAM**
2MB	Reserved	Reserved
4MB	Reserved	Reserved
8MB	Reserved	Reserved
16MB	8Mx16 (128Mbits)	8Mx16 (128Mbits)
32MB	16Mx16 (256Mbits)	16Mx16 (256Mbits)
64MB	32Mx16 (512Mbits)	32Mx16 (512Mbits)
128MB	64Mx16 (1Gbits)	64Mx16 (1Gbits)

Table 6.5-3 SDRAM Type (Byte) Table

SDRAM Mode Register (SDIC_MR)

The SDRAM mode registers is used to configure the Mode Register of SDRAM device. This Mode Register value will be applied to both SDRAM 0 and SDRAM 1 devices.

Write this register, the SDRAM controller will generate a Load Mode Register (LMR) command to the SDRAM device.

Register	Offset	R/W	Description	Reset Value
SDIC_MR	SDIC_BA + 0x018	R/W	SDRAM Mode Register	0x0000_0032

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

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13:7]	**Configure**	**SDRAM Dependent Configuration** The value of this field is SDRAM type dependent. The definition of bits in this field is different between SDR SDRAM, DDR SDRAM and DDR2 SDRAM. Please refer the SDRAM initial sequence and related SDRAM specification to know what value should be configured in this field.
[6:4]	**LATENCY**	**CAS Latency** This field defines the CAS latency parameter of external SDRAM device. For SDR type SDRAM 010 = CAS latency is 2. 011 = CAS latency is 3. 100 = CAS latency is 4. Others = Reserved. For DDR type SDRAM 010 = CAS latency is 2. 011 = CAS latency is 3. Others = Reserved. For DDR2 type SDRAM 011 = CAS latency is 3. 100 = CAS latency is 4. Others = Reserved.

[3]	**BrstType**	**Burst Type** This bit indicates the burst type of SDRAM device is sequential or interleaved. In this chip, the SDRAM controller only support sequential burst type and this bit will be fixed at 0. 0 = Sequential burst type. (Default) 1 = Reserved.
[2:0]	**BrstLength**	**Burst Length** This field defines the burst length of external SDRAM device. SDRAM controller only supports the burst length 4. Setting burst length to be other value is inhibited. 010 = 4. Others = Reserved.

SDRAM Extended Mode Register (SDIC_EMR)

The SDRAM Extended Mode Register is used to configure SDRAM Extend Mode Register. This Extended Mode Register value will be applied to both SDRAM 0 and SDRAM 1 devices.

Write this register, the SDRAM controller will generate a Load Mode Register (LMR) command to the SDRAM. This Extended Mode Register is only used for DDR and DDR2 SDRAM.

Register	Offset	R/W	Description	Reset Value
SDIC_EMR	SDIC_BA + 0x01C	R/W	SDRAM Extended Mode Register	0x0000_4000

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

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13:2]	**Configure**	**SDRAM Dependent Configuration** The value of this field is SDRAM type dependent. The definition of bits in this field is different between DDR SDRAM and DDR2 SDRAM. Please refer the SDRAM initial sequence and related SDRAM specification to know what value should be configured in this field.
[1]	**DrvStrength**	**Output Drive Strength** This bit sets the SDRAM output drive strength. 0 = Normal drive strength. 1 = Reduced drive strength.
[0]	**DLLEN**	**DLL Enable Bit** This bit is to enable or disable the DLL of SDRAM device. 0 = DLL of SDRAM device Enabled. 1 = DLL of SDRAM device Disabled.

SDRAM Extended Mode Register 2 (SDIC_EMR2)

The SDRAM Extended Mode Register 2 is used to configure SDRAM Extend Mode Register 2. This Extended Mode Register 2 value will be applied to both SDRAM 0 and SDRAM 1 devices.

Write this register, the SDRAM controller will generate a Load Mode Register (LMR) command to the SDRAM. This Extended Mode Register is only used for DDR2 SDRAM.

Register	Offset	R/W	Description	Reset Value
SDIC_EMR2	SDIC_BA + 0x020	R/W	SDRAM Extended Mode Register 2	0x0000_8000

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

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13:0]	**Configure**	**SDRAM Dependent Configuration** The value of this field is SDRAM type dependent and only available for DDR2 SDRAM. Please refer to the SDRAM initial sequence and related SDRAM specification to know what value should be configured in this field.

SDRAM Extended Mode Register 3 (SDIC_EMR3)

The SDRAM Extended Mode Register 3 is used to configure SDRAM Extend Mode Register 3. This Extended Mode Register 3 value will be applied to both SDRAM 0 and SDRAM 1 devices.

Write this register, the SDRAM controller will generate a Load Mode Register (LMR) command to the SDRAM. This Extended Mode Register is only used for DDR2 SDRAM.

Register	Offset	R/W	Description	Reset Value
SDIC_EMR3	SDIC_BA + 0x024	R/W	SDRAM Extended Mode Register 3	0x0000_C000

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

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17:15]	**MR_DEF**	**Mode Register Definition** For Extended Mode Register 3, this field is fixed at 3AM.th S
[14:0]	**Configure**	**SDRAM Dependent Configuration** The value of this field is SDRAM type dependent and only available for DDR2 SDRAM. Please refer to the SDRAM initial sequence and related SDRAM specification to know what value should be configured in this field.

SDRAM Timing Control Register (SDIC_TIME)

This timing control register defines some SDRAM timing parameters that should be followed during SDRAM access. These timing parameters are SDRAM dependent. Refer to SDRAM devicend related SDRAM specification to know what value should be configured.

Register	Offset	R/W	Description	Reset Value
SDIC_TIME	SDIC_BA + 0x028	R/W	SDRAM Timing Control Register	0x2BDE_9649

31	30	29	28	27	26	25	24
Reserved	**tWTR**		**tRRD**		**tRC**		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
tRC		**tXSR**					**tRFC**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
tRFC				**tRAS**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
tRCD			**tRP**			**tWR**	

Bits	Description	
[31]	**Reserved**	Reserved.
[30:29]	**tWTR**	**Internal Write to Read Command Delay** This timing defines the minimum delay latency from last write data to next new valid READ command and only takes effect while SDRAM type is DDR or DDR2. tWTR = tHCLK * (tWTR+1). HCLK = It's the operating clock of SDRAM controller.
[28:27]	**tRRD**	**Active Bank a to Active Bank B Command Delay** This timing defines the minimum delay latency between SDRAM bank a ACTIVE command to SDRAM bank B ACTIVE command. tRRD = tHCLK * (tRRD+1). HCLK = It's the operating clock of SDRAM controller.
[26:22]	**tRC**	**Active to Active Command Delay** This timing defines the minimum delay latency between two ACTIVE commands. tRC = tHCLK * (tRC+1). HCLK = It's the operating clock of SDRAM controller.
[21:17]	**tXSR**	**Exit SELF REFRESH to ACTIVE Command Delay** This timing defines the minimum delay latency from SDRAM exiting self refresh mode to next valid ACTIVE command. tXSR = tHCLK * (tXSR+1). HCLK = It's the operating clock of SDRAM controller.

[16:12]	**tRFC**	**AUTO REFRESH Period** This timing defines the minimum delay latency from AUTO-REFRESH command to any other command. tRFC = tHCLK * (tRFC+1). HCLK = It's the operating clock of SDRAM controller.
[11:8]	**tRAS**	**ACTIVE to PRECHARGE Command Delay** This timing defines the minimum delay latency from a valid ACTIVE command to PRECHARGE command. tRAS = tHCLK * (tRAS+1). HCLK = It's the operating clock of SDRAM controller.
[7:5]	**tRCD**	**Active to READ or WRITE Delay** This timing defines the minimum delay latency from a ACTIVE command to READ or WRITE command. tRCD = tHCLK * (tRCD+1). HCLK = It's the operating clock of SDRAM controller.
[4:2]	**tRP**	**PRECHARGE Command Period** This timing defines the minimum delay latency from PRECHARGE command to any other command. tRP = tHCLK * (tRP+1). HCLK = It's the operating clock of SDRAM controller.
[1:0]	**tWR**	**WRITE Recovery Time** This timing defines the minimum delay latency from last valid write data to PRECHARGE command. tWR = tHCLK * (tWR+1). HCLK = It's the operating clock of SDRAM controller.

DQS Output Delay Selection Register (SDIC_DQSODS)

This register controls the DQS output delay and source selection circuit for DQS0 and DQS1 output generation. This control register only takes effect while SDRAM type is DDR or DDR2. The function equivalent circuit for DQS output delay selection is shown below. There are two same circuits in the SDRAM controller. One is for DQS0 generation and the other is for DQS1 generation.

Register	Offset	R/W	Description	Reset Value
SDIC_DQSODS	SDIC_BA + 0x030	R/W	DQS Output Delay Selection Register	0x0000_1010

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**DQSInvEn**	**DQS1_ODS**				
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**DQS0_ODS**				

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13]	**DQSInvEn**	**DQS Invert Enable Bit** This bit controls if the clock DRAM_CLK or DDR_CLK/2 is inverted for DQS0 output generation while DQS0_ODS [3:0] is 4'b0000. This control bit takes the same effect for DQS1 output generation while DQS1_ODS [11:8] is 4'b0000. 0 = DRAM_CLK and DDR_CLK/2 is not inverted for DQS0/DQS1 output generation. (Default) 1 = DRAM_CLK and DDR_CLK/2 is inverted for DQS0/DQS1 output generation.
[12:8]	**DQS1_ODS**	**DQS1 Output Delay Selection** This field controls the DQS1 output delay value and source selection circuit for DQS1 output generation. Figure 6.5-2 shows the reference circuit for this function. 00000 = The DQS1 is generated from DRAM_CLK. 10000 = The DQS1 is generated from DDR_CLK/2. Others = If DQS1_ODS[12] is 0, The DQS1 is generated from DRAM_CLK with a delay value and the delay value is controlled by the equation DQS1 delay = DQS1_ODS[11:8] * DelayCLKMUX. If DQS1_ODS[12] is 1, The DQS1 is generated from DDR_CLK/2 with a delay value and the delay value is controlled by the equation DQS1 delay = DQS1_ODS[11:8] * DelayCLKMUX. **Note:** DelayCLKMUX: It's the gate delay of a CLKMUX gate.
[7:5]	**Reserved**	Reserved.

[4:0]	DQS0_ODS	**DQS0 Output Delay Selection** This field controls the DQS0 output delay value and source selection circuit for DQS0 output generation. Figure 6.5-2 shows the reference circuit for this function. 00000 = The DQS0 is generated from DRAM_CLK. 10000 = The DQS0 is generated from DDR_CLK/2. Others = If DQS0_ODS[12] is 0, The DQS0 is generated from DRAM_CLK with a delay value and the delay value is controlled by the equation DQS0 delay = DQS0_ODS[11:8] * DelayCLKMUX. If DQS0_ODS[12] is 1, The DQS0 is generated from DDR_CLK/2 with a delay value and the delay value is controlled by the equation DQS0 delay = DQS0_ODS[11:8] * DelayCLKMUX. **Note:** DelayCLKMUX: It's the gate delay of a CLKMUX gate.

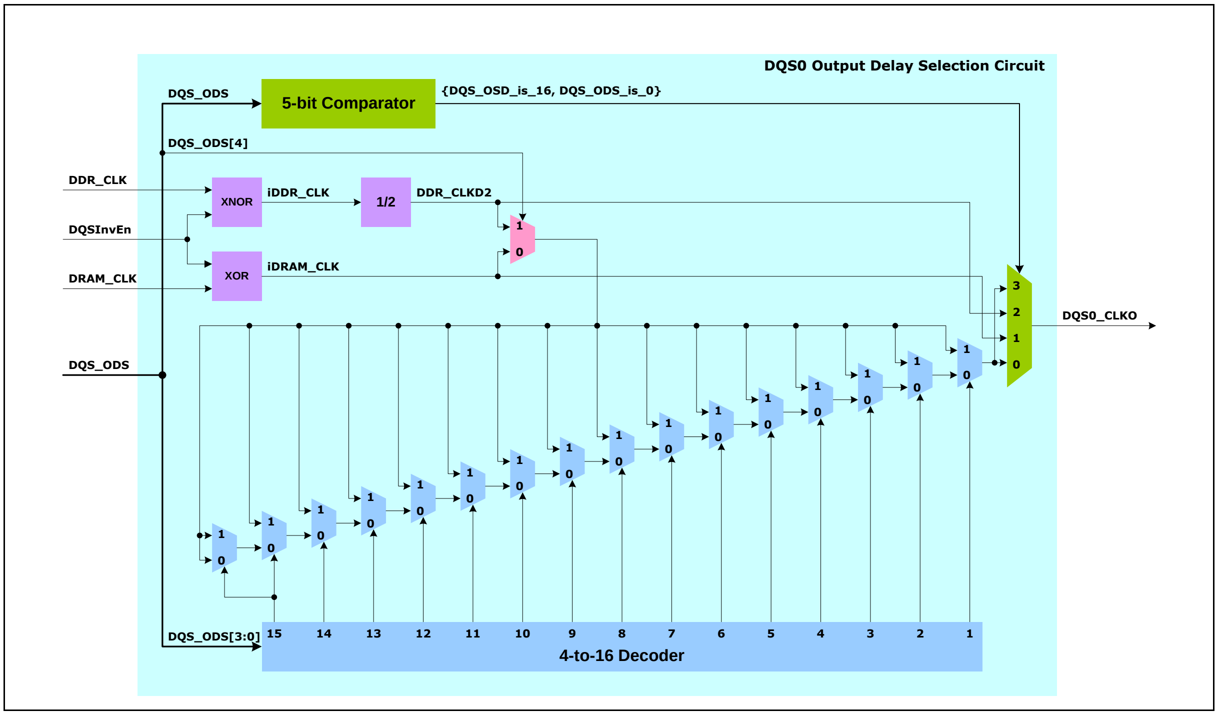


Figure 6.5-2 Clock Delay Circuit

Clock and DQS Delay Selection Register (SDIC_CKDQSDS)

Register	Offset	R/W	Description	Reset Value
SDIC_CKDQSDS	SDIC_BA + 0x034	R/W	Clock and DQS Delay Selection Register	0x0044_4400

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DQS1_DS1				**DQS1_DS0**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DQS0_DS1				**DQS0_DS0**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DCLK_DS				**DCLKSrcSel**	**MCLK_ODS**		

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:20]	**DQS1_DS1**	**DQS1 Input Delay Selection 1** This field controls the DQS1 input delay selection circuit to generate a clock signal DQS11_CLKIn. DQS11_CLKIn is used to sample the data bits [15:12] outputted by SDRAM device. This field only takes effect while the SDRAM type is DDR or DDR2. This delay value is controlled by the following equation: DQS11_CLKIn delay = DQS1_DS1 * DelayCLKMUX. DelayCLKMUX = It's the gate delay of a CLKMUX gate.
[19:16]	**DQS1_DS0**	**DQS1 Input Delay Selection 0** This field controls the DQS1 input delay selection circuit to generate a clock signal DQS10_CLKIn. DQS10_CLKIn is used to sample the data bits [11:8] outputted by SDRAM device. This field only takes effect while the SDRAM type is DDR or DDR2. This delay value is controlled by the following equation: DQS10_CLKIn delay = DQS1_DS0 * DelayCLKMUX. DelayCLKMUX = It's the gate delay of a CLKMUX gate.
[15:12]	**DQS0_DS1**	**DQS0 Input Delay Selection 1** This field controls the DQS0 input delay selection circuit to generate a clock signal DQS01_CLKIn. DQS01_CLKIn is used to sample the data bits [7:4] outputted by SDRAM device. This field only takes effect while the SDRAM type is DDR or DDR2. This delay value is controlled by the following equation: DQS01_CLKIn delay = DQS0_DS1 * DelayCLKMUX. DelayCLKMUX = It's the gate delay of a CLKMUX gate.

[11:8]	**DQS0_DS0**	**DQS0 Input Delay Selection 0** This field controls the DQS0 input delay selection circuit to generate a clock signal DQS00_CLKIn. DQS00_CLKIn is used to sample the data bits [3:0] outputted by SDRAM device. This field only takes effect while the SDRAM type is DDR or DDR2. This delay value is controlled by the following equation: DQS00_CLKIn delay = DQS0_DS0 * DelayCLKMUX. DelayCLKMUX: It's the gate delay of a CLKMUX gate.
[7:4]	**DCLK_DS**	**Data Clock Delay Selection** This field controls the delay selection circuit to generate a clock signal DataCLK. If SDRAM type is DDR or DDR2, the DataCLK is used to sample the data registered by {DQS11_CLKIn, DQS10_CLKIn, DQS01_CLKIn, DQS00_CLKIn}. Or, the DataCLK is used to sample the data outputted by SDRAM device. This control field only takes effect while DCLKSrcSel is set low. The delay value is controlled by the following equation: DataCLK delay = DCLK_DS * DelayCLKMUX. DelayCLKMUX = It's the gate delay of a CLKMUX gate.
[3]	**DCLKSrcSel**	**Data Clock Source Selection** This bit controls if the DataCLK source is from HCLK or MCLK (from SDRAM clock MCLK I/O buffer). 0 = DataCLK is from MCLK. (Default) 1 = DataCLK is from HCLK.
[2:0]	**MCLK_ODS**	**MCLK Output Delay Selection** This field controls the delay selection circuit for SDRAM clock MCLK generation. The delay value is controlled by the following equation: MCLK delay = MCLK_ODS * DelayCLKMUX. DelayCLKMUX = It's the gate delay of a CLKMUX gate.

Data Latch Enable Selection Register (SDIC_DAENSEL)

Register	Offset	R/W	Description	Reset Value
SDIC_DAENSEL	SDIC_BA + 0x038	R/W	Data Latch Enable Selection Register	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**DALATDLY**	**Data Latch Delay 1 MCLK Enable Bit** 0 = Data latch delay 1 MCLK Disabled. 1 = Data latch delay 1 MCLK Enabled.
[6:4]	**Reserved**	Reserved.
[3:0]	**DALATDS**	**Data Latch Enable Delay Selection** This field controls the delay selection circuit for enable to latch the data from SDRAM. The delay value is controlled by the following equation: Data Latch Enable Delay = DALATDS * DelayCLKMUX. DelayCLKMUX = It's the gate delay of a CLKMUX gate.

6.6 External Bus Interface (EBI)

6.6.1 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 three chip selects that can connect three external devices with different timing setting requirements.

6.6.2 Features

- Supports up to three 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 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

6.6.3 Block Diagram

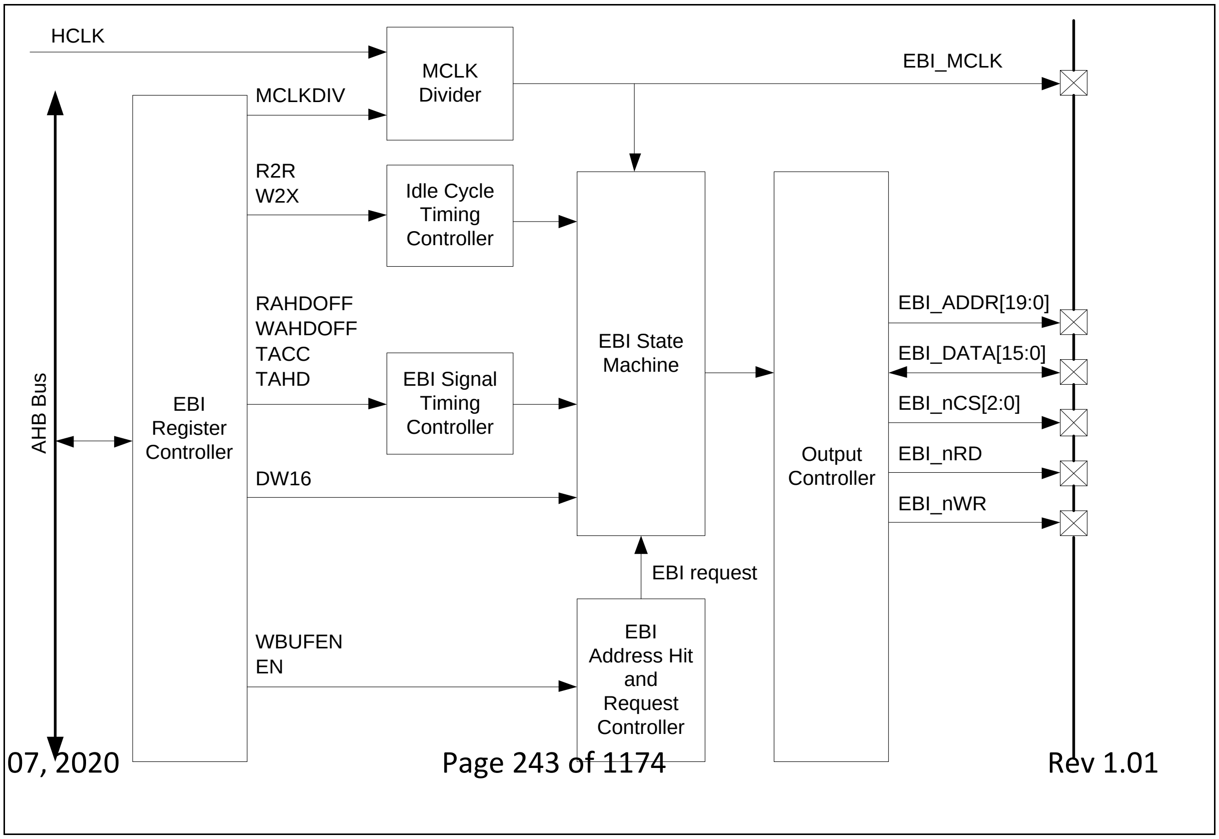


Figure 6.6-1 EBI Block Diagram

6.6.4 Basic Configuration

- Clock Source Configuration
  - Enable EBI controller clock in EBI (CLK_HCLKEN[9]).
- Reset Configuration
  - Reset EBI controller in EBIRST (SYS_AHBIPRST[3]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
EBI	EBI_ADDR0	PG.0	MFP1
	EBI_ADDR1	PG.1	MFP1
	EBI_ADDR2	PB.2, PG.2	MFP1
	EBI_ADDR3	PG.3	MFP1
	EBI_ADDR4	PG.6	MFP1
	EBI_ADDR5	PG.7	MFP1
	EBI_ADDR6	PG.8	MFP1
	EBI_ADDR7	PG.9	MFP1
	EBI_ADDR8	PA.12	MFP1
	EBI_ADDR9	PA.11	MFP1
	EBI_ADDR10	PA.10	MFP1
	EBI_ADDR11	PB.8	MFP1
	EBI_ADDR12	PB.0, PG.5	MFP1
	EBI_ADDR13	PA.13, PB.6	MFP1
	EBI_ADDR14	PA.14, PB.4	MFP1
	EBI_ADDR15	PB.7	MFP1
	EBI_ADDR16	PB.5	MFP1
	EBI_ADDR17	PB.1	MFP1
	EBI_ADDR18	PB.3, PG.4	MFP1
	EBI_ADDR19	PA.15	MFP1
	EBI_DATA0	PC.0, PG.10	MFP1
		PB.13	MFP8
	EBI_DATA1	PC.1	MFP1
		PD.12	MFP8
	EBI_DATA2	PC.2	MFP1

		PD.13	MFP8
	EBI_DATA3	PC.3	MFP1
		PD.14	MFP8
	EBI_DATA4	PC.4	MFP1
		PD.15	MFP8
	EBI_DATA5	PC.5	MFP1
		PF.0	MFP8
	EBI_DATA6	PC.6	MFP1
		PF.1	MFP8
	EBI_DATA7	PC.7	MFP1
		PF.2	MFP8
	EBI_DATA8	PC.8	MFP1
		PF.3	MFP8
	EBI_DATA9	PC.9	MFP1
		PF.4	MFP8
	EBI_DATA10	PC.10	MFP1
		PF.5	MFP8
	EBI_DATA11	PC.11	MFP1
		PF.6	MFP8
	EBI_DATA12	PC.12	MFP1
		PF.7	MFP8
	EBI_DATA13	PC.13	MFP1
		PF.8	MFP8
	EBI_DATA14	PC.14	MFP1
		PF.9	MFP8
	EBI_DATA15	PC.15	MFP1
		PF.10	MFP8
	EBI_MCLK	PA.1	MFP2
	EBI_nCS0	PA.9	MFP1
	EBI_nCS1	PA.6	MFP1
	EBI_nCS2	PA.1	MFP1
	EBI_nRE	PA.8	MFP1
	EBI_nWE	PA.7	MFP1

6.6.5 Functional Description

6.6.5.1 EBI Area and Address Hit

The EBI mapping address is located at 0x6000_0000 ~ 0x602F_FFFF and the total memory space is 3MB. 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
EBI_nCS2	0x6020_0000 ~ 0x602F_FFFF

Table 6.6-1 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.

6.6.5.2 EBI Data Width Connection - Address Bus and Data Bus Separate Mode

The EBI supports address and data bus separate mode. EBI_DATA is dedicated for data bus and connected directly to device data bus, EBI_ADDR is dedicated for address bus.

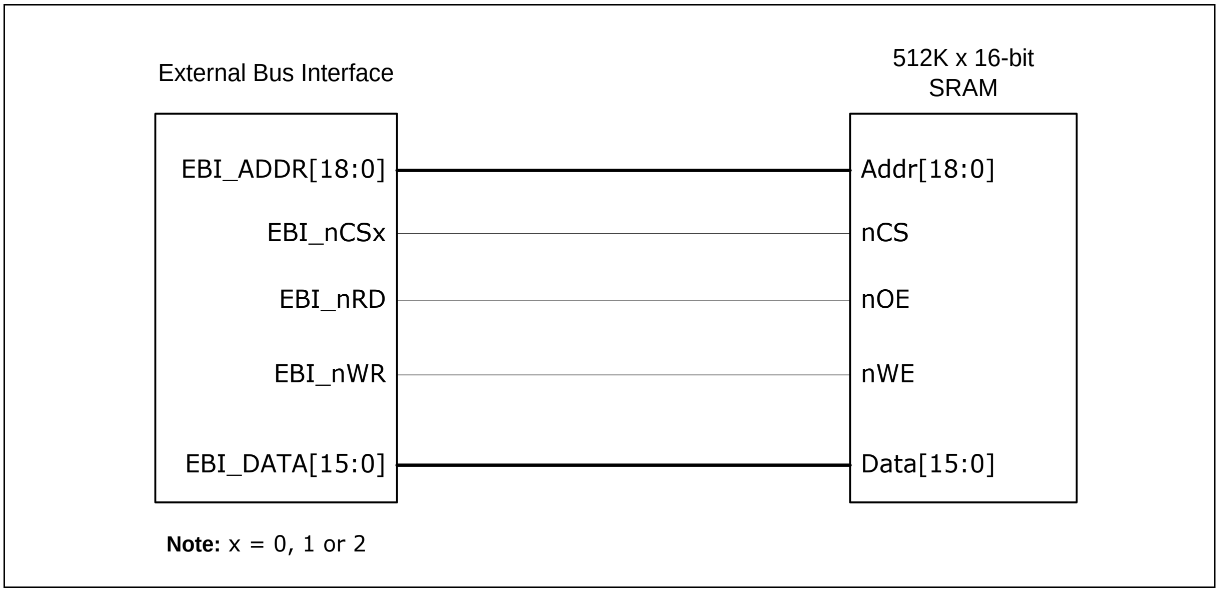


Figure 6.6-2 Connection of 16-bit EBI Data Width with 16-bit Device in Separate mode

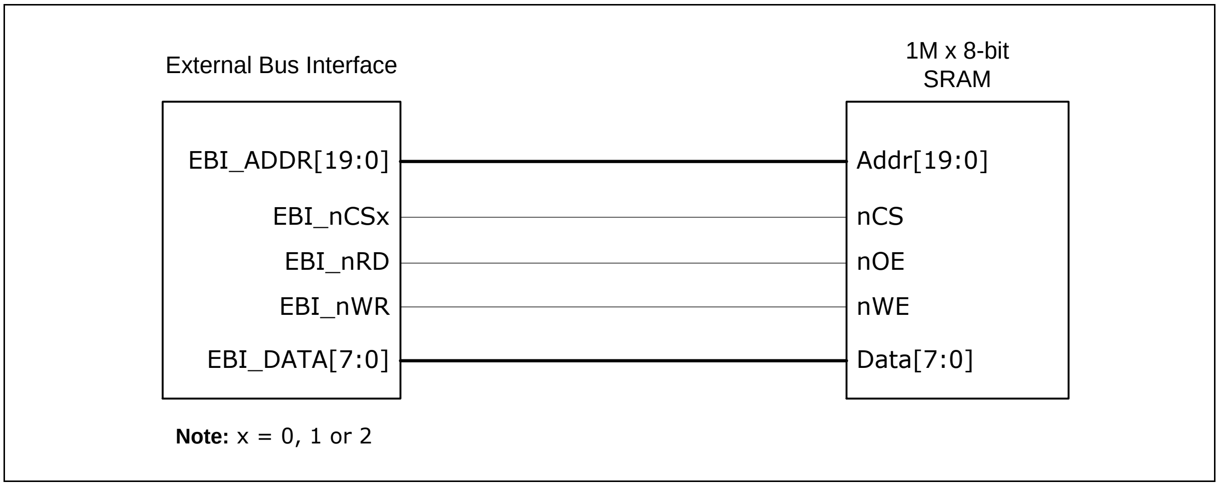


Figure 6.6-3 Connection of 8-bit EBI Data Width with 8-bit Device in Separate mode

6.6.5.3 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/32 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_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 is 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]). 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]).

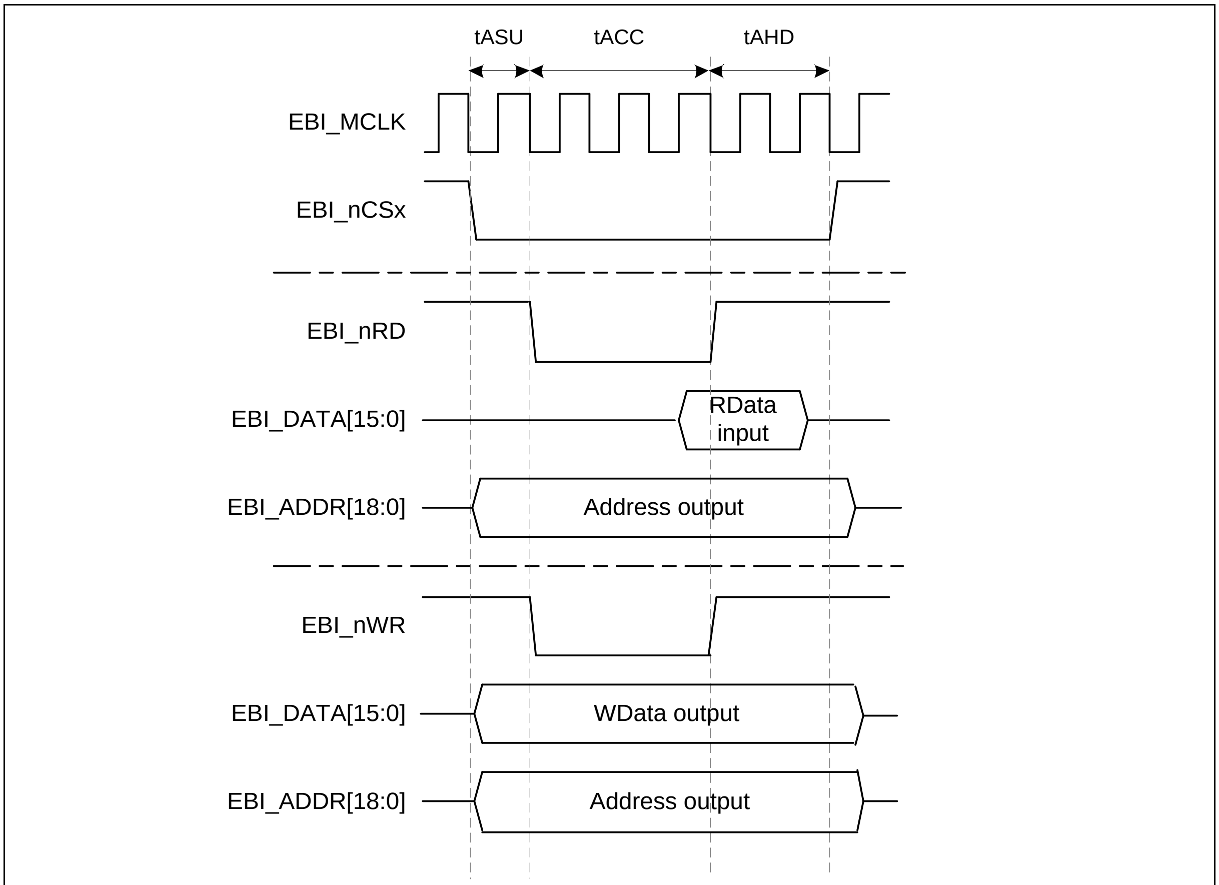
For each chip select, the EBI provides individual register with timing control except that tALE can only be controlled by EBI_CTL0.

Parameter	Value	Unit	Description
tASU	1	MCLK	Address Latch Setup Time.
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 6.6-2 Timing Control Parameter

Address Data Separate Mode

EBI_DATA and EBI_ADDR are dedicated for data and address bus separately.



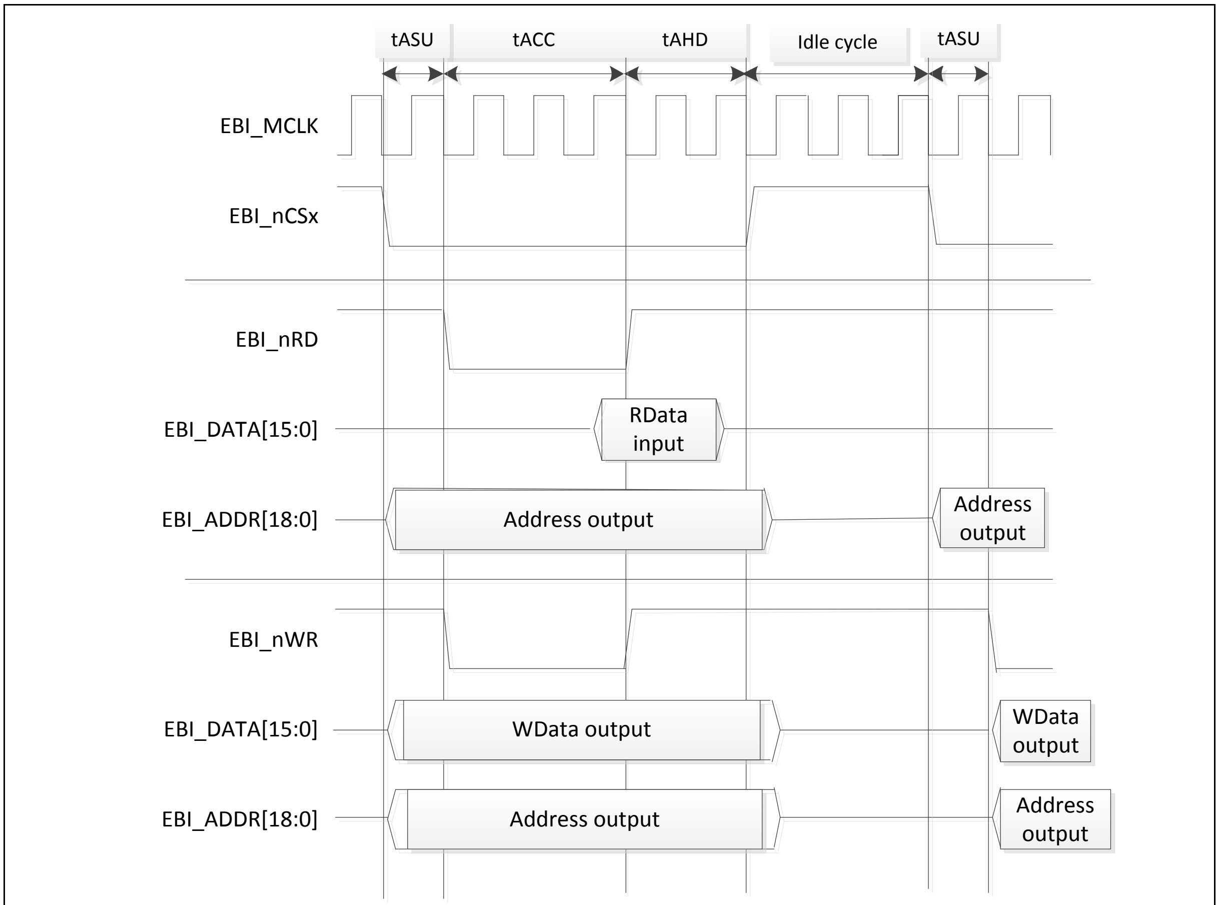
Note1: The EBI_MCLK is HCLK/2, MCLKDIV (EBI_CTLx[10:8]) = 1
Note2: x = 0, 1 or 2

Figure 6.6-4 Timing Control Waveform for 16-bit Data Width for Separate Mode

The difference between 8-bit and 16-bit data width is EBI_DATA[15:8]. In 8-bit data width setting, EBI_DATA[15:8] is always Address [15:8] output so that external latch needs only 8-bit width.

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 6.6-5 shows idle cycles.



Note1: The EBI_MCLK is HCLK/2, MCLKDIV (EBI_CTLx[10:8]) = 1
Note2: x = 0, 1 or 2

Figure 6.6-5 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]).

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 cycle is 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 6.6-6.

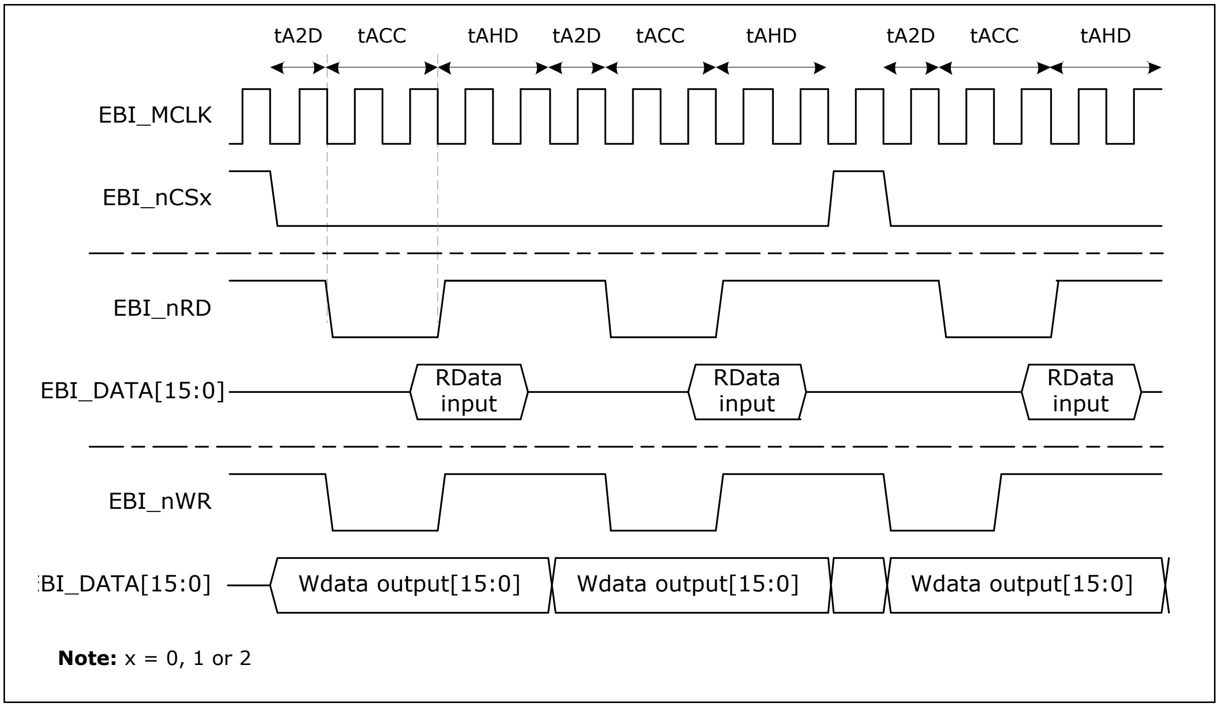


Figure 6.6-6 Timing Control Waveform for Continuous Data Access Mode

6.6.6 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 = 0xB001_0000**				
EBI_CTL0	EBI_BA+0x00	R/W	External Bus Interface Bank0 Control Register	0x0000_0008
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_0008
EBI_TCTL1	EBI_BA+0x14	R/W	External Bus Interface Bank1 Timing Control Register	0x0000_0000
EBI_CTL2	EBI_BA+0x20	R/W	External Bus Interface Bank2 Control Register	0x0000_0008
EBI_TCTL2	EBI_BA+0x24	R/W	External Bus Interface Bank2 Timing Control Register	0x0000_0000

6.6.7 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_0008
EBI_CTL1	EBI_BA+0x10	R/W	External Bus Interface Bank1 Control Register	0x0000_0008
EBI_CTL2	EBI_BA+0x20	R/W	External Bus Interface Bank2 Control Register	0x0000_0008

31	30	29	28	27	26	25	24
Reversed							WBUFEN
23	22	21	20	19	18	17	16
Reversed					Reserved		
15	14	13	12	11	10	9	8
Reversed					MCLKDIV		
7	6	5	4	3	2	1	0
Reversed			CACCESS	Reserved	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 only available in EBI_CTL0 register
[23:11]	**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 con tenuous access mode enabled, the tASU cycle is bypass for continuous data transfer request.

		0 = Continuous data access mode Disabled. 1 = Continuous data access mode Enabled.
[3]	**Reserved**	Reserved.
[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
EBI_TCTL2	EBI_BA+0x24	R/W	External Bus Interface Bank2 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). 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). 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.

6.7 General Purpose I/O (GPIO)

6.7.1 Overview

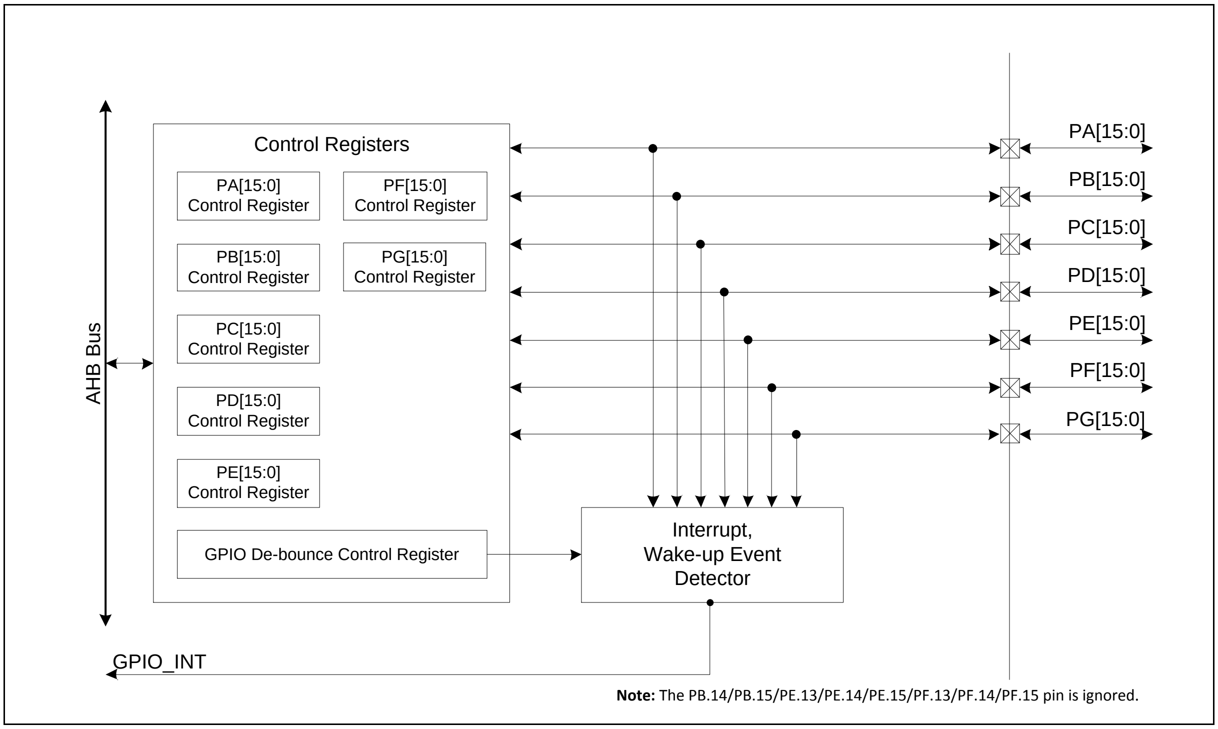
This chip has up to 104 General-Purpose I/O (GPIO) pins and can be shared with other function pins depending on the chip configuration. These 104 pins are arranged in 7 ports named as PA, PB, PC, PD, PE, PF and PG. PA, PC, PD and PG has 16 pins on port. PB has 14 pins on port. PE and PF has 13 pins on port. Each of the 104 I/O pins is independent and can be easily configured by user to meet various system configurations and design requirements.

The I/O type of each of I/O pins can be configured by software individually as Input, Push-pull output or Open-drain output. After reset, all 104 I/O pins are configured in General-Purpose I/O Input mode. Each I/O pin has a very weakly individual pull-up resistor which is about 50 kΩ for V_{DD} is from 5.5 V to 1.65 V.

6.7.2 Features

- Three I/O modes:
  - Push-Pull Output mode
  - Open-Drain Output mode
  - Input only with high impendence mode
- TTL/Schmitt trigger input selectable
- I/O pin can be configured as interrupt source with edge/level setting
- Supports High Drive and High Slew Rate I/O mode
- Enabling the pin interrupt function will also enable the wake-up function

6.7.3 Block Diagram


Control Registers
PA[15:0] Control Register
PF[15:0] Control Register
PB[15:0] Control Register
PG[15:0] Control Register
PC[15:0] Control Register
PD[15:0] Control Register
PE[15:0] Control Register
GPIO De-bounce Control Register
AHB Bus
Interrupt, Wake-up Event Detector
GPIO_INT
PA[15:0]
PB[15:0]
PC[15:0]
PD[15:0]
PE[15:0]
PF[15:0]
PG[15:0]

Note: The PB.14/PB.15/PE.13/PE.14/PE.15/PF.13/PF.14/PF.15 pin is ignored.

Figure 6.7-1 GPIO Block Diagram

6.7.4 Basic Configuration

- Clock source configuration
  - Enable GPIO clock in CLK_HCLKEN[11]
- Reset configuration
  - Reset GPIO in GPIORST SYS_AHBIPRST [7]
- Pin configuration

Group	Pin Name	GPIO	MFP
EINT0	EINT0	PA.0	MFP5
		PA.13	MFP8
EINT1	EINT1	PA.1	MFP5
		PA.14	MFP8
EINT2	EINT2	PB.13	MFP2
		PB.3	MFP3
		PD.0	MFP4
		PE.10	MFP5

EINT3	EINT3	PD.1, PG.15	MFP4
		PE.12	MFP5

To configure pin Px.n as a General-Purpose I/O, set the corresponding field of register SYS_GPA_MFPL, SYS_GPA_MFPH, SYS_GPB_MFPL, SYS_GPB_MFPH, SYS_GPC_MFPL, SYS_GPC_MFPH, SYS_GPD_MFPL, SYS_GPD_MFPH, SYS_GPE_MFPL, SYS_GPE_MFPH, SYS_GPF_MFPL, SYS_GPF_MFPH, SYS_GPG_MFPL and SYS_GPG_MFPH to 0.

For example, if user want to configure pin PA.0 as a General-Purpose I/O, it's necessary to set MFP_GPA0 (SYS_GPA_MFPL[3:0]) to 0.Functional Description

6.7.4.1 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.

6.7.4.2 Push-pull Output Mode

Figure 6.7-2 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.

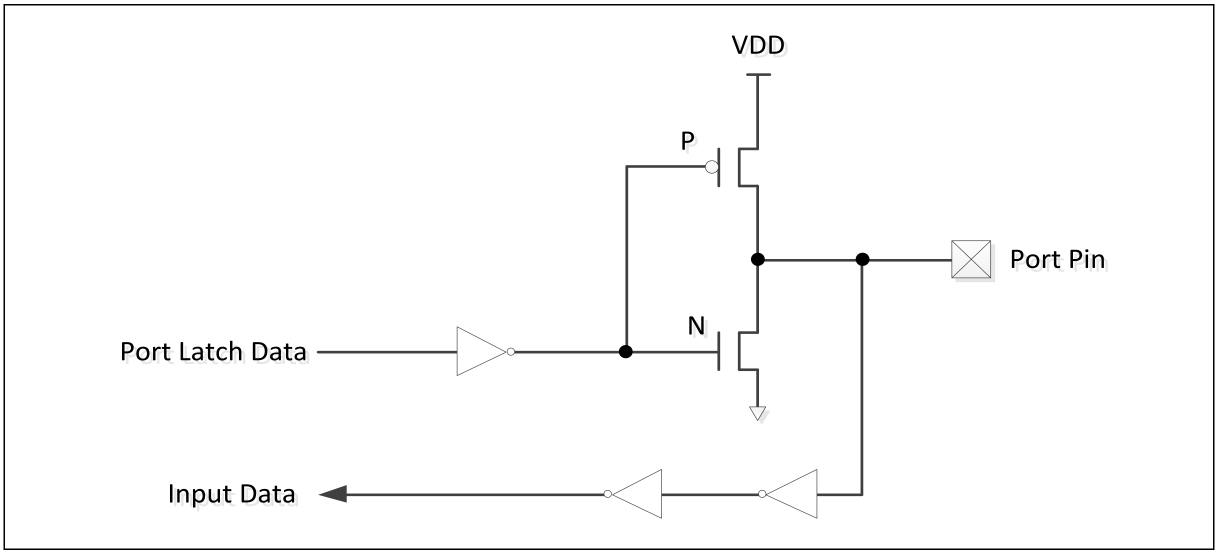


Figure 6.7-2 Push-Pull Output

6.7.4.3 Open-drain Mode

Figure 6.7-3 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.

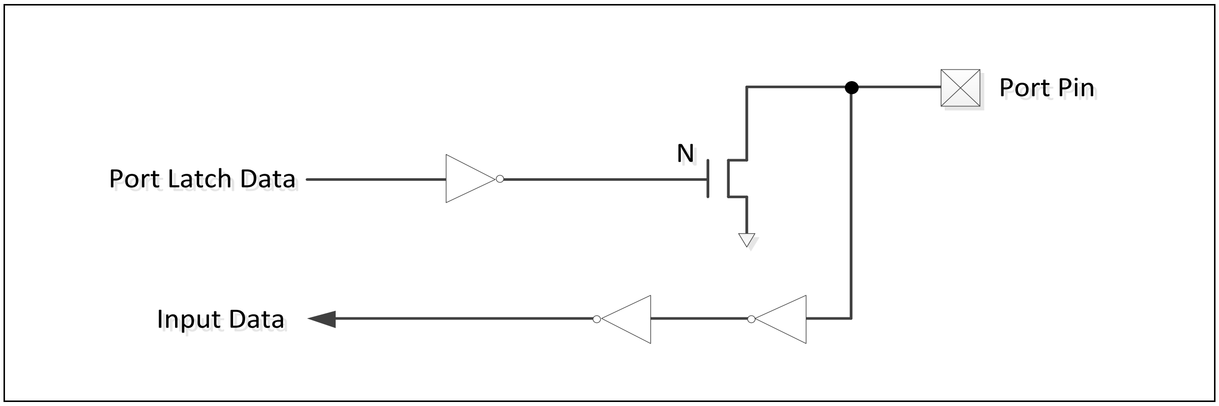


Figure 6.7-3 Open-Drain Output

6.7.4.4 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.

6.7.4.5 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. 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 LXT by setting DBCLKSRC (GPIO_DBCTL[4]) register. And DBCLKSEL (GPIO_DBCTL[3:0]) register can control sampling cycle period.

Figure 6.7-4 shows GPIO rising edge trigger interrupt. The interval of time between the two valid sample signal is determined by DBCLKSRC (GPIO_DBCTL[4]) and DBCLKSEL (GPIO_DBCTL[3:0]). Each valid data from GPIO pin need to be sample twice. For rising edge setting, if pin status is low before setting DBEN (Px_DBEN), 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 6.7-5 shows the situation is opposite to rising edge trigger.

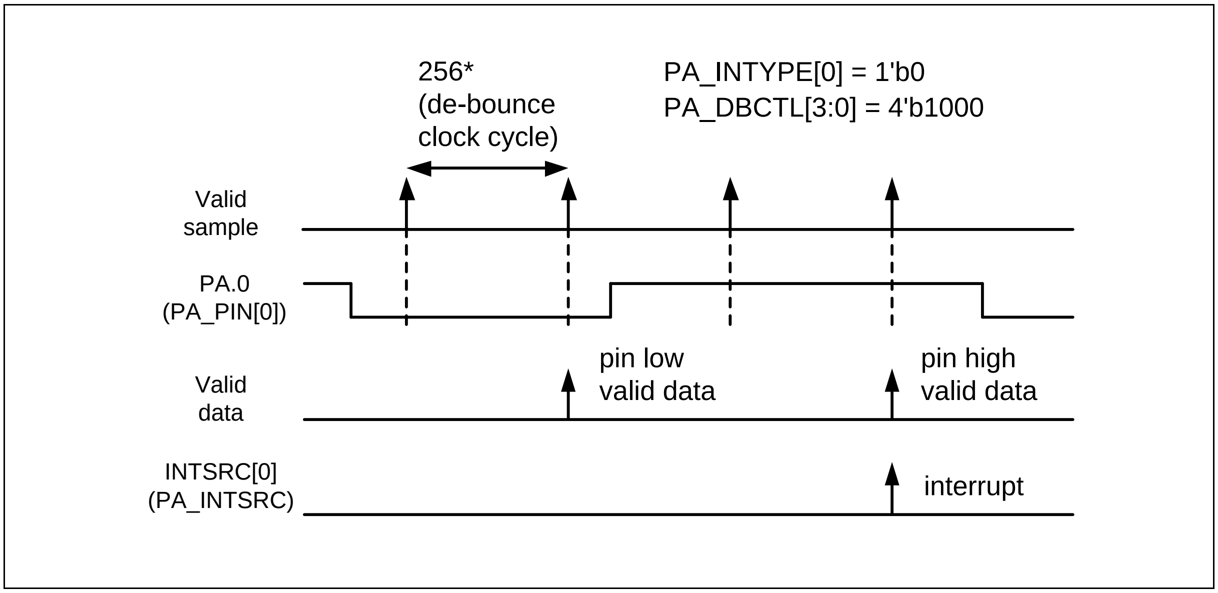


Figure 6.7-4 GPIO Rising Edge Trigger Interrupt

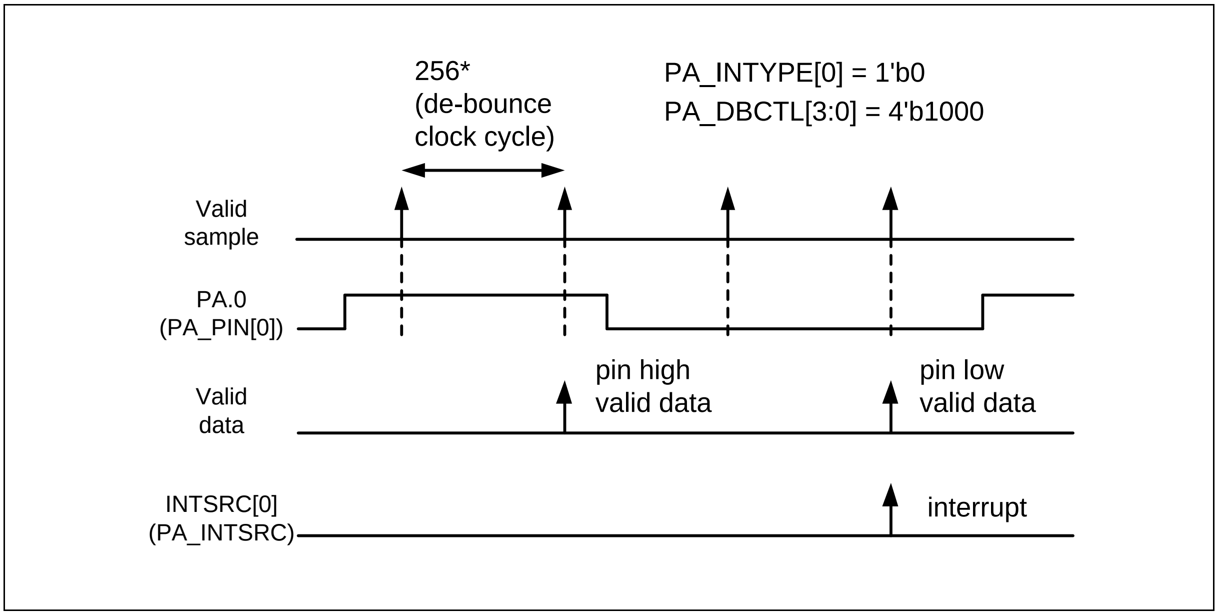


Figure 6.7-5 GPIO Falling Edge Trigger Interrupt

6.7.4.6 GPIO Digital Input Path Disable Control

User can disable GPIO digital input path by setting DINOFF (Px_DINOFF[n]). 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

6.7.5 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 = 0xB000_4000**				
PA_MODE	GPIO_BA+0x000	R/W	PA I/O Mode Control	0x0000_0000
PA_DINOFF	GPIO_BA+0x004	R/W	PA Digital Input Path Disable Control	0x0000_0000
PA_DOUT	GPIO_BA+0x008	R/W	PA Data Output Value	0x0000_FFFF
PA_DATMSK	GPIO_BA+0x00C	R/W	PA Data Output Write Mask	0x0000_0000
PA_PIN	GPIO_BA+0x010	R	PA Pin Value	0x0000_XXXX
PA_DBEN	GPIO_BA+0x014	R/W	PA De-bounce Enable Control Register	0x0000_0000
PA_INTTYPE	GPIO_BA+0x018	R/W	PA Interrupt Trigger Type Control	0x0000_0000
PA_INTEN	GPIO_BA+0x01C	R/W	PA Interrupt Enable Control Register	0x0000_0000
PA_INTSRC	GPIO_BA+0x020	R/W	PA Interrupt Source Flag	0x0000_XXXX
PA_SMTEN	GPIO_BA+0x024	R/W	PA Input Schmitt Trigger Enable Register	0x0000_0000
PA_SLEWCTL	GPIO_BA+0x028	R/W	PA High Slew Rate Control Register	0x0000_0000
PA_PUSEL	GPIO_BA+0x030	R/W	PA Pull-up and Pull-down Selection Register	0x0000_0000
PB_MODE	GPIO_BA+0x040	R/W	PB I/O Mode Control	0x0000_0000
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_3FFF
PB_DATMSK	GPIO_BA+0x04C	R/W	PB Data Output Write Mask	0x0000_0000
PB_PIN	GPIO_BA+0x050	R	PB Pin Value	0x0000_XXXX
PB_DBEN	GPIO_BA+0x054	R/W	PB De-bounce Enable Control Register	0x0000_0000
PB_INTTYPE	GPIO_BA+0x058	R/W	PB Interrupt Trigger Type Control	0x0000_0000
PB_INTEN	GPIO_BA+0x05C	R/W	PB Interrupt Enable Control Register	0x0000_0000
PB_INTSRC	GPIO_BA+0x060	R/W	PB Interrupt Source Flag	0x0000_XXXX
PB_SMTEN	GPIO_BA+0x064	R/W	PB Input Schmitt Trigger Enable Register	0x0000_0000
PB_SLEWCTL	GPIO_BA+0x068	R/W	PB High Slew Rate Control Register	0x0000_0000
PB_PUSEL	GPIO_BA+0x070	R/W	PB Pull-up and Pull-down Selection Register	0x0000_0000
PC_MODE	GPIO_BA+0x080	R/W	PC I/O Mode Control	0x0000_0000

PC_DINOFF	GPIO_BA+0x084	R/W	PC Digital Input Path Disable Control	0x0000_0000
PC_DOUT	GPIO_BA+0x088	R/W	PC Data Output Value	0x0000_FFFF
PC_DATMSK	GPIO_BA+0x08C	R/W	PC Data Output Write Mask	0x0000_0000
PC_PIN	GPIO_BA+0x090	R	PC Pin Value	0x0000_XXXX
PC_DBEN	GPIO_BA+0x094	R/W	PC De-bounce Enable Control Register	0x0000_0000
PC_INTTYPE	GPIO_BA+0x098	R/W	PC Interrupt Trigger Type Control	0x0000_0000
PC_INTEN	GPIO_BA+0x09C	R/W	PC Interrupt Enable Control Register	0x0000_0000
PC_INTSRC	GPIO_BA+0x0A0	R/W	PC Interrupt Source Flag	0x0000_XXXX
PC_SMTEN	GPIO_BA+0x0A4	R/W	PC Input Schmitt Trigger Enable Register	0x0000_0000
PC_SLEWCTL	GPIO_BA+0x0A8	R/W	PC High Slew Rate Control Register	0x0000_0000
PC_PUSEL	GPIO_BA+0x0B0	R/W	PC Pull-up and Pull-down Selection Register	0x0000_0000
PD_MODE	GPIO_BA+0x0C0	R/W	PD I/O Mode Control	0x0000_0000
PD_DINOFF	GPIO_BA+0x0C4	R/W	PD Digital Input Path Disable Control	0x0000_0000
PD_DOUT	GPIO_BA+0x0C8	R/W	PD Data Output Value	0x0000_FFFF
PD_DATMSK	GPIO_BA+0x0CC	R/W	PD Data Output Write Mask	0x0000_0000
PD_PIN	GPIO_BA+0x0D0	R	PD Pin Value	0x0000_XXXX
PD_DBEN	GPIO_BA+0x0D4	R/W	PD De-bounce Enable Control Register	0x0000_0000
PD_INTTYPE	GPIO_BA+0x0D8	R/W	PD Interrupt Trigger Type Control	0x0000_0000
PD_INTEN	GPIO_BA+0x0DC	R/W	PD Interrupt Enable Control Register	0x0000_0000
PD_INTSRC	GPIO_BA+0x0E0	R/W	PD Interrupt Source Flag	0x0000_XXXX
PD_SMTEN	GPIO_BA+0x0E4	R/W	PD Input Schmitt Trigger Enable Register	0x0000_0000
PD_SLEWCTL	GPIO_BA+0x0E8	R/W	PD High Slew Rate Control Register	0x0000_0000
PD_PUSEL	GPIO_BA+0x0F0	R/W	PD Pull-up and Pull-down Selection Register	0x0000_0000
PE_MODE	GPIO_BA+0x100	R/W	PE I/O Mode Control	0x0000_0000
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_1FFF
PE_DATMSK	GPIO_BA+0x10C	R/W	PE Data Output Write Mask	0x0000_0000
PE_PIN	GPIO_BA+0x110	R	PE Pin Value	0x0000_XXXX
PE_DBEN	GPIO_BA+0x114	R/W	PE De-bounce Enable Control Register	0x0000_0000
PE_INTTYPE	GPIO_BA+0x118	R/W	PE Interrupt Trigger Type Control	0x0000_0000

PE_INTEN	GPIO_BA+0x11C	R/W	PE Interrupt Enable Control Register	0x0000_0000
PE_INTSRC	GPIO_BA+0x120	R/W	PE Interrupt Source Flag	0x0000_XXXX
PE_SMTEN	GPIO_BA+0x124	R/W	PE Input Schmitt Trigger Enable Register	0x0000_0000
PE_SLEWCTL	GPIO_BA+0x128	R/W	PE High Slew Rate Control Register	0x0000_0000
PE_PUSEL	GPIO_BA+0x130	R/W	PE Pull-up and Pull-down Selection Register	0x0000_0000
PF_MODE	GPIO_BA+0x140	R/W	PF I/O Mode Control	0x0000_0000
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_1FFF
PF_DATMSK	GPIO_BA+0x14C	R/W	PF Data Output Write Mask	0x0000_0000
PF_PIN	GPIO_BA+0x150	R	PF Pin Value	0x0000_XXXX
PF_DBEN	GPIO_BA+0x154	R/W	PF De-bounce Enable Control Register	0x0000_0000
PF_INTTYPE	GPIO_BA+0x158	R/W	PF Interrupt Trigger Type Control	0x0000_0000
PF_INTEN	GPIO_BA+0x15C	R/W	PF Interrupt Enable Control Register	0x0000_0000
PF_INTSRC	GPIO_BA+0x160	R/W	PF Interrupt Source Flag	0x0000_XXXX
PF_SMTEN	GPIO_BA+0x164	R/W	PF Input Schmitt Trigger Enable Register	0x0000_0000
PF_SLEWCTL	GPIO_BA+0x168	R/W	PF High Slew Rate Control Register	0x0000_0000
PF_PUSEL	GPIO_BA+0x170	R/W	PF Pull-up and Pull-down Selection Register	0x0000_0000
PG_MODE	GPIO_BA+0x180	R/W	PG I/O Mode Control	0x0000_0000
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_FFFF
PG_DATMSK	GPIO_BA+0x18C	R/W	PG Data Output Write Mask	0x0000_0000
PG_PIN	GPIO_BA+0x190	R	PG Pin Value	0x0000_XXXX
PG_DBEN	GPIO_BA+0x194	R/W	PG De-bounce Enable Control Register	0x0000_0000
PG_INTTYPE	GPIO_BA+0x198	R/W	PG Interrupt Trigger Type Control	0x0000_0000
PG_INTEN	GPIO_BA+0x19C	R/W	PG Interrupt Enable Control Register	0x0000_0000
PG_INTSRC	GPIO_BA+0x1A0	R/W	PG Interrupt Source Flag	0x0000_XXXX
PG_SMTEN	GPIO_BA+0x1A4	R/W	PG Input Schmitt Trigger Enable Register	0x0000_0000
PG_SLEWCTL	GPIO_BA+0x1A8	R/W	PG High Slew Rate Control Register	0x0000_0000
PG_PUSEL	GPIO_BA+0x1B0	R/W	PG Pull-up and Pull-down Selection Register	0x0000_0000

GPIO_DBCTL	GPIO_BA+0x440	R/W	Interrupt De-bounce Control Register	0x0000_0020
PAn_PDIO **n=0,1..15**	GPIO_BA+0x800+(0x04 * n)	R/W	GPIO PA.n Pin Data Input/Output Register	0x0000_000X
PBn_PDIO **n=0,1..13**	GPIO_BA+0x840+(0x04 * n)	R/W	GPIO PB.n Pin Data Input/Output Register	0x0000_000X
PCn_PDIO **n=0,1..15**	GPIO_BA+0x880+(0x04 * n)	R/W	GPIO PC.n Pin Data Input/Output Register	0x0000_000X
PDn_PDIO **n=0,1..15**	GPIO_BA+0x8C0+(0x04 * n)	R/W	GPIO PD.n Pin Data Input/Output Register	0x0000_000X
PEn_PDIO **n=0,1..12**	GPIO_BA+0x900+(0x04 * n)	R/W	GPIO PE.n Pin Data Input/Output Register	0x0000_000X
PFn_PDIO **n=0,1..12**	GPIO_BA+0x940+(0x04 * n)	R/W	GPIO PF.n Pin Data Input/Output Register	0x0000_000X
PGn_PDIO **n=0,1..15**	GPIO_BA+0x980+(0x04 * n)	R/W	GPIO PG.n Pin Data Input/Output Register	0x0000_000X

6.7.6 Register Description

Port A-G 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	0x0000_0000
PB_MODE	GPIO_BA+0x040	R/W	PB I/O Mode Control	0x0000_0000
PC_MODE	GPIO_BA+0x080	R/W	PC I/O Mode Control	0x0000_0000
PD_MODE	GPIO_BA+0x0C0	R/W	PD I/O Mode Control	0x0000_0000
PE_MODE	GPIO_BA+0x100	R/W	PE I/O Mode Control	0x0000_0000
PF_MODE	GPIO_BA+0x140	R/W	PF I/O Mode Control	0x0000_0000
PG_MODE	GPIO_BA+0x180	R/W	PG I/O Mode Control	0x0000_0000

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-G 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 = Reserved. **Note1:** The default value is 0x0000_0000 and all pins will be input mode after chip powered on. **Note2:** The PB.14~15/PE.13~15/PF.13~15 pin is ignored.

Port A-G 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

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-G 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 PB.14~15/PE.13~15/PF.13~15 pin is ignored.
[15:0]	**Reserved**	Reserved.

Port A-G 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_3FFF
PC_DOUT	GPIO_BA+0x088	R/W	PC Data Output Value	0x0000_FFFF
PD_DOUT	GPIO_BA+0x0C8	R/W	PD Data Output Value	0x0000_FFFF
PE_DOUT	GPIO_BA+0x108	R/W	PE Data Output Value	0x0000_1FFF
PF_DOUT	GPIO_BA+0x148	R/W	PF Data Output Value	0x0000_1FFF
PG_DOUT	GPIO_BA+0x188	R/W	PG Data Output Value	0x0000_FFFF

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**DOUT[n]**	**Port A-G 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. 1 = Px.n will drive High if the Px.n pin is configured as Push-pull output. **Note:** The PB.14~15/PE.13~15/PF.13~15 pin is ignored.

Port A-G 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

31	30	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-G Pin[n] Data Output Write Mask** These bits are used to protect the corresponding DOUT (Px_DOUT[n]) bit. When the DATMSK (Px_DATMSK[n]) bit is set to 1, the corresponding DOUT (Px_DOUT[n]) bit is protected. If the write signal is masked, writing data to the protect bit is ignored. 0 = Corresponding DOUT (Px_DOUT[n]) bit can be updated. 1 = Corresponding DOUT (Px_DOUT[n]) bit protected. **Note1:** This function only protects the corresponding DOUT (Px_DOUT[n]) bit, and will not protect the corresponding PDIO (Pxn_PDIO[0]) bit. **Note2:** The PB.14~15/PE.13~15/PF.13~15 pin is ignored.

Port A-G 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

31	30	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-G 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 PB.14~15/PE.13~15/PF.13~15 pin is ignored.

Port A-G 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

31	30	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-G 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 PB.14~15/PE.13~15/PF.13~15 pin is ignored.

Port A-G 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

31	30	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-G 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 PB.14~15/PE.13~15/PF.13~15 pin is ignored.

Port A-G 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

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-G 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 PB.14~15/PE.13~15/PF.13~15 pin is ignored.
[n] n=0,1..15	**FLIEN[n]**	**Port A-G 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 PB.14~15/PE.13~15/PF.13~15 pin is ignored.

Port A-G 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

31	30	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-G 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 PB.14~15/PE.13~15/PF.13~15 pin is ignored.

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

Register	Offset	R/W	Description	Reset Value
PA_SMTEN	GPIO_BA+0x024	R/W	PA Input Schmitt Trigger Enable Register	0x0000_0000
PB_SMTEN	GPIO_BA+0x064	R/W	PB Input Schmitt Trigger Enable Register	0x0000_0000
PC_SMTEN	GPIO_BA+0x0A4	R/W	PC Input Schmitt Trigger Enable Register	0x0000_0000
PD_SMTEN	GPIO_BA+0x0E4	R/W	PD Input Schmitt Trigger Enable Register	0x0000_0000
PE_SMTEN	GPIO_BA+0x124	R/W	PE Input Schmitt Trigger Enable Register	0x0000_0000
PF_SMTEN	GPIO_BA+0x164	R/W	PF Input Schmitt Trigger Enable Register	0x0000_0000
PG_SMTEN	GPIO_BA+0x1A4	R/W	PG Input Schmitt Trigger Enable Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[n] n=0,1..15	**SMTEN[n]**	**Port A-G Pin[n] Input Schmitt Trigger Enable Bit** 0 = Px.n input schmitt trigger function Disabled. 1 = Px.n input schmitt trigger function Enabled. **Note:** The PB.14~15/PE.13~15/PF.13~15 pin is ignored.

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

Register	Offset	R/W	Description	Reset Value
PA_SLEWCTL	GPIO_BA+0x028	R/W	PA High Slew Rate Control Register	0x0000_0000
PB_SLEWCTL	GPIO_BA+0x068	R/W	PB High Slew Rate Control Register	0x0000_0000
PC_SLEWCTL	GPIO_BA+0x0A8	R/W	PC High Slew Rate Control Register	0x0000_0000
PD_SLEWCTL	GPIO_BA+0x0E8	R/W	PD High Slew Rate Control Register	0x0000_0000
PE_SLEWCTL	GPIO_BA+0x128	R/W	PE High Slew Rate Control Register	0x0000_0000
PF_SLEWCTL	GPIO_BA+0x168	R/W	PF High Slew Rate Control Register	0x0000_0000
PG_SLEWCTL	GPIO_BA+0x1A8	R/W	PG High Slew Rate Control Register	0x0000_0000

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

Bits	Description	
[2n+1:2n] n=0,1..15	**HSRENn**	**Port A-G Pin[n] High Slew Rate Control** 00 = Px.n output with normal slew rate mode (minimum 16 MHz at 2.7V). 01 = Px.n output with high slew rate mode (minimum 25 MHz at 2.7V). 10 = Reserved. 11 = Reserved. **Note:** The PB.14~15/PE.13~15/PF.13~15 pin is ignored.

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

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

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

Bits	Description	
[2n+1:2n] n=0,1..15	**PUSELn**	**Port A-G Pin[n] Pull-up and Pull-down Enable Register** Determine each I/O Pull-up/pull-down of Px.n pins. 00 = Px.n pull-up and pull-up disable. 01 = Px.n pull-up enable. 10 = Px.n pull-down enable. 11 = Px.n pull-up and pull-up disable. **Note 1:** Basically, the pull-up control and pull-down control has following behavior limitation. The independent pull-up control register only valid when MODEn set as tri-state and open-drain mode The independent pull-down control register only valid when MODEn set as tri-state mode When both pull-up pull-down is set as 1 at “tri-state” mode, keep I/O in tri-state mode **Note 2:** The PB.14~15/PE.13~15/PF.13~15 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 HXT. 1 = De-bounce counter clock source is the LXT. **Note:** This bit is reserved if the chip package without LXT. The de-bounce counter clock source is only from HXT. And setting this bit does not guarantee what will occur.

Bits	Description	
[3:0]	**DBCLKSEL**	**De-bounce Sampling Cycle Selection** 0000 = Sample interrupt input once per 1 clocks. 0001 = Sample interrupt input once per 2 clocks. 0010 = Sample interrupt input once per 4 clocks. 0011 = Sample interrupt input once per 8 clocks. 0100 = Sample interrupt input once per 16 clocks. 0101 = Sample interrupt input once per 32 clocks. 0110 = Sample interrupt input once per 64 clocks. 0111 = Sample interrupt input once per 128 clocks. 1000 = Sample interrupt input once per 256 clocks. 1001 = Sample interrupt input once per 2*256 clocks. 1010 = Sample interrupt input once per 4*256 clocks. 1011 = Sample interrupt input once per 8*256 clocks. 1100 = Sample interrupt input once per 16*256 clocks. 1101 = Sample interrupt input once per 32*256 clocks. 1110 = Sample interrupt input once per 64*256 clocks. 1111 = Sample interrupt input once per 128*256 clocks.

GPIO Px.n Pin Data Input/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..12,14,15**	GPIO_BA+0x880+(0x04 * n)	R/W	GPIO PC.n Pin Data Input/Output Register	0x0000_000X
PDn_PDIO **n=0,1..13,15**	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..7**	GPIO_BA+0x940+(0x04 * n)	R/W	GPIO PF.n Pin Data Input/Output Register	0x0000_000X
PGn_PDIO **n=0,1..7**	GPIO_BA+0x980+(0x04 * n)	R/W	GPIO PG.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 PB.14~15/PE.13~15/PF.13~15 pin is ignored.

6.8 Peripheral DMA Controller (PDMA)

6.8.1 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 20 channels and each channel can perform transfer between memory and peripherals or between memory and memory.

6.8.2 Features

- Supports 2 PDMA controllers, PDMA0 and PDMA1
- Supports 10 independently configurable channels
- Supports selectable 2 level of priority (fixed priority or round-robin priority)
- Supports transfer data width of 8, 16, and 32 bits
- Supports source and destination address increment size can be byte, half-word, word or no increment
- Supports software and UART, SPI 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 from channel 0 to channel 9
- Supports stride function from channel 0 to channel 5

6.8.3 Basic Configuration

6.8.3.1 Basic Configuration of PDMA0

- Clock Source Configuration
  - Enable PDMA0 controller clock in PDMA0CKEN (CLK_HCLKEN [12]).
- Reset Configuration
  - Reset PDMA0 controller in PDMA0RST (SYS_AHBIPRST[4]).

6.8.3.2 Basic Configuration of PDMA1

- Clock Source Configuration
  - Enable PDMA1 controller clock in PDMA1CKEN (CLK_HCLKEN [13]).
- Reset Configuration
  - Reset PDMA1 controller in PDMA1RST (SYS_AHBIPRST[5]).

6.8.4 Functional Description

The PDMA controller transfers data from one address to another without CPU intervention. It supports 10 independent channels and serves only one channel at one time. As the result, the 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 6.8-1 shows the diagram of descriptor table (DSCT) data structure.

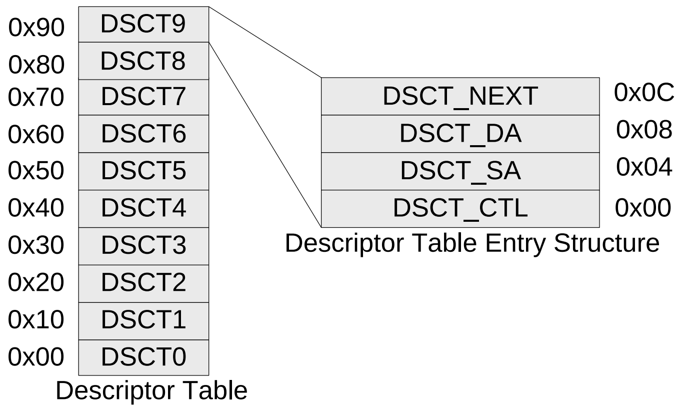


Figure 6.8-1 Channel Priority Table

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).

6.8.4.1 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 6.8-1

PDMA_PRISET	Channel Number	Priority Setting	Arbitration Priority In Descending Order
1	9	Channel9, Fixed Priority	Highest
1	8	Channel8, Fixed Priority	---
---	---	---	---
1	0	Channel0, Fixed Priority	---
0	9	Channel9, Round-Robin Priority	---
0	8	Channel8, Round-Robin Priority	---
---	---	---	---
0	0	Channel0, Round-Robin Priority	Lowest

Table 6.8-1 Descriptor Table Entry Structure

6.8.4.2 PDMA Operation Mode

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

Basic Mode

Basic mode is used to perform one descriptor table transfer mode. This mode can be used to transfer data between memory and memory or peripherals and memory. PDMA controller operation mode can be set from OPMODE (PDMA_DSCTn_CTL[1:0], n denotes PDMA channel), 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[9:0]) is enabled and the operation mode will be updated to idle state as shown in Figure 6.8-2. 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.

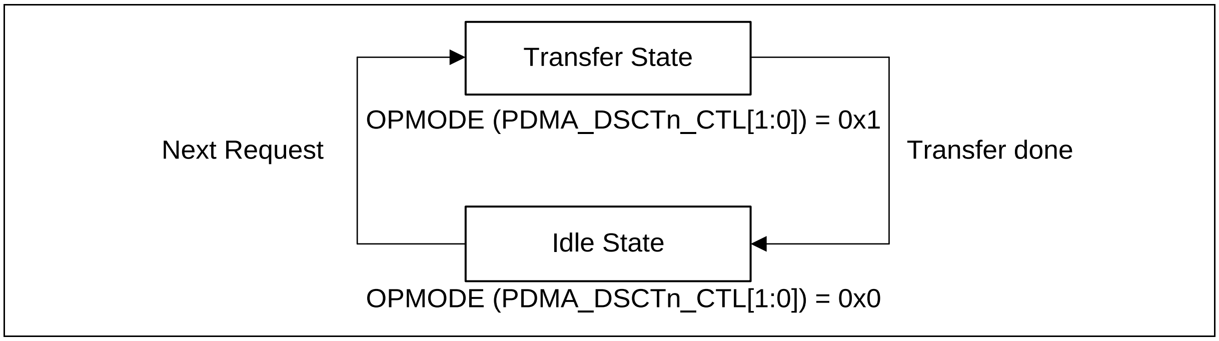

Transfer State
OPMODE (PDMA_DSCTn_CTL[1:0]) = 0x1
Next Request
Transfer done
Idle State
OPMODE (PDMA_DSCTn_CTL[1:0]) = 0x0

Figure 6.8-2 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 6.8-3. 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 only 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[9:0]) flag will be asserted and if this bit is “1” TDIFn will not be active).

If channel 9 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 PDMA_DSCTn_NEXT (link address). For example, the address in PDMA_DSCTn_NEXT is 0x2000_0100, and then the next DSCT entry start address is 0x2000_0100.

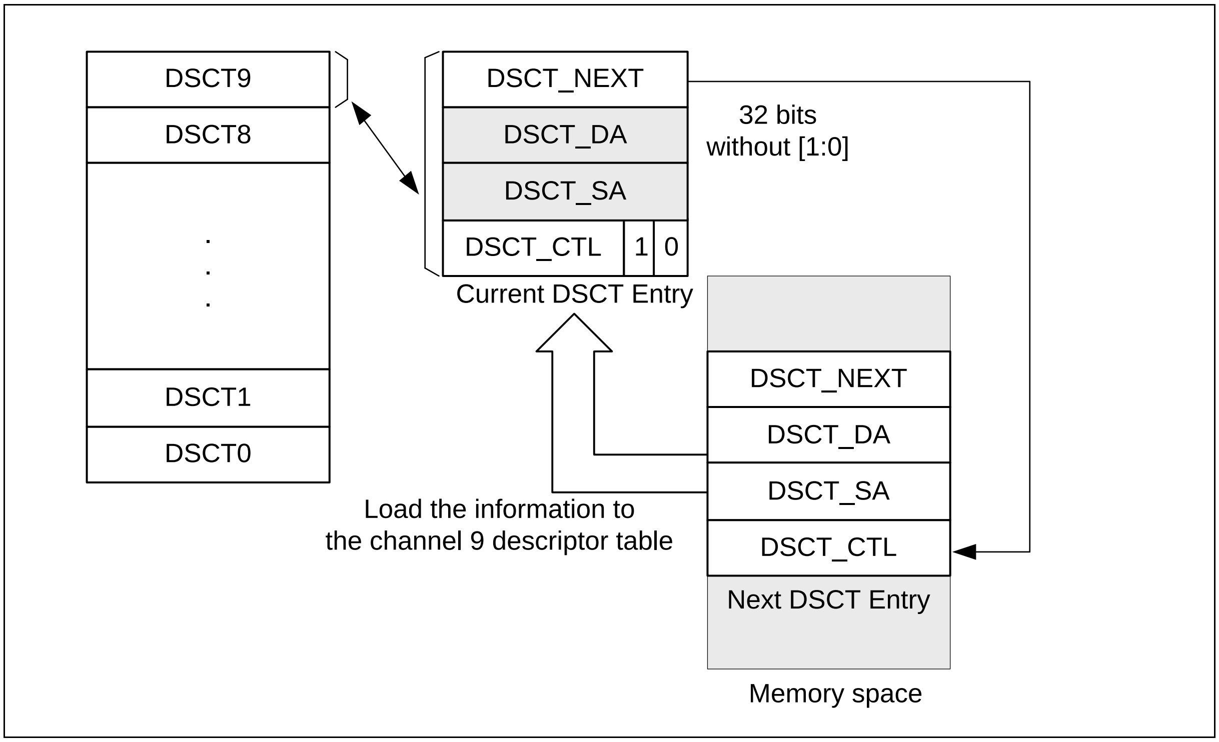


Figure 6.8-3 Descriptor Table Link List Structure

The above link list table operation is DSCT state in Scatter-Gather Mode as shown in Figure 6.7 5. When loading the information is finished, it will go to transfer state and start transfer by this information automatically. However, if the next PDMA information is also in the 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.

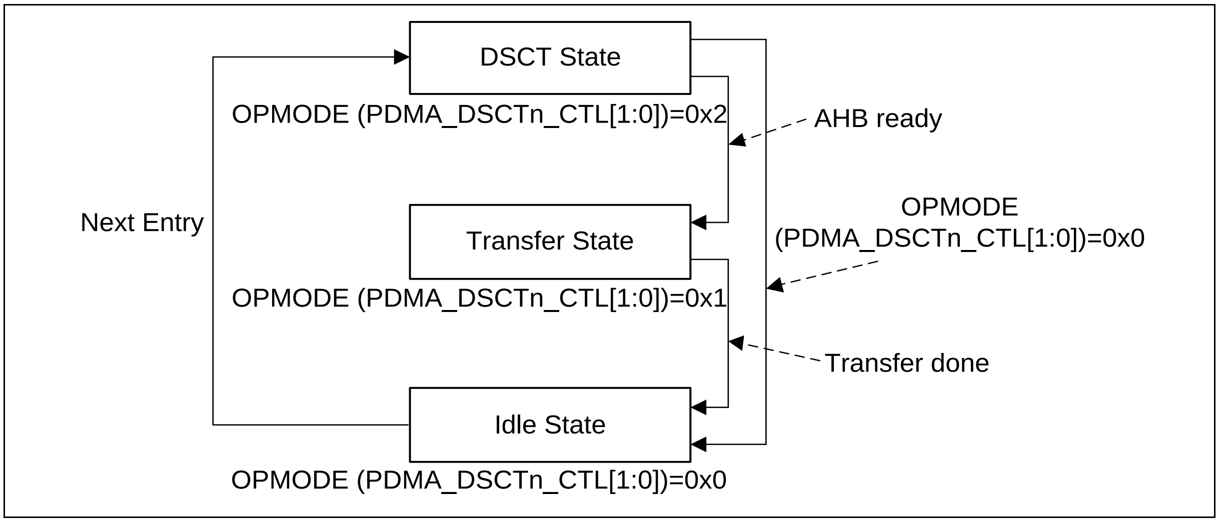

DSCT State
OPMODE (PDMA_DSCTn_CTL[1:0])=0x2
AHB ready
Next Entry
Transfer State
OPMODE
(PDMA_DSCTn_CTL[1:0])=0x0
OPMODE (PDMA_DSCTn_CTL[1:0])=0x1
Transfer done
Idle State
OPMODE (PDMA_DSCTn_CTL[1:0])=0x0

Figure 6.8-4 Scatter-Gather Mode Finite State Machine

6.8.4.3 Transfer Type

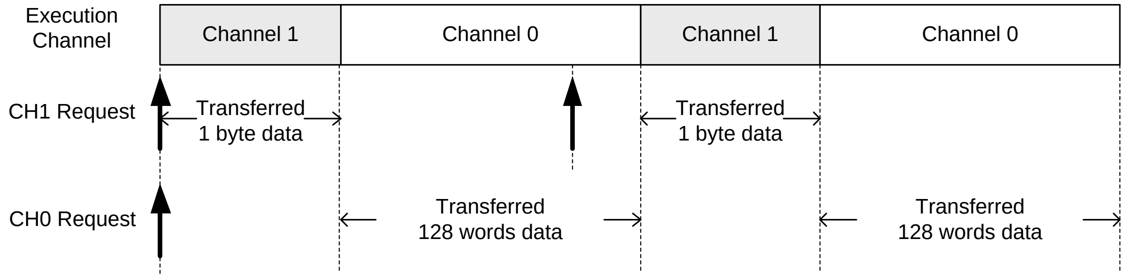
The PDMA controller supports two transfer types: single transfer type and burst transfer type, configure by setting TXTYPE (PDMA_DSCTn_CTL[2]).

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

For the burst transfer type, the PDMA controller transfers TXCNT (PDMA_DSCTn_CTL[31:16]) of data and need only one request signal. After transferred BURSIZE (PDMA_DSCTn_CTL[6:4]) of data, TXCNT (PDMA_DSCTn_CTL[31:16]) will decrease BURSIZE number. Transfer will be done after the transfer count TXCNT (PDMA_DSCTn_CTL[31:16]) decreases to 0. Note that burst transfer type can only be used for PDMA controller to do burst transfer between memory and memory. User must use single request type for memory-to-peripheral and peripheral-to-memory transfers. Please note that, PDMA transfer data between Flash and memory should finish before MCU enter idle mode or power done mode to prevent access wrong data.

Figure 6.7 6 shows an example about single and burst transfer type in basic mode. In this example, channel 1 uses single transfer type and TXCNT (PDMA_DSCTn_CTL[31:16]) = 127. Channel 0 uses burst transfer type, BURSIZE (PDMA_DSCTn_CTL[6:4]) = 128 and TXCNT (PDMA_DSCTn_CTL[31:16]) = 255. The operation sequence is described below:

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



	CH1	CH0
TXCNT (PDMA_DSCTn_CTL[31:16])	127	255
BURSIZE (PDMA_DSCTn_CTL[6:4])	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 6.8-5 Example of Single Transfer Type and Burst Transfer Type in Basic Mode

6.8.4.4 Channel Time-out

All PDMA channels support time-out function. When the transfer channel is enabled and selected to the peripheral, corresponding channel time-out TOUTENn (PDMA_TOUTEN [n], n=0~9) 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=0~9). If time-out counter counts up from 0 to corresponding channel's TOCn (PDMA_TOC0_1 [16(n+1)-1):16n], n=0~9), the PDMA controller will generate interrupt signal when corresponding TOUTIENn (PDMA_TOUTIEN [n], n=0~9) is enabled. When time-out occurred, the corresponding channel's REQTOFn (PDMA_INTSTS [n+8], n=0~9) will be set to indicate channel time-out is happened.

Time-out counter resets to 0 while counter count to TOCn (PDMA_TOC0_1 [16(n+1)-1:16n], n=0~9), received trigger signal, time-out function is disabled or chip enters Power-down mode.

Figure 6.8-6 shows an example about time-out counter operation. The operation sequence is described below:

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

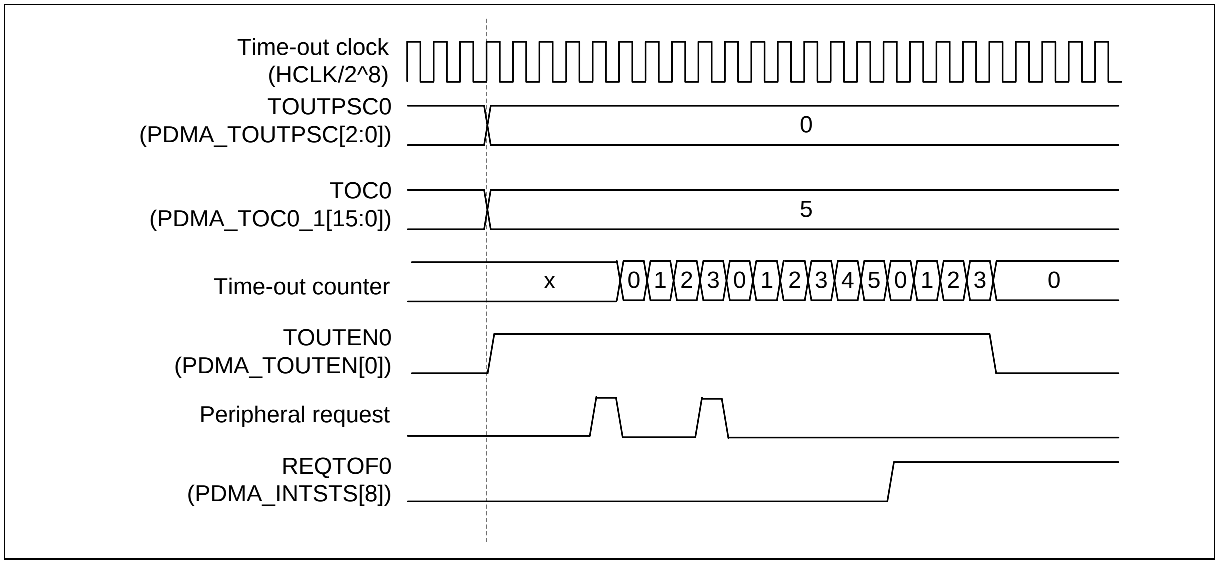


Figure 6.8-6 Example of PDMA Channel 0 Time-out Counter Operation

6.8.4.5 Stride Function

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

Set STRIDEEN (PDMA_DSCTn_CTL[15]) to enable the stride function, and then write a valid source address to the PDMA_DSCTn_SA register and a source address offset count to SASOL (PDMA_ASOCRn[15:0]) register, a destination address to the PDMA_DSCTn_DA register and a destination address offset count to DASOL (PDMA_ASOCRn[31:16]), and a transfer count to the TXCNT (PDMA_DSCTn_CTL) register and a stride transfer count to STC (PDMA_STCn[15:0]). Next, trigger the SWREQn (PDMA_SWREQ[5:0]). The PDMA will start and then stop the transfer after TXCNT (PDMA_DSCTn_CTL) counts down to 0. Figure 6.8-7 shows the block transfer relationship between source memory and destination memory. The stride function also supports peripheral to memory or memory to peripheral transfer.

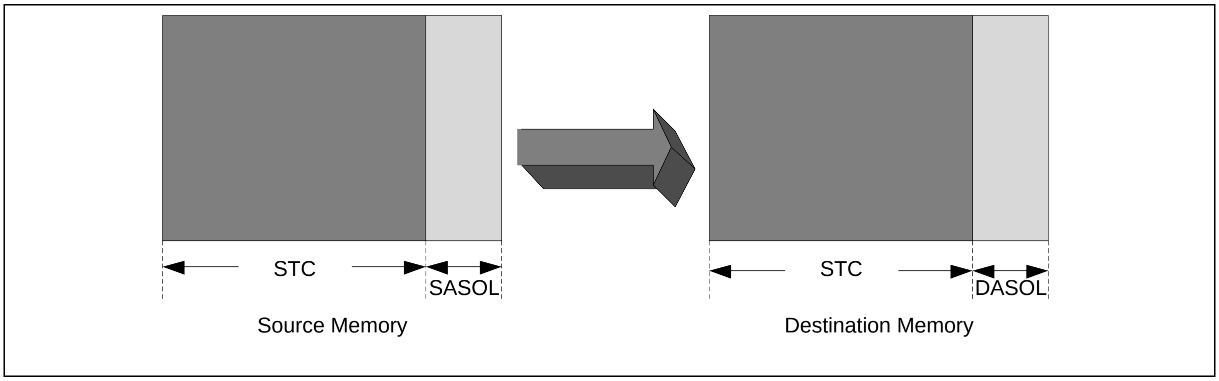


Figure 6.8-7 Stride Function Block Transfer

6.8.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
PDMA Base Address: **PDMAx_BA = 0xB000_8000 + (0x1000*x)** **x=0, 1**				
PDMAx_DSCTn_CTL **n = 0,1..9**	PDMAx_BA+0x10 * n	R/W	Descriptor Table Control Register of PDMA Channel n	0xXXXX_XXXX
PDMAx_DSCTn_SA **n = 0,1..9**	PDMAx_BA+0x0004+0x10 * n	R/W	Source Address Register of PDMA Channel n	0xXXXX_XXXX
PDMAx_DSCTn_DA **n = 0,1..9**	PDMAx_BA+0x0008+0x10 * n	R/W	Destination Address Register of PDMA Channel n	0xXXXX_XXXX
PDMAx_DSCTn_NEXT **n = 0,1..9**	PDMAx_BA+0x000C+0x10 * n	R/W	Next Scatter-gather Descriptor Table Offset Address of PDMA Channel n	0xXXXX_XXXX
PDMAx_CURSCATn **n = 0,1..9**	PDMAx_BA+0x00A0+0x004 * n	R	Current Scatter-gather Descriptor Table Address of PDMA Channel n	0xXXXX_XXXX
PDMAx_CHCTL	PDMAx_BA + 0x400	R/W	PDMA Channel Control Register	0x0000_0000
PDMAx_PAUSE	PDMAx_BA + 0x404	W	PDMA Transfer Pause Control Register	0x0000_0000
PDMAx_SWREQ	PDMAx_BA + 0x408	W	PDMA Software Request Register	0x0000_0000
PDMAx_TRGSTS	PDMAx_BA + 0x40C	R	PDMA Channel Request Status Register	0x0000_0000
PDMAx_PRISET	PDMAx_BA + 0x410	R/W	PDMA Fixed Priority Setting Register	0x0000_0000
PDMAx_PRICLR	PDMAx_BA + 0x414	W	PDMA Fixed Priority Clear Register	0x0000_0000
PDMAx_INTEN	PDMAx_BA + 0x418	R/W	PDMA Interrupt Enable Register	0x0000_0000
PDMAx_INTSTS	PDMAx_BA + 0x41C	R/W	PDMA Interrupt Status Register	0x0000_0000
PDMAx_ABTSTS	PDMAx_BA + 0x420	R/W	PDMA Channel Read/Write Target Abort Flag Register	0x0000_0000
PDMAx_TDSTS	PDMAx_BA + 0x424	R/W	PDMA Channel Transfer Done Flag Register	0x0000_0000
PDMAx_ALIGN	PDMAx_BA + 0x428	R/W	PDMA Transfer Alignment Status Register	0x0000_0000
PDMAx_TACTSTS	PDMAx_BA + 0x42C	R	PDMA Transfer Active Flag Register	0x0000_0000
PDMAx_TOUTPSC0	PDMAx_BA + 0x430	R/W	PDMA Time-out Prescaler Register 0	0x0000_0000
PDMAx_TOUTEN	PDMAx_BA + 0x434	R/W	PDMA Time-out Enable Register	0x0000_0000
PDMAx_TOUTIEN	PDMAx_BA + 0x438	R/W	PDMA Time-out Interrupt Enable Register	0x0000_0000
PDMAx_TOC0_1	PDMAx_BA + 0x440	R/W	PDMA Time-out Counter Ch1 and Ch0 Register	0x0000_0000
PDMAx_TOC2_3	PDMAx_BA + 0x444	R/W	PDMA Time-out Counter Ch3 and Ch2 Register	0x0000_0000
PDMAx_TOC4_5	PDMAx_BA + 0x448	R/W	PDMA Time-out Counter Ch5 and Ch4 Register	0x0000_0000
PDMAx_TOC6_7	PDMAx_BA + 0x44C	R/W	PDMA Time-out Counter Ch7 and Ch6 Register	0x0000_0000

PDMAx_TOC8_9	PDMAx_BA + 0x450	R/W	PDMA Time-out Counter Ch9 and Ch8 Register	0x0000_0000
PDMAx_CHRST	PDMAx_BA + 0x460	R/W	PDMA Channel Reset Register	0x0000_0000
PDMAx_TOUTPSC1	PDMAx_BA + 0x470	R/W	PDMA Time-out Prescaler Register 1	0x0000_0000
PDMAx_REQSEL0_3	PDMAx_BA + 0x480	R/W	PDMA Request Source Select Register 0	0x0000_0000
PDMAx_REQSEL4_7	PDMAx_BA + 0x484	R/W	PDMA Request Source Select Register 1	0x0000_0000
PDMAx_REQSEL8_11	PDMAx_BA + 0x488	R/W	PDMA Request Source Select Register 2	0x0000_0000
PDMAx_STCR0	PDMAx_BA + 0x500	R/W	Stride Transfer Count Register of PDMA Channel 0	0x0000_0000
PDMAx_ASOCR0	PDMAx_BA + 0x504	R/W	Address Stride Offset Register of PDMA Channel 0	0x0000_0000
PDMAx_STCR1	PDMAx_BA + 0x508	R/W	Stride Transfer Count Register of PDMA Channel 1	0x0000_0000
PDMAx_ASOCR1	PDMAx_BA + 0x50C	R/W	Address Stride Offset Register of PDMA Channel 1	0x0000_0000
PDMAx_STCR2	PDMAx_BA + 0x510	R/W	Stride Transfer Count Register of PDMA Channel 2	0x0000_0000
PDMAx_ASOCR2	PDMAx_BA + 0x514	R/W	Address Stride Offset Register of PDMA Channel 2	0x0000_0000
PDMAx_STCR3	PDMAx_BA + 0x518	R/W	Stride Transfer Count Register of PDMA Channel 3	0x0000_0000
PDMAx_ASOCR3	PDMAx_BA + 0x51C	R/W	Address Stride Offset Register of PDMA Channel 3	0x0000_0000
PDMAx_STCR4	PDMAx_BA + 0x520	R/W	Stride Transfer Count Register of PDMA Channel 4	0x0000_0000
PDMAx_ASOCR4	PDMAx_BA + 0x524	R/W	Address Stride Offset Register of PDMA Channel 4	0x0000_0000
PDMAx_STCR5	PDMAx_BA + 0x528	R/W	Stride Transfer Count Register of PDMA Channel 5	0x0000_0000
PDMAx_ASOCR5	PDMAx_BA + 0x52C	R/W	Address Stride Offset Register of PDMA Channel 5	0x0000_0000

6.8.6 Register Description

Descriptor Table Control Register (PDMAx_DSCTn_CTL)

Register	Offset	R/W	Description	Reset Value
PDMAx_DSCTn_CTL	PDMAx_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**
STRIDEEN	**Reserved**	**TXWIDTH**		**DAINC**		**SAINC**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TBINTDIS	**BURSIZE**			**Reserved**	**TXTYPE**	**OPMODE**	

Bits	Description	
[31:16]	**TXCNT**	**Transfer Count** The TXCNT represents the required number of PDMA transfer, the real transfer count is (TXCNT + 1); The maximum transfer count is 65536, every transfer may be byte, half-word or word that is dependent on TXWIDTH field. **Note:** When PDMA finish each transfer data, this field will be decrease immediately.
[15]	**STRIDEEN**	**Stride Mode Enable Bit** 0 = Stride transfer mode Disabled. 1 = Stride transfer mode Enabled.
[14]	**Reserved**	Reserved.
[13:12]	**TXWIDTH**	**Transfer Width Selection** This field is used for transfer width. 00 = One byte (8 bit) is transferred for every operation. 01 = One half-word (16 bit) is transferred for every operation. 10 = One word (32-bit) is transferred for every operation. 11 = Reserved. **Note:** The PDMA transfer source address (PDMA_DSCT_SA) and PDMA transfer destination address (PDMA_DSCT_DA) should be alignment under the TXWIDTH selection
[11:10]	**DAINC**	**Destination Address Increment** This field is used to set the destination address increment size. 11 = No increment (fixed address). Others = Increment and size is depended on TXWIDTH selection.
[9:8]	**SAINC**	**Source Address Increment** This field is used to set the source address increment size.

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

Start Source Address Register (PDMAx_DSCTn_SA)

Register	Offset	R/W	Description	Reset Value
PDMAx_DSCTn_SA	PDMAx_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 (PDMAx_DSCTn_DA)

Register	Offset	R/W	Description	Reset Value
PDMAx_DSCTn_DA	PDMAx_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 (PDMAx_DSCTn_NEXT)

Register	Offset	R/W	Description	Reset Value
PDMAx_DSCTn_NEXT	PDMAx_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
NEXT							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
NEXT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
NEXT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NEXT							

Bits	Description	
[31:0]	**NEXT**	**PDMA Next Descriptor Table Offset** This field indicates the offset of the next descriptor table address in system memory. **Write Operation:** If the next descriptor table is start from 0x2000_0100, then this field must fill in 0x2000_0100. **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 (PDMAx_CURSCATn)

Register	Offset	R/W	Description	Reset Value
PDMAx_CURSCATn	PDMAx_BA+0x00A0+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 (PDMAx_CHCTL)

Register	Offset	R/W	Description	Reset Value
PDMAx_CHCTL	PDMAx_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						CHEN9	CHEN8
7	6	5	4	3	2	1	0
CHEN7	CHEN6	CHEN5	CHEN4	CHEN3	CHEN2	CHEN1	CHEN0

Bits	Description	
[31:10]	**Reserved**	Reserved.
[n] n=0,1..9	**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 (PDMAx_PAUSE)

Register	Offset	R/W	Description	Reset Value
PDMAx_PAUSE	PDMAx_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						**PAUSE9**	**PAUSE8**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PAUSE 7	**PAUSE 6**	**PAUSE 5**	**PAUSE4**	**PAUSE3**	**PAUSE2**	**PAUSE1**	**PAUSE0**

Bits	Description	
[31:10]	**Reserved**	Reserved.
[n] n=0,1..9	**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..9) and clear request active flag. If re-enable the paused channel agian, the remaining transfers will be processed. 0 = No effect. 1 = Pause PDMA channel n transfer.

PDMA Software Request Register (PDMAx_SWREQ)

Register	Offset	R/W	Description	Reset Value
PDMAx_SWREQ	PDMAx_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						**SWREQ9**	**SWREQ8**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SWREQ7	**SWREQ6**	**SWREQ5**	**SWREQ4**	**SWREQ3**	**SWREQ2**	**SWREQ1**	**SWREQ0**

Bits	Description	
[31:10]	**Reserved**	Reserved.
[n] n=0,1..9	**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 (PDMAx_TRGSTS)

Register	Offset	R/W	Description	Reset Value
PDMAx_TRGSTS	PDMAx_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						**REQSTS9**	**REQSTS8**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
REQSTS7	**REQSTS6**	**REQSTS5**	**REQSTS4**	**REQSTS3**	**REQSTS2**	**REQSTS1**	**REQSTS0**

Bits	Description	
[31:10]	**Reserved**	Reserved.
[n] n=0,1..9	**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 (PDMAx_PRISET)

Register	Offset	R/W	Description	Reset Value
PDMAx_PRISET	PDMAx_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						**FPRISET9**	**FPRISET8**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FPRISET7	**FPRISET6**	**FPRISET5**	**FPRISET4**	**FPRISET3**	**FPRISET2**	**FPRISET1**	**FPRISET0**

Bits	Description	
[31:10]	**Reserved**	Reserved.
[n] n=0,1..9	**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 (PDMAx_PRICLR)

Register	Offset	R/W	Description	Reset Value
PDMAx_PRICLR	PDMAx_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						**FPRICLR9**	**FPRICLR8**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FPRICLR7	**FPRICLR6**	**FPRICLR5**	**FPRICLR4**	**FPRICLR3**	**FPRICLR2**	**FPRICLR1**	**FPRICLR0**

Bits	Description	
[31:10]	**Reserved**	Reserved.
[n] n=0,1..9	**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 (PDMAx_INTEN)

Register	Offset	R/W	Description	Reset Value
PDMAx_INTEN	PDMAx_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						**INTEN9**	**INTEN8**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
INTEN7	**INTEN6**	**INTEN5**	**INTEN4**	**INTEN3**	**INTEN2**	**INTEN1**	**INTEN0**

Bits	Description	
[31:10]	**Reserved**	Reserved.
[n] n=0,1..9	**INTENn**	**PDMA Interrupt Enable Its** This field is used to enable PDMA channel[n] interrupt. 0 = PDMA channel n interrupt Disabled. 1 = PDMA channel n interrupt Enabled.

PDMA Interrupt Status Register (PDMAx_INTSTS)

Register	Offset	R/W	Description	Reset Value
PDMAx_INTSTS	PDMAx_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 this bit. 0 = No request time-out. 1 = Peripheral request time-out.
[8]	**REQTOF0**	**Request Time-out Flag for Channel 0** This flag indicates that PDMA controller has waited peripheral request for a period defined by PDMA_TOC0, user can write 1 to clear this bit. 0 = No request time-out. 1 = Peripheral request time-out.
[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 (PDMAx_ABTSTS)

Register	Offset	R/W	Description	Reset Value
PDMAx_ABTSTS	PDMAx_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						ABTIF9	ABTIF8
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ABTIF7	ABTIF6	ABTIF5	ABTIF4	ABTIF3	ABTIF2	ABTIF1	ABTIF0

Bits	Description	
[31:10]	**Reserved**	Reserved.
[n] n=0,1..9	**ABTIFn**	**PDMA Read/Write Target Abort Interrupt Status Flag** This bit indicates which PDMA controller has target abort error; User can write 1 to clear these bits. 0 = No AHB bus ERROR response received when channel n transfer. 1 = AHB bus ERROR response received when channel n transfer. **Note:** If channel x target abort, REQSRCx should set 0 to disable peripheral request.

PDMA Channel Transfer Done Flag Register (PDMAx_TDSTS)

Register	Offset	R/W	Description	Reset Value
PDMAx_TDSTS	PDMAx_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						**TDIF9**	**TDIF8**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TDIF7	**TDIF6**	**TDIF5**	**TDIF4**	**TDIF3**	**TDIF2**	**TDIF1**	**TDIF0**

Bits	Description	
[31:10]	**Reserved**	Reserved.
[n] n=0,1..9	**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 (PDMAx_ALIGN)

Register	Offset	R/W	Description	Reset Value
PDMAx_ALIGN	PDMAx_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						**ALIGN9**	**ALIGN8**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ALIGN7	**ALIGN6**	**ALIGN5**	**ALIGN4**	**ALIGN3**	**ALIGN2**	**ALIGN1**	**ALIGN0**

Bits	Description	
[31:10]	**Reserved**	Reserved.
[n] n=0,1..9	**ALIGNn**	**Transfer Alignment Flag** 0 = PDMA channel source address and destination address both follow transfer width setting. 1 = PDMA channel source address or destination address is not follow transfer width setting.

PDMA Transfer Active Flag Register (PDMAx_TACTSTS)

Register	Offset	R/W	Description	Reset Value
PDMAx_TACTSTS	PDMAx_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						**TXACTF9**	**TXACTF8**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXACTF7	**TXACTF6**	**TXACTF5**	**TXACTF4**	**TXACTF3**	**TXACTF2**	**TXACTF1**	**TXACTF0**

Bits	Description	
[31:10]	**Reserved**	Reserved.
[n] n=0,1..9	**TXACTFn**	**Transfer on Active Flag (Read Only)** This bit indicates which PDMA channel is in active. 0 = PDMA channel is not finished. 1 = PDMA channel is active.

PDMA Time-out Prescaler Register0 (PDMAx_TOUTPSC0)

Register	Offset	R/W	Description	Reset Value
PDMAx_TOUTPSC0	PDMAx_BA + 0x430	R/W	PDMA Time-out Prescaler Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	TOUTPSC7			Reserved	TOUTPSC6		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	TOUTPSC5			Reserved	TOUTPSC4		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	TOUTPSC3			Reserved	TOUTPSC2		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	TOUTPSC1			Reserved	TOUTPSC0		

Bits	Description	
[31]	**Reserved**	Reserved.
[30:28]	**TOUTPSC7**	**PDMA Channel 7 Time-out Clock Source Prescaler Bits** The configuration is the same as TOUTPSC0 field. Please refer to the explanation of TOUTPSC0.
[27]	**Reserved**	Reserved.
[26:24]	**TOUTPSC6**	**PDMA Channel 6 Time-out Clock Source Prescaler Bits** The configuration is the same as TOUTPSC0 field. Please refer to the explanation of TOUTPSC0.
[23]	**Reserved**	Reserved.
[22:20]	**TOUTPSC5**	**PDMA Channel 5 Time-out Clock Source Prescaler Bits** The configuration is the same as TOUTPSC0 field. Please refer to the explanation of TOUTPSC0.
[19]	**Reserved**	Reserved.
[18:16]	**TOUTPSC4**	**PDMA Channel 4 Time-out Clock Source Prescaler Bits** The configuration is the same as TOUTPSC0 field. Please refer to the explanation of TOUTPSC0.
[15]	**Reserved**	Reserved.
[14:12]	**TOUTPSC3**	**PDMA Channel 3 Time-out Clock Source Prescaler Bits** The configuration is the same as TOUTPSC0 field. Please refer to the explanation of TOUTPSC0.
[1]	**Reserved**	Reserved.
[10:8]	**TOUTPSC2**	**PDMA Channel 2 Time-out Clock Source Prescaler Bits** The configuration is the same as TOUTPSC0 field. Please refer to the explanation of TOUTPSC0.

Bits	Description	
[7]	**Reserved**	Reserved.
[6:4]	**TOUTPSC1**	**PDMA Channel 1 Time-out Clock Source Prescaler Bits** The configuration is the same as TOUTPSC0 field. Please refer to the explanation of TOUTPSC0.
[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 (PDMAx_TOUTEN)

Register	Offset	R/W	Description	Reset Value
PDMAx_TOUTEN	PDMAx_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						**TOUTEN9**	**TOUTEN8**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TOUTEN7	**TOUTEN6**	**TOUTEN5**	**TOUTEN4**	**TOUTEN3**	**TOUTEN2**	**TOUTEN1**	**TOUTEN0**

Bits	Description	
[31:10]	**Reserved**	Reserved.
[n] n=0,1..9	**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 (PDMAx_TOUTIEN)

Register	Offset	R/W	Description	Reset Value
PDMAx_TOUTIEN	PDMAx_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						TOUTIEN9	TOUTIEN8
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TOUTIEN7	TOUTIEN6	TOUTIEN5	TOUTIEN4	TOUTIEN3	TOUTIEN2	TOUTIEN1	TOUTIEN0

Bits	Description	
[31:10]	**Reserved**	Reserved.
[n] n=0,1..9	**TOUTIENn**	**PDMA Time-out Interrupt Enable Bits** 0 = PDMA Channel n time-out interrupt Disabled. 1 = PDMA Channel n time-out interrupt Enabled.

PDMA Time-out Period Counter Register 0 (PDMAx_TOC0_1)

Register	Offset	R/W	Description	Reset Value
PDMAx_TOC0_1	PDMAx_BA + 0x440	R/W	PDMA Time-out Counter Ch1 and Ch0 Register	0x0000_0000

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

Bits	Description	
[31:16]	**TOC1**	**Time-out Counter for Channel 1** This controls the period of time-out function for channel 1. The calculation unit is based on TOUTPSC1 (PDMA_TOUTPSC0[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_TOUTPSC0[2:0]) clock. Time-out period = (Period of time-out clock) * (16-bit TOCn), n = 0,1.

PDMA Time-out Period Counter Register 1 (PDMAx_TOC2_3)

Register	Offset	R/W	Description	Reset Value
PDMAx_TOC2_3	PDMAx_BA + 0x444	R/W	PDMA Time-out Counter Ch3 and Ch2 Register	0x0000_0000

31	30	29	28	27	26	25	24
TOC3							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TOC3							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TOC2							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TOC2							

Bits	Description	
[31:16]	**TOC3**	**Time-out Counter for Channel 3** This controls the period of time-out function for channel 3. The calculation unit is based on TOUTPSC3 (PDMA_TOUTPSC0[14:12]) clock. The example of time-out period can refer TOC0 bit description.
[15:0]	**TOC2**	**Time-out Counter for Channel 2** This controls the period of time-out function for channel 2. The calculation unit is based on TOUTPSC2 (PDMA_TOUTPSC0[10:8]) clock. Time-out period = (Period of time-out clock) * (16-bit TOCn), n = 2,3.

PDMA Time-out Period Counter Register 2 (PDMAx_TOC4_5)

Register	Offset	R/W	Description	Reset Value
PDMAx_TOC4_5	PDMAx_BA + 0x448	R/W	PDMA Time-out Counter Ch5 and Ch4 Register	0x0000_0000

31	30	29	28	27	26	25	24
TOC5							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TOC5							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TOC4							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TOC4							

Bits	Description	
[31:16]	**TOC5**	**Time-out Counter for Channel 5** This controls the period of time-out function for channel 5. The calculation unit is based on TOUTPSC5 (PDMA_TOUTPSC0[22:20]) clock. The example of time-out period can refer TOC0 bit description.
[15:0]	**TOC4**	**Time-out Counter for Channel 4** This controls the period of time-out function for channel 4. The calculation unit is based on TOUTPSC4 (PDMA_TOUTPSC0[18:16]) clock. Time-out period = (Period of time-out clock) * (16-bit TOCn), n = 4,5.

PDMA Time-out Period Counter Register 3 (PDMAx_TOC6_7)

Register	Offset	R/W	Description	Reset Value
PDMAx_TOC6_7	PDMAx_BA + 0x44C	R/W	PDMA Time-out Counter Ch7 and Ch6 Register	0x0000_0000

31	30	29	28	27	26	25	24
TOC7							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TOC7							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TOC6							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TOC6							

Bits	Description	
[31:16]	**TOC7**	**Time-out Counter for Channel 7** This controls the period of time-out function for channel 7. The calculation unit is based on TOUTPSC7 (PDMA_TOUTPSC0[30:28]) clock. The example of time-out period can refer TOC0 bit description.
[15:0]	**TOC6**	**Time-out Counter for Channel 6** This controls the period of time-out function for channel 6. The calculation unit is based on TOUTPSC6 (PDMA_TOUTPSC0[26:24]) clock. Time-out period = (Period of time-out clock) * (16-bit TOCn), n = 6,7.

PDMA Time-out Period Counter Register 4 (PDMAx_TOC8_9)

Register	Offset	R/W	Description	Reset Value
PDMAx_TOC8_9	PDMAx_BA + 0x450	R/W	PDMA Time-out Counter Ch9 and Ch8 Register	0x0000_0000

31	30	29	28	27	26	25	24
TOC9							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TOC9							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
TOC8							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TOC8							

Bits	Description	
[31:16]	**TOC9**	**Time-out Counter for Channel 9** This controls the period of time-out function for channel 9. The calculation unit is based on TOUTPSC9 (PDMA_TOUTPSC1[5:3]) clock. The example of time-out period can refer TOC0 bit description.
[15:0]	**TOC8**	**Time-out Counter for Channel 8** This controls the period of time-out function for channel 8. The calculation unit is based on TOUTPSC8 (PDMA_TOUTPSC1[2:0]) clock. Time-out period = (Period of time-out clock) * (16-bit TOCn), n = 8,9.

PDMA Channel Reset Register (PDMAx_CHRST)

Register	Offset	R/W	Description	Reset Value
PDMAx_CHRST	PDMAx_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						**CH9RST**	**CH8RST**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CH7RST	**CH6RST**	**CH5RST**	**CH4RST**	**CH3RST**	**CH2RST**	**CH1RST**	**CH0RST**

Bits	Description	
[31:10]	**Reserved**	Reserved.
[n] n=0,1..9	**CHnRST**	**Channel n Reset** 0 = corresponding channel n is not reset. 1 = corresponding channel n is reset.

PDMA Time-out Prescaler Register1 (PDMAx_TOUTPSC1)

Register	Offset	R/W	Description	Reset Value
PDMAx_TOUTPSC1	PDMAx_BA + 0x470	R/W	PDMA Time-out Prescaler 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	TOUTPSC9			Reserved	TOUTPSC8		

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6:4]	**TOUTPSC9**	**PDMA Channel 9 Time-out Clock Source Prescaler Bits** The configuration is the same as TOUTPSC8 field. Please refer to the explanation of TOUTPSC8.
[3]	**Reserved**	Reserved.
[2:0]	**TOUTPSC8**	**PDMA Channel 8 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}.

NUC980 SERIES TECHNICAL REFERENCE MANUAL

PDMA Request Source Select Register 0 (PDMAx_REQSEL0_3)

Register	Offset	R/W	Description	Reset Value
PDMAx_REQSEL0_3	PDMAx_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]	**Reserved**	Reserved.
[30:24]	**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]	**Reserved**	Reserved.
[22:16]	**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]	**Reserved**	Reserved.
[14:8]	**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]	**Reserved**	Reserved.
[6:0]	**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 = Reserved. 1 = Reserved. 2 = Reserved. 3 = Reserved.

Bits	Description	
		4 = UART0_TX. 5 = UART0_RX. 6 = UART1_TX. 7 = UART1_RX. 8 = UART2_TX. 9 = UART2_RX. 10 = UART3_TX. 11 = UART3_RX. 12 = UART4_TX. 13 = UART4_RX. 14 = UART5_TX. 15 = UART5_RX. 16 = UART6_TX. 17 = UART6_RX. 18 = UART7_TX. 19 = UART7_RX. 20 = SPI0_TX. 21 = SPI0_RX. 22 = SPI1_TX. 23 = SPI1_RX. 24 = SPI2_TX. 25 = SPI2_RX. 26 = UART8_TX. 27 = UART8_RX. 28 = UART9_TX. 29 = UART9_RX. 30 = Reserved. 31 = Reserved. 32 = Reserved. 33 = Reserved. 34 = Reserved. 35 = Reserved. 36 = Reserved. 37 = Reserved. 38 = Reserved. 39 = Reserved. 40 = Reserved. 41 = Reserved. 42 = Reserved. 43 = Reserved. 44 = Reserved. 45 = Reserved. 46 = TIMER0. 47 = TIMER1. 48 = TIMER2. 49 = TIMER3.

Bits	Description	
		50 = TIMER4. 51 = TIMER5. 52 = Reserved. 53 = Reserved. 54 = Reserved. 55 = Reserved. 56 = Reserved. 57 = Reserved. 58 = Reserved. 59 = Reserved. 60 = Reserved. 61 = Reserved. 62 = Reserved. 63 = Reserved. 64 = Reserved. 65 = Reserved. 66 = Reserved. 67 = Reserved. 68 = Reserved. 69 = Reserved. Others = Reserved. **Note 1:** A peripheral cannot be assigned to two channels at the same time. **Note 2:** This field is useless when transfer between memory and memory.

PDMA Request Source Select Register 1 (PDMAx_REQSEL4_7)

Register	Offset	R/W	Description	Reset Value
PDMAx_REQSEL4_7	PDMAx_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]	**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]	**Reserved**	Reserved.
[22:16]	**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]	**Reserved**	Reserved.
[14:8]	**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]	**Reserved**	Reserved.
[6:0]	**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 (PDMAx_REQSEL8_11)

Register	Offset	R/W	Description	Reset Value
PDMAx_REQSEL8_11	PDMAx_BA + 0x488	R/W	PDMA Request Source Select Register 2	0x0000_0000

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

Bits	Description	
[31]	**Reserved**	Reserved.
[29:24]	**REQSRC11**	**Channel 11 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 11. User can configure the peripheral setting by REQSRC11. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[23]	**Reserved**	Reserved.
[22:16]	**REQSRC10**	**Channel 10 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 10. User can configure the peripheral setting by REQSRC10. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[15]	**Reserved**	Reserved.
[14:8]	**REQSRC9**	**Channel 9 Request Source Selection** This filed defines which peripheral is connected to PDMA channel 9. User can configure the peripheral setting by REQSRC9. **Note:** The channel configuration is the same as REQSRC0 field. Please refer to the explanation of REQSRC0.
[7]	**Reserved**	Reserved.
[6:0]	**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.

PDMA Stride Transfer Count Register n (PDMAx_STCRn)

Register	Offset	R/W	Description	Reset Value
PDMAx_STCR0	PDMAx_BA + 0x500	R/W	Stride Transfer Count Register of PDMA Channel 0	0x0000_0000
PDMAx_STCR1	PDMAx_BA + 0x508	R/W	Stride Transfer Count Register of PDMA Channel 1	0x0000_0000
PDMAx_STCR2	PDMAx_BA + 0x510	R/W	Stride Transfer Count Register of PDMA Channel 2	0x0000_0000
PDMAx_STCR3	PDMAx_BA + 0x518	R/W	Stride Transfer Count Register of PDMA Channel 3	0x0000_0000
PDMAx_STCR4	PDMAx_BA + 0x520	R/W	Stride Transfer Count Register of PDMA Channel 4	0x0000_0000
PDMAx_STCR5	PDMAx_BA + 0x528	R/W	Stride Transfer Count Register of PDMA Channel 5	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**STC**	**PDMA Stride Transfer Count** The 16-bit register defines the stride transfer count of each row.

NUC980 SERIES TECHNICAL REFERENCE MANUAL

PDMA Address Stride Offset Control Register n (PDMAx_ASOCRn)

Register	Offset	R/W	Description	Reset Value
PDMAx_ASOCR0	PDMAx_BA + 0x504	R/W	Address Stride Offset Register of PDMA Channel 0	0x0000_0000
PDMAx_ASOCR1	PDMAx_BA + 0x50C	R/W	Address Stride Offset Register of PDMA Channel 1	0x0000_0000
PDMAx_ASOCR2	PDMAx_BA + 0x514	R/W	Address Stride Offset Register of PDMA Channel 2	0x0000_0000
PDMAx_ASOCR3	PDMAx_BA + 0x51C	R/W	Address Stride Offset Register of PDMA Channel 3	0x0000_0000
PDMAx_ASOCR4	PDMAx_BA + 0x524	R/W	Address Stride Offset Register of PDMA Channel 4	0x0000_0000
PDMAx_ASOCR5	PDMAx_BA + 0x52C	R/W	Address Stride Offset Register of PDMA Channel 5	0x0000_0000

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

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

6.9 Timer Controller (TMR)

6.9.1 Overview

The timer controller includes six 32-bit timers, Timer0 ~ Timer5, allowing user to easily implement a timer control 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.

6.9.2 Features

- Six sets of 32-bit timers with 24-bit up counter and one 8-bit prescale counter
- Independent Clock Source for each Timer
- Provides one-shot, periodic, toggle-output and continuous counting operation modes
- 24-bit up counter value is readable through CNT (TIMERx_CNT[23:0])
- Supports event counting function to count input event from pin TMx_ECNT (x = 0~5)
- Supports toggle output to pin TMx_TGL (x = 0~5)
- 24-bit capture value is readable through CAPDAT (TIMERx_CAP[23:0])
- Supports event capture from external pin TMx_EXT (x = 0~5) for interval measurement
- Supports event capture from RTC 1Hz signal for RTC clock calibration
- Supports event capture from external pin TMx_EXT (x = 0~5) to reset 24-bit up counter
- Supports chip wake-up from Idle/Power-down mode if a timer interrupt signal is generated
- Supports time-out interrupt or capture interrupt to trigger ADC and PDMA.
- Supports Inter-Timer trigger that Timer 0 can trigger Timer 1, Timer 2 can trigger Timer 3, and Timer4 can trigger Timer5.

6.9.3 Block Diagram

Each timer is equipped with an 8-bit pre-scale counter, a 24-bit up-counter, a 24-bit compare register and an interrupt request signal. Refer to Figure 6.9-1 and Figure 6.9-2 for the timer controller block diagram. There are six options of clock sources for each timer, illustrate the Clock Source control function.

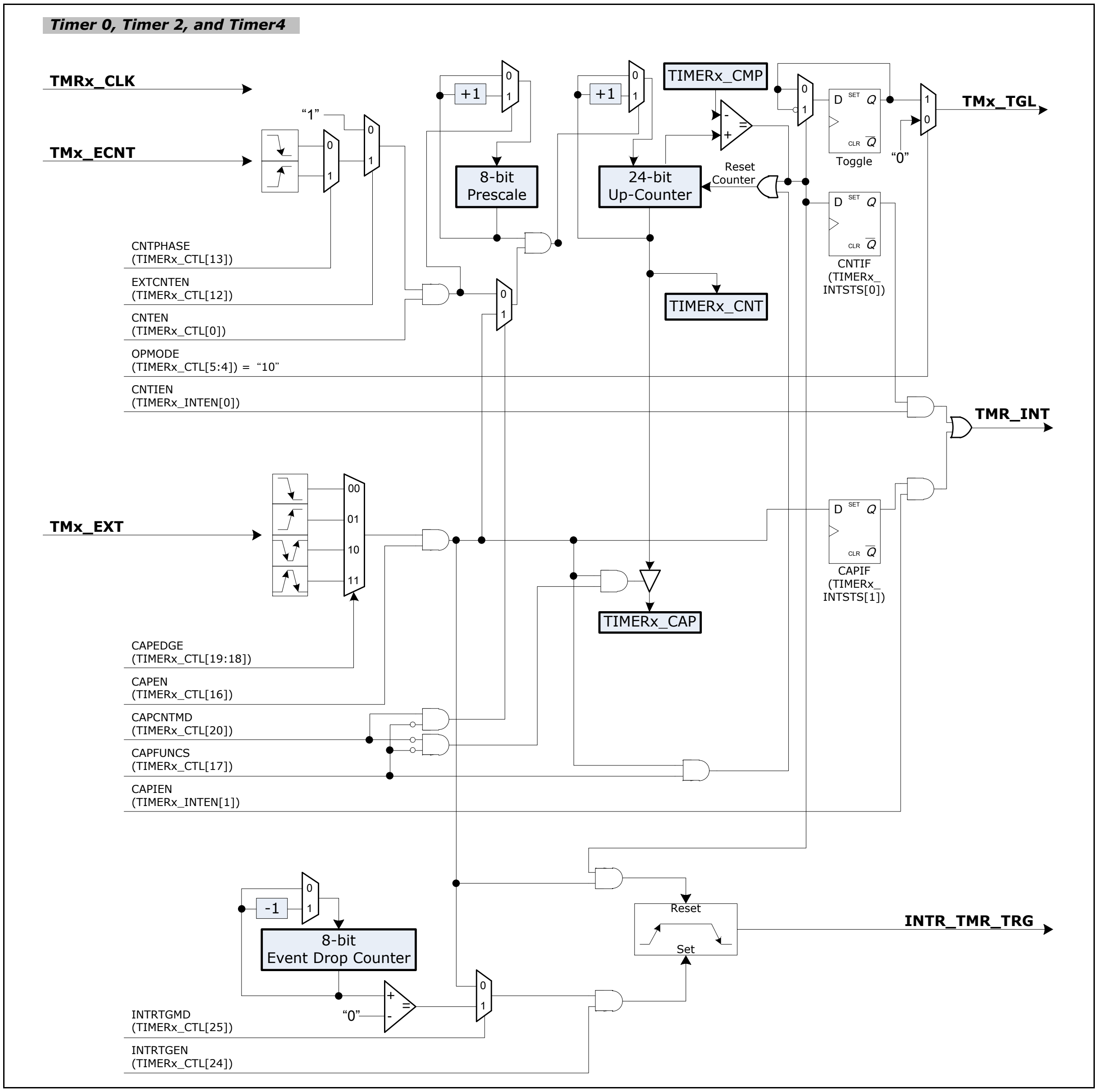


Figure 6.9-1 Timer 0, Timer 2, and Timer 4 Controller Block Diagram

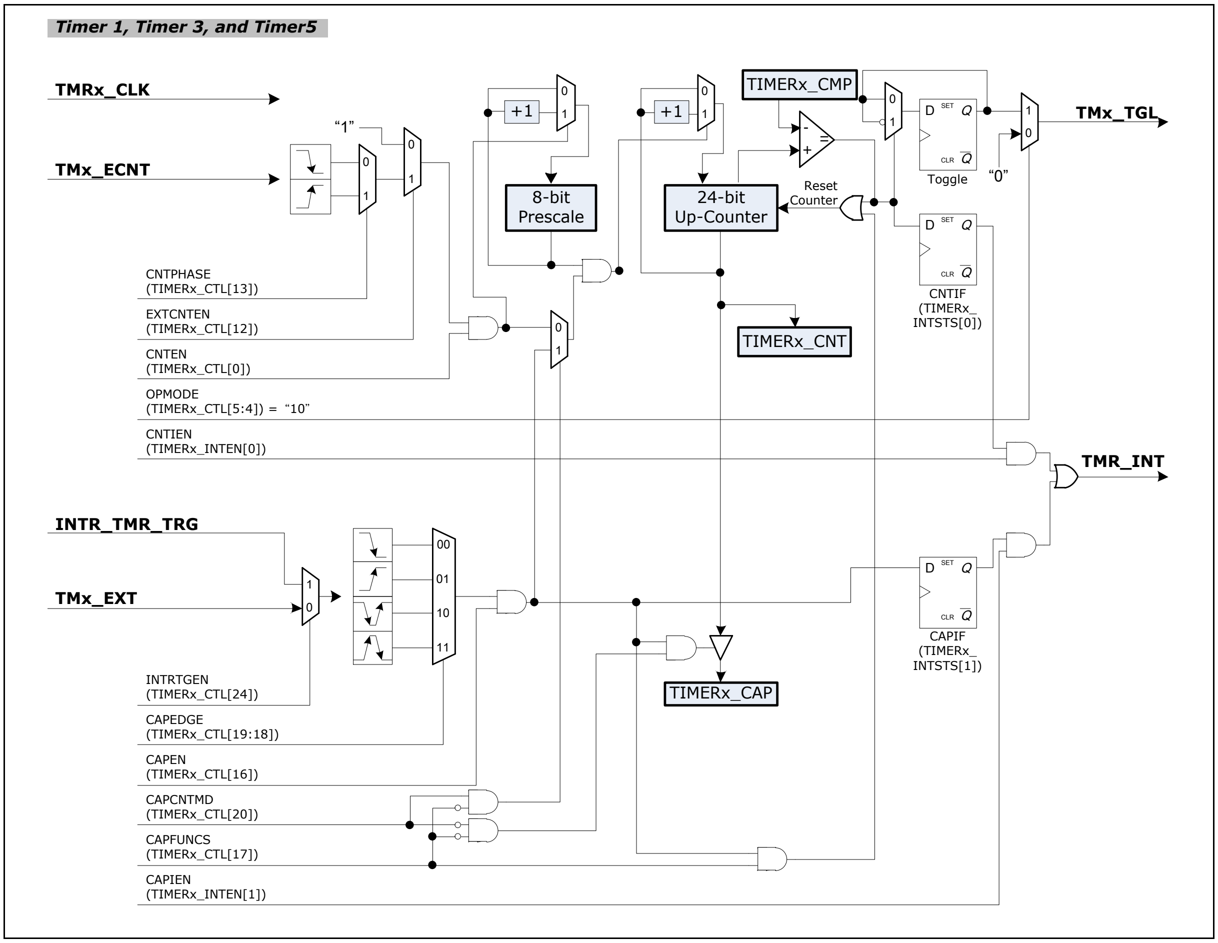
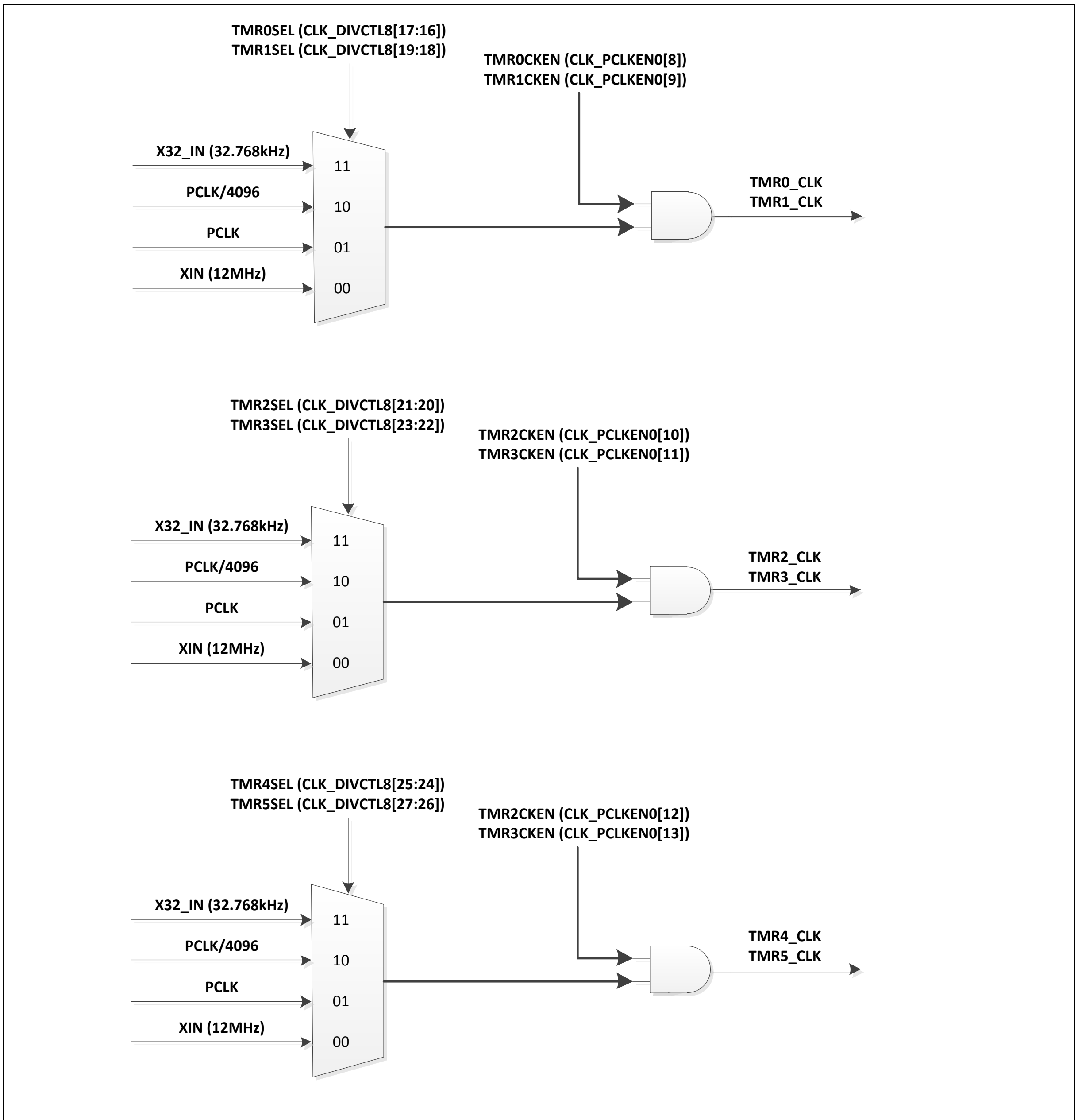


Figure 6.9-2 Timer 1, Timer 3, and Timer 5 Controller Block Diagram


TMR0SEL (CLK_DIVCTL8[17:16])
TMR1SEL (CLK_DIVCTL8[19:18])
TMR0CKEN (CLK_PCLKEN0[8])
TMR1CKEN (CLK_PCLKEN0[9])
X32_IN (32.768kHz)
PCLK/4096
PCLK
XIN (12MHz)
11
10
01
00
TMR0_CLK
TMR1_CLK
TMR2SEL (CLK_DIVCTL8[21:20])
TMR3SEL (CLK_DIVCTL8[23:22])
TMR2CKEN (CLK_PCLKEN0[10])
TMR3CKEN (CLK_PCLKEN0[11])
X32_IN (32.768kHz)
PCLK/4096
PCLK
XIN (12MHz)
11
10
01
00
TMR2_CLK
TMR3_CLK
TMR4SEL (CLK_DIVCTL8[25:24])
TMR5SEL (CLK_DIVCTL8[27:26])
TMR2CKEN (CLK_PCLKEN0[12])
TMR3CKEN (CLK_PCLKEN0[13])
X32_IN (32.768kHz)
PCLK/4096
PCLK
XIN (12MHz)
11
10
01
00
TMR4_CLK
TMR5_CLK

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

Figure 6.9-3 Timer Clock Controller Diagram

6.9.4 Basic Configuration

6.9.4.1 TIMER01 Basic Configurations

- Clock source configuration
  - Enable TIMER0 peripheral clock in TMR0CKEN (CLK_PCLKEN0[8]).
  - Enable TIMER1 peripheral clock in TMR1CKEN (CLK_PCLKEN0[9]).
- Reset Configuration
  - Reset TIMER0 controller in TIMER0 (SYS_APBIPRST0[8]).
  - Reset TIMER1 controller in TIMER1 (SYS_APBIPRST0[9]).
- Pin configuration

Group	Pin Name	GPIO	MFP
TM0	TM0_ECNT	PD. 6, PF. 0	MFP3
		PA. 0	MFP6
	TM0_EXT	PB. 1	MFP5
		PB. 8	MFP7
		PB. 10	MFP3
	TM0_TGL	PB.9	MFP3
		PB.3	MFP5
		PC.0	MFP7
TM1	TM1_ECNT	PD. 7, PF. 1	MFP3
		PA. 1	MFP6
	TM1_EXT	PA. 13, PD.1, PG. 12, PF. 9	MFP3
	TM1_TGL	PA.14, PD.0, PF.8, PG.11	MFP3

Table 6.9-1 TIMER01 Pin Configuration

6.9.4.2 TIMER23 Basic Configurations

- Clock source configuration
  - Enable TIMER2 peripheral clock in TMR2CKEN (CLK_PCLKEN0[10]).
  - Enable TIMER3 peripheral clock in TMR3CKEN (CLK_PCLKEN0[11]).
- Reset Configuration
  - Reset TIMER2 controller in TIMER2(SYS_APBIPRST0[10]).
  - Reset TIMER3 controller in TIMER3 (SYS_APBIPRST0[11]).
- Pin configuration

Group	Pin Name	GPIO	MFP
TM2	TM2_ECNT	PD. 8, PF. 2	MFP3
		PA. 2	MFP6

	TM2_EXT	PA. 9, PB. 11	MFP3
		PD. 13	MFP2
	TM2_TGL	PD.12	MFP2
		PA.10, PB.12	MFP3
TM3	TM3_ECNT	PD. 9, PF. 3	MFP3
		PA. 3	MFP6
	TM3_EXT	PA. 7	MFP3
		PD. 15	MFP2
	TM3_TGL	PD.14	MFP2
		PA.8	MFP3

6.9.4.3 TIMER45 Basic Configurations

- Clock source configuration
  - Enable TIMER4 peripheral clock in TMR2CKEN (CLK_PCLKEN0[12]).
  - Enable TIMER5 peripheral clock in TMR3CKEN (CLK_PCLKEN0[13]).
- Reset Configuration
  - Reset TIMER4 controller in TIMER4 (SYS_APBIPRST0[12]).
  - Reset TIMER5 controller in TIMER5 (SYS_APBIPRST0[13]).
- Pin configuration

Group	Pin Name	GPIO	MFP
TM4	TM4_ECNT	PD. 10, PF. 4	MFP3
		PA. 4	MFP6
	TM4_EXT	PA. 11, PD. 2, PF. 6	MFP3
	TM4_TGL	PA.12, PB.13, PD.3	MFP3
TM5	TM5_ECNT	PD. 11, PF. 5	MFP3
		PA. 5	MFP6
	TM5_EXT	PA. 15, PD.4, PF. 7	MFP3
	TM5_TGL	PD.5, PF.10, PG.10	MFP3

6.9.5 Functional Description

The timer controller provides four timer counting modes: One-shot, Periodic, Toggle-output and Continuous Counting operation modes. In addition, the timer controller provides event counting function to count events from external pin, external pin capture function to measure interval or reset timer counter, and Inter-Timer Trigger function to measure input frequency precisely.

6.9.5.1 One-Shot Mode

If the timer controller is configured in One-shot mode (OPMODE (TIMERx_CTL[5:4]) is 00) and CNTEN (TIMERx_CTL[0]) is set to 1, the timer counter starts up counting. Once the CNT

(TIMERx_CNT[23:0]) value reaches CMPDAT (TIMERx_CMP[23:0]) value, the CNTIF (TIMERx_INTSTS[0]) will be set to 1, CNT (TIMERx_CNT) value and CNTEN (TIMERx_CTL[0]) bit are cleared to 0 automatically by timer controller then timer counting operation stops. In the meantime, if the CNTIEN (TIMERx_INTEN[0]) is enabled, the timer interrupt signal is generated and sent to NVIC to inform CPU also.

6.9.5.2 Periodic Mode

If the timer controller is configured in Periodic mode (OPMODE (TIMERx_CTL[5:4]) is 01) and CNTEN (TIMERx_CTL[0]) is set to 1, the timer counter starts up counting. Once the CNT (TIMERx_CNT[23:0]) value reaches CMPDAT (TIMERx_CMP[23:0]) value, the CNTIF (TIMERx_INTSTS[0]) will be set to 1, CNT (TIMERx_CNT[23:0]) value will be cleared automatically by timer controller and timer counter operates counting again. In the meantime, if the CNTIEN (TIMERx_INTEN[0]) bit is enabled, the timer interrupt signal is generated and sent to NVIC to inform CPU also. In this mode, timer controller operates counting and compares with CMPDAT (TIMERx_CMP[23:0])value periodically until the CNTEN (TIMERx_CTL[0])bit is cleared to 0 by user.

6.9.5.3 Toggle-Output Mode

If the timer controller is configured in Toggle-output mode (OPMODE (TIMERx_CTL[5:4]) is 10) and CNTEN (TIMERx_CTL[0]) is set to 1, 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_TGL ~ TM5_TGL to output signal while specified CNTIF (TIMERx_INTSTS[0]) is set to 1. Thus, the toggle-output signal on TM0_TGL ~ TM5_TGL pin is high and changing back and forth with 50% duty cycle.

6.9.5.4 Continuous Counting Mode

If the timer controller is configured in Continuous Counting mode (OPMODE (TIMERx_CTL[5:4]) is 11) and CNTEN (TIMERx_CTL[0]) is set to 1, the timer counter starts up counting. Once the CNT (TIMERx_CNT[23:0]) value reaches CMPDAT (TIMERx_CMP[23:0]) value, the CNTIF (TIMERx_INTSTS[0]) will be set to 1 and CNT (TIMERx_CNT[23:0]) value keeps up counting. In the meantime, if the CNTIEN (TIMERx_INTEN[0]) is enabled, the timer interrupt signal is generated and sent to NVIC to inform CPU also. User can change different CMPDAT (TIMERx_CMP[23:0]) value immediately without disabling timer counting and restarting timer counting in this mode.

For example, CMPDAT (TIMERx_CMP[23:0]) value is set as 80, first. The CNTIF (TIMERx_INTSTS[0]) will set to 1 when CNT (TIMERx_CNT[23:0]) value is equal to 80, timer counter is kept counting and CNT (TIMERx_CNT[23:0]) value will not goes back to 0, it continues to count 81, 82, 83,⋯ to 2^{24-1}, 0, 1, 2, 3, ⋯ to 2^{24-1} again and again. Next, if user programs CMPDAT (TIMERx_CMP[23:0]) value as 200 and clears CNTIF (TIMERx_INTSTS[0]), the CNTIF (TIMERx_INTSTS[0]) will set to 1 again when CNT (TIMERx_CNT[23:0]) value reaches to 200. At last, user programs CMPDAT (TIMERx_CMP[23:0]) as 500 and clears CNTIF (TIMERx_INTSTS[0]), the CNTIF (TIMERx_INTSTS[0]) will set to 1 again when CNT (TIMERx_CNT[23:0]) value reaches to 500.

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

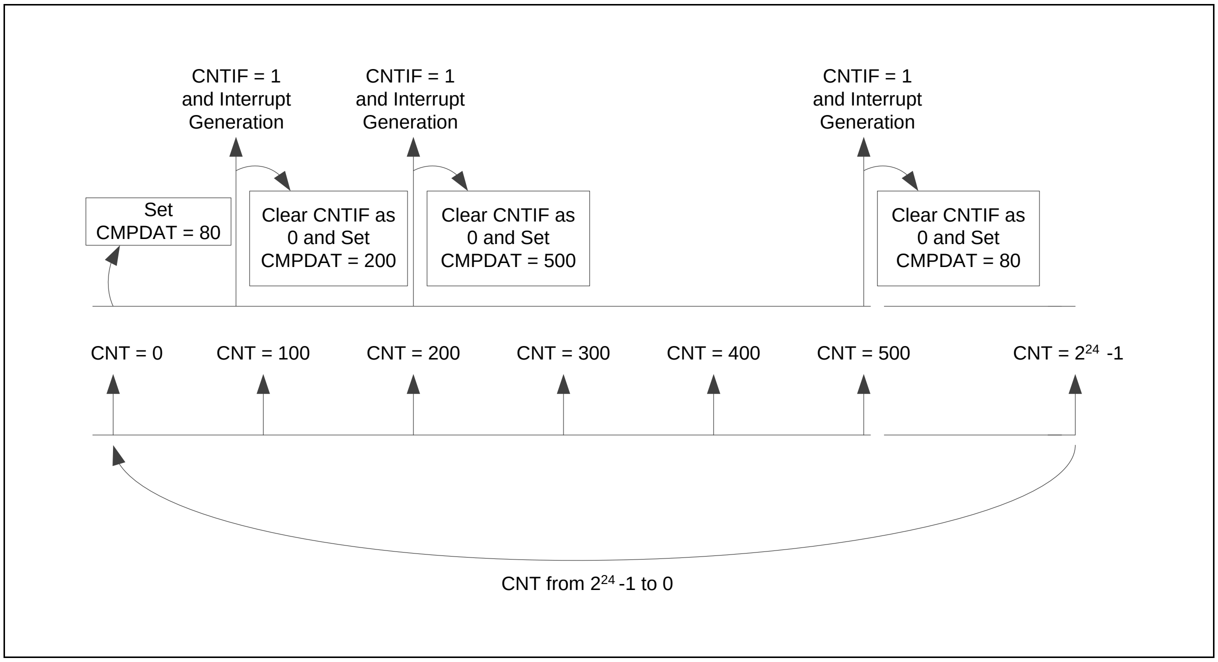


Figure 6.9-4 Continuous Counting Mode

6.9.5.5 Event Counting Mode

The timer controller also provides an application which can count the input event from TMx_ECNT (x= 0~5) pin and the number of event will reflect to CNT (TIMERx_CNT[23:0]) value. It is also called as event counting function.

User can enable or disable TMx_ECNT pin de-bounce circuit by setting CNTDBEN (TIMERx_CTL[14]). The input event frequency should be less than 1/4 TMRx_CLK if TMx_ECNT pin de-bounce disabled, or less than 1/8 TMRx_CLK if TMx_ECNT pin de-bounce enabled to assure the returned CNT (TIMERx_CNT[23:0]) value is correct, and user can also select edge detection phase of TMx_ECNT pin by setting CNTPHASE (TIMERx_CTL[13]) bit.

In event counting mode, the timer counting operation mode can be selected as One-shot, Periodic, Toggle-output and Continuous Counting mode to counts the counter value CNT (TIMERx_CNT[23:0]) for TMx_ECNT pin.

6.9.5.6 External Capture Mode

The event capture function is used to load CNT (TIMERx_CNT[23:0]) value to CAPDAT (TIMERx_CAP[23:0]) value while edge transition detected on TMx_EXT (x= 0~5) pin. In this mode, CAPFUNCS (TIMERx_CTL[17]) should be as 0 for select TMx_EXT transition is using to trigger event capture function.

User can enable or disable TMx_EXT pin de-bounce circuit by setting CAPDBEN (TIMERx_CTL[22]). The transition frequency of TMx_EXT pin should be less than 1/4 TMRx_CLK if TMx_EXT pin de-bounce disabled, or less than 1/8 TMRx_CLK 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_CTL[19:18]).

In event capture mode, if CAPCNTMD (TIMERx_CTL[20] is 0 (named as free-counting mode), user does not consider what timer counting operation mode is selected, the capture event occurred only if edge transition on TMx_EXT pin is detected.

In event capture mode, If CAPCNTMD (TIMERx_CTL[20]) is 1 (named as trigger-counting mode), the timer counter starts up counting only if the 1st edge transition (defined by CAPEDGE

(TIMERx_CTL[19:18])) on TMx_EXT pin detected and the capture event occurred only if the 2nd edge transition (defined by CAPEDGE (TIMERx_CTL[19:18])) on TMx_EXT pin detected. When capture event occurred, the timer counter restarts up counting.

Users must consider the Timer will keep register TIMERx_CAP unchanged and drop the new capture value, if the CPU does not clear the CAPIF (TIMERx_INTSTS[1]) status.

User can set SRCSEL (TIMERx_ECTL[16]) to select capture source. When SRCSEL is 0, capture source is from TMx_EXT pin. When SRC_SEL is 1, capture source is from RTC 1Hz signal for RTC clock calibration.

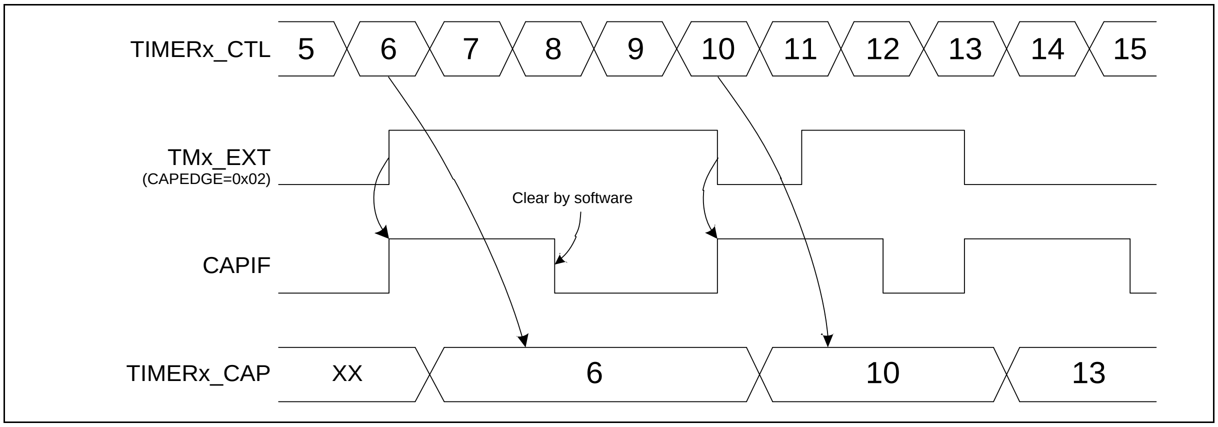


Figure 6.9-5 External CaptureFree-Counting Mode (CAPCNTMD (TIMERx_CTL[20] is 0))

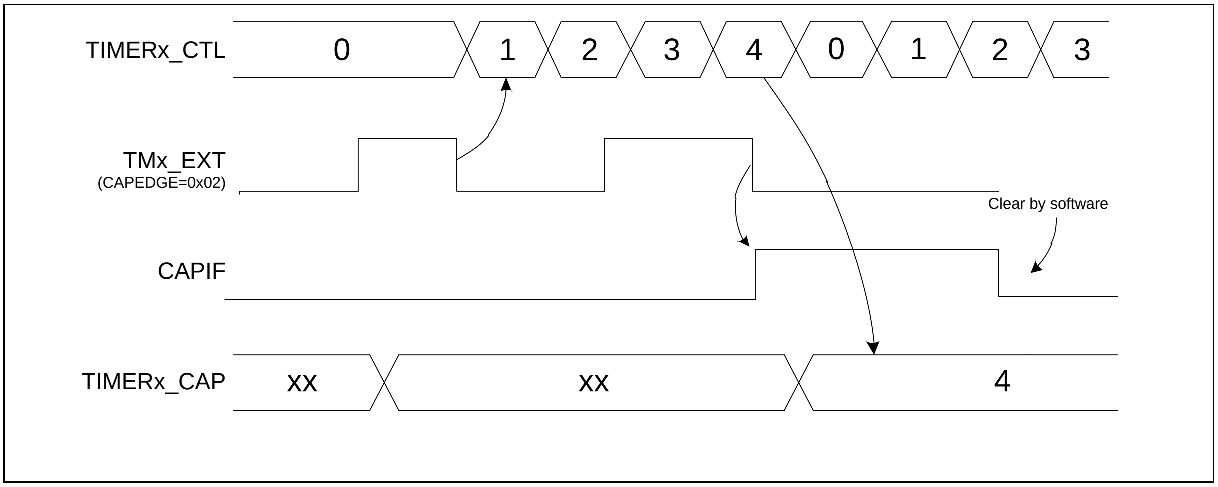


Figure 6.9-6 External CaptureTrigger-Counting Mode (CAPCNTMD (TIMERx_CTL[20] is 1))

6.9.5.7 External Reset Counter Mode

The timer controller also provides reset counter function to load CNT (TIMERx_CNT[23:0]) value to CAPDAT (TIMERx_CAP[23:0]) value and then reset CNT (TIMERx_CNT[23:0]) value while edge transition detected on TMx_EXT (x= 0~5). In this mode, most the settings are the same as event capture mode except CAPFUNCS (TIMERx_CTL[17]) should be as 1 for select TMx_EXT transition is using to trigger reset counter value.

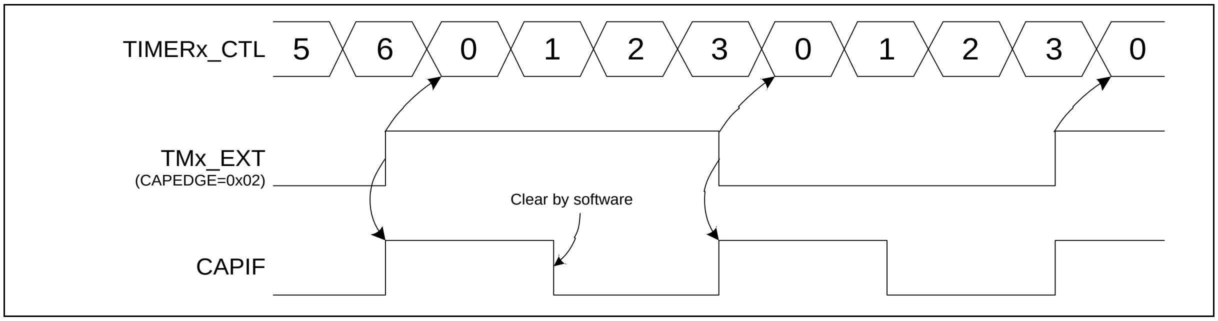


Figure 6.9-7 External CaptureReset Counter Mode (CAPCNTMD (TIMERx_CTL[20] is 0))

6.9.5.8 Inter-Timer Trigger Mode

The timer controller provides inter-timer trigger function to measure input frequency precisely. In inter-timer trigger function, Timer0 can trigger Timer1, Timer2 can trigger Timer3, and Timer4 can trigger Timer5.

If Timer0 and Timer1 are configured in Inter-timer trigger mode (INTRTGEN (TIMER0_CTL[24]) is 1), Timer0 is operating at event counting mode to count the input event from TM0_ECNT pin and generate an internal signal (INTR_TMR_TRG) to Timer1. Timer0 transit internal signal INTR_TMR_TRG from low to high if 1st input event detected on TM0_ECNT pin and then transit internal signal INTR_TMR_TRG from high to low if CNT (TIMER0_CNT[23:0]) value reaches CMPDAT (TIMERx_CMP[23:0]) value. Timer1 is operating at external capture trigger-counting mode to starts the timer counter up counting if rising edge transition on INTR_TMR_TRG detected and load CNT (TIMER1_CNT[23:0]) value to CAPDAT (TIMER1_CAP[23:0]) if falling edge transition on INTR_TMR_TRG detected.

If INTRTGEN (TIMER2_CTL[24]) is set to 1, Timer2 and Timer3 are configured in Inter-timer Trigger mode. The operation behavior of Timer2 and Timer3 in inter-timer trigger function is the same as the operation behavior of Timer0 and Timer1. The inter-timer trigger settings for Timer4 and Timer5 are the same.

Figure 6.9-8 describes how inter-timer trigger function operated with Timer0 and Timer1.In the end of inter-timer trigger function, the CNTIF (TIMER0_INTSTS[0]) will set to 1 and INTRTGEN (TIMER0_CTL[24]) is cleared automatically. In the meantime, if the CNTIEN (TIMER0_INTEN[0]) bit is set to 1, the timer interrupt signal is generated and sent to NVIC to inform CPU as well.

By using Inter-timer trigger function, the frequency of input event from TM0_ECNT pin could be measured more precisely.In Figure 6.9-8 when Timer0 counts 100 input events, the counter of Timer1 counts to 999. If Timer1's clock frequency is 10 MHz, then we know the time for 100 events is 99900ns. Therefore, the period of an input event is 999ns and the frequency of input event will be 1.001 MHz.

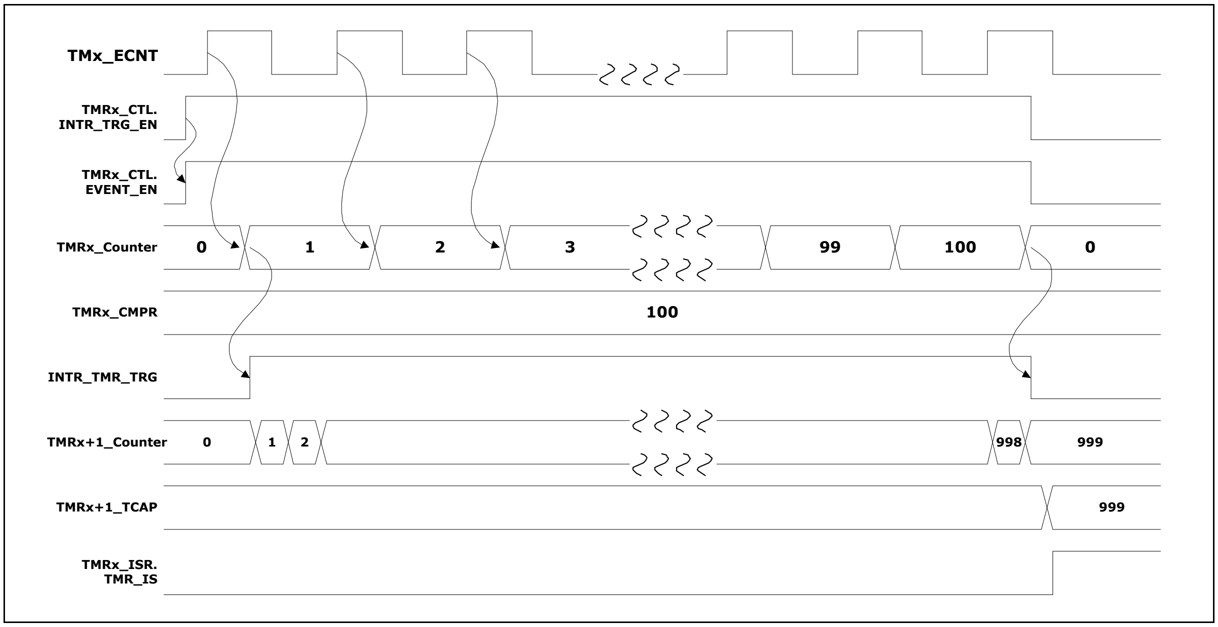


Figure 6.9-8 Inter-Timer Trigger Mode

6.9.5.9 Timer Trigger Function

The timer controller provides timer time-out interrupt or capture interrupt to trigger ADC and PDMA. If TRGSSEL (TIMERx_CTL[11]) is set to 0, time-out interrupt signal is used to trigger ADC and PDMA. If TRGSSEL (TIMERx_CTL[11]) is set to 1, capture interrupt signal is used to trigger ADC and PDMA.

When the TRGADC (TIMERx_CTL[8]) is set to 1, if the timer interrupt signal is generated, the timer controller will trigger ADC to start converter.

When the TRGPDMA (TIMERx_CTL[10]) is set to 1, if the timer interrupt signal is generated, the timer controller will trigger PDMA to start data transfer.

6.9.5.10 Timer Interrupt Flag

The timer controller supports two interrupt flags; one is CNTIF (TIMERx_INTSTS[0]) and is set while timer counter value CNT (TIMERx_CNT[23:0]) matches the timer compared value CMPDAT (TIMERx_CMP[23:0]), the other is CAPIF (TIMERx_INTSTS[1]) and its set when the transition on the TMx_EXT (x = 0 ~ 5) pin associated CAPEDGE (TIMERx_CTL[19:18]) setting detected.

6.9.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
TMR Base Address: **TMR_BA01 = 0xB005_0000** **TMR_BA23 = 0xB005_1000** **TMR_BA45 = 0xB005_2000**				
TIMER0_CTL	TMR_BA01+0x000	R/W	Timer 0 Control and Status Register	0x0000_0000
TIMER0_PRECNT	TMR_BA01+0x004	R/W	Timer 0 Pre-scale Counter Register	0x0000_0000
TIMER0_CMP	TMR_BA01+0x008	R/W	Timer 0 Compare Register	0x0000_0000
TIMER0_INTEN	TMR_BA01+0x00C	R/W	Timer 0 Interrupt Enable Register	0x0000_0000
TIMER0_INTSTS	TMR_BA01+0x010	R/W	Timer 0 Interrupt Status Register	0x0000_0000
TIMER0_CNT	TMR_BA01+0x014	R/W	Timer 0 Counter Data Register	0x0000_0000
TIMER0_CAP	TMR_BA01+0x018	R	Timer 0 Capture Data Register	0x0000_0000
TIMER0_ECTL	TMR_BA01+0x020	R/W	Timer 0 Extended Control Register	0x0000_0000
TIMER1_CTL	TMR_BA01+0x100	R/W	Timer 1 Control and Status Register	0x0000_0000
TIMER1_PRECNT	TMR_BA01+0x104	R/W	Timer 1 Pre-scale Counter Register	0x0000_0000
TIMER1_CMP	TMR_BA01+0x108	R/W	Timer 1 Compare Register	0x0000_0000
TIMER1_INTEN	TMR_BA01+0x10C	R/W	Timer 1 Interrupt Enable Register	0x0000_0000
TIMER1_INTSTS	TMR_BA01+0x110	R/W	Timer 1 Interrupt Status Register	0x0000_0000
TIMER1_CNT	TMR_BA01+0x114	R/W	Timer 1 Counter Data Register	0x0000_0000
TIMER1_CAP	TMR_BA01+0x118	R	Timer 1 Capture Data Register	0x0000_0000
TIMER1_ECTL	TMR_BA01+0x120	R/W	Timer 1 Extended Control Register	0x0000_0000
TIMER2_CTL	TMR_BA23+0x000	R/W	Timer 2 Control and Status Register	0x0000_0000
TIMER2_PRECNT	TMR_BA23+0x004	R/W	Timer 2 Pre-scale Counter Register	0x0000_0000
TIMER2_CMP	TMR_BA23+0x008	R/W	Timer 2 Compare Register	0x0000_0000
TIMER2_INTEN	TMR_BA23+0x00C	R/W	Timer 2 Interrupt Enable Register	0x0000_0000
TIMER2_INTSTS	TMR_BA23+0x010	R/W	Timer 2 Interrupt Status Register	0x0000_0000
TIMER2_CNT	TMR_BA23+0x014	R/W	Timer 2 Counter Data Register	0x0000_0000
TIMER2_CAP	TMR_BA23+0x018	R	Timer 2 Capture Data Register	0x0000_0000

TIMER2_ECTL	TMR_BA23+0x020	R/W	Timer 2 Extended Control Register	0x0000_0000
TIMER3_CTL	TMR_BA23+0x100	R/W	Timer 3 Control and Status Register	0x0000_0000
TIMER3_PRECNT	TMR_BA23+0x104	R/W	Timer 3 Pre-scale Counter Register	0x0000_0000
TIMER3_CMP	TMR_BA23+0x108	R/W	Timer 3 Compare Register	0x0000_0000
TIMER3_INTEN	TMR_BA23+0x10C	R/W	Timer 3 Interrupt Enable Register	0x0000_0000
TIMER3_INTSTS	TMR_BA23+0x110	R/W	Timer 3 Interrupt Status Register	0x0000_0000
TIMER3_CNT	TMR_BA23+0x114	R/W	Timer 3 Counter Data Register	0x0000_0000
TIMER3_CAP	TMR_BA23+0x118	R	Timer 3 Capture Data Register	0x0000_0000
TIMER3_ECTL	TMR_BA23+0x120	R/W	Timer 3 Extended Control Register	0x0000_0000
TIMER4_CTL	TMR_BA45+0x000	R/W	Timer 4 Control and Status Register	0x0000_0000
TIMER4_PRECNT	TMR_BA45+0x004	R/W	Timer 4 Pre-scale Counter Register	0x0000_0000
TIMER4_CMP	TMR_BA45+0x008	R/W	Timer 4 Compare Register	0x0000_0000
TIMER4_INTEN	TMR_BA45+0x00C	R/W	Timer 4 Interrupt Enable Register	0x0000_0000
TIMER4_INTSTS	TMR_BA45+0x010	R/W	Timer 4 Interrupt Status Register	0x0000_0000
TIMER4_CNT	TMR_BA45+0x014	R/W	Timer 4 Counter Data Register	0x0000_0000
TIMER4_CAP	TMR_BA45+0x018	R	Timer 4 Capture Data Register	0x0000_0000
TIMER4_ECTL	TMR_BA45+0x020	R/W	Timer 4 Extended Control Register	0x0000_0000
TIMER5_CTL	TMR_BA45+0x100	R/W	Timer 5 Control and Status Register	0x0000_0000
TIMER5_PRECNT	TMR_BA45+0x104	R/W	Timer 5 Pre-scale Counter Register	0x0000_0000
TIMER5_CMP	TMR_BA45+0x108	R/W	Timer 5 Compare Register	0x0000_0000
TIMER5_INTEN	TMR_BA45+0x10C	R/W	Timer 5 Interrupt Enable Register	0x0000_0000
TIMER5_INTSTS	TMR_BA45+0x110	R/W	Timer 5 Interrupt Status Register	0x0000_0000
TIMER5_CNT	TMR_BA45+0x114	R/W	Timer 5 Counter Data Register	0x0000_0000
TIMER5_CAP	TMR_BA45+0x118	R	Timer 5 Capture Data Register	0x0000_0000
TIMER5_ECTL	TMR_BA45+0x120	R/W	Timer 5 Extended Control Register	0x0000_0000

6.9.7 Register Description

Timer Control Register (TIMERx_CTL)

Register	Offset	R/W	Description	Reset Value
TIMER0_CTL	TMR_BA01+0x000	R/W	Timer 0 Control and Status Register	0x0000_0000
TIMER1_CTL	TMR_BA01+0x100	R/W	Timer 1 Control and Status Register	0x0000_0000
TIMER2_CTL	TMR_BA23+0x000	R/W	Timer 2 Control and Status Register	0x0000_0000
TIMER3_CTL	TMR_BA23+0x100	R/W	Timer 3 Control and Status Register	0x0000_0000
TIMER4_CTL	TMR_BA45+0x000	R/W	Timer 4 Control and Status Register	0x0000_0000
TIMER5_CTL	TMR_BA45+0x100	R/W	Timer 5 Control and Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				Reserved		INTRTGMD	INTRTGEN
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CMPCTL	CAPDBEN	Reserved	CAPCNTMD	CAPEDGE		CAPFUNCS	CAPEN
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	CNTDBEN	CNTPHASE	EXTCNTEN	TRGSSEL	TRGPDMA	Reserved	TRGADC
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ACTSTS	Reserved	OPMODE		ICEDEBUG	WKEN	RSTCNT	CNTEN

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:26]	**Reserved**	Reserved.
[25]	**INTRTGMD**	**Inter-timer Trigger Mode Selection** This bit controls the TIMERx (x = 0, 2, or 4) operating behavior when INTRTGEN (TIMERx_CTL[24])is set to 1. If INTRTGMD is set to 0 and INTRTGEN (TIMERx_CTL[24]) is set to 1, the TIMERx is operating at event counting mode to count the all input events from TMx_ECNT (x = 0, 2, or 4) pin. If INTRTGMD and INTRTGEN (TIMERx_CTL[24]) are both set to 1, TIMERx is operating at event counting mode and the number of first incoming events (defined by EVNTDPCNT (TIMERx_ECTL[31:24])) are ignored. 0 = TIMERx count the all input events from TMx_ECNT (x = 0, 2, or 4) pin. 1 = TIMERx ignored the number of first incoming events based on EVNTDPCNT (TIMERx_ECTL[31:24]). **Note:**In TIMERx+1_CTL, this bit is always 0.

Bits	Description	
[24]	**INTRTGEN**	**Inter-timer Trigger Function EnableBit** If INTRTGEN is set to 1 TIMERx (x = 0, 2, or 4), TIMERx and Timerx+1 are operating at inter-timer trigger mode. When Inter-timer Trigger functionis enabled, TIMERx is operating at event counting mode to count the input event from TMx_ECNT (x = 0, 2, or 4) pin and TIMERx+1 is operating at external capture trigger-counting mode. 0 = Inter-timer trigger function Disabled. 1 = Inter-timer trigger function Enabled. **Note:**In TIMERx+1_CTL, this bit is always 0.
[23]	**CMPCTL**	**Timer Compared Mode Selection** 0 = The behavior selection in one-shot, periodic or Toggle-output mode Disabled. When user updates CMPDAT (TIMERx_CMP) while timer is running in One-shot, Periodic or Toggle-output mode, CNT (TIMERx_CNT) will be reset to default value. 1 = The behavior selection in one-shot, periodic or Toggle-output mode Enabled. When user updates CMPDAT (TIMERx_CMP) while timer is running in One-shot, Periodic or Toggle-output mode, the limitations as bellows list, If updated CMPDAT (TIMERx_CMP) value > CNT (TIMERx_CNT), CMPDAT (TIMERx_CMP) will be updated and CNT (TIMERx_CNT) keep running continually. If updated CMPDAT (TIMERx_CMP) value = CNT (TIMERx_CNT), timer time-out interrupt will be asserted immediately. If updated CMPDAT (TIMERx_CMP) value < CNT (TIMERx_CNT), CNT (TIMERx_CNT) will be reset to default value. At the same time, prescale counter reloaded.
[22]	**CAPDBEN**	**Timer External Capture Pin De-bounce Enable Bit** 0 = TMx_EXT (x= 0~3) pin de-bounce Disabled. 1 = TMx_EXT (x= 0~3) pin de-bounce Enabled. **Note 1:** If this bit is enabled, the edge detection of TMx_EXT pin is detected with de-bounce circuit. **Note 2:** For Timer 1 and 3, when INTRTGEN (TIMERx_CTL[24]) is high, the capture signal is from internal of chip and the de-bounce circuit would not take effect no matter this bit is high or low.
[21]	**Reserved**	Reserved.
[20]	**CAPCNTMD**	**Timer Capture Counting Mode Selection** This bit indicates the behavior of 24-bit up-counting timer while CAPEN (TIMERx_CTL[16]) is set to high. If this bit is 0, the free-counting mode, the behavior of 24-bit up-counting timer is defined by OPMODE (TIMERx_CTL[5:4]) field. When CAPEN (TIMERx_CTL[16]) is set, CAPFUNCS (TIMERx_CTL[17]) is 0, and the transition of TC pin matches the CAPEDGE (TIMERx_CTL[19:18]) setting, the value of 24-bit up-counting timer will be saved into register TIMERx_CAP. If this bit is 1, Trigger-counting mode, 24-bit up-counting timer will be not counting and keep its value at 0. When CAPEN (TIMERx_CTL[16]) is set, CAPFUNCS (TIMERx_CTL[17]) is 0, and once the transition of external pin matches the 1st transition of CAPEDGE (TIMERx_CTL[19:18]) setting, the 24-bit up-counting timer will start counting. And then if the transition of external pin matches the 2nd transition of CAPEDGE (TIMERx_CTL[19:18]) setting, the 24-bit up-counting timer will stop counting. And its value will be saved into register TIMERx_CAP. 0 = Capture with free-counting timer mode. 1 = Capture with trigger-counting timer mode. **Note:** For TIMERx+1_CTL, if INTRTGEN (TIMERx_CTL[24]) is set, the CAPCNTMD will be forced to high, the capture with Trigger-counting Timer mode (where x = 0, 2, or 4).

Bits	Description	
[19:18]	**CAPEDGE**	**Timer External Capture Pin Edge Detection** For timer counter reset function and free-counting mode of timer capture function, the configurations are: 00 = A Falling edge on TMx_EXT (x= 0~5) pin will be detected. 01 = A Rising edge on TMx_EXT (x= 0~5) pin will be detected. 10 = Either Rising or Falling edge on TMx_EXT (x= 0~5) pin will be detected. 11 = Either Rising or Falling edge on TMx_EXT (x= 0~5) pin will be detected. For trigger-counting mode of timer capture function, the configurations are: 00 = 1stfalling edge on TMx_EXT (x= 0~5) pin triggers 24-bit timer to start counting, while 2nd falling edge triggers 24-bit timer to stop counting. 01 = 1st rising edge on TMx_EXT (x= 0~5) pin triggers 24-bit timer to start counting, while 2nd rising edge triggers 24-bit timer to stop counting. 10 = Falling edge on TMx_EXT (x= 0~5) pin triggers 24-bit timer to start counting, while rising edge triggers 24-bit timer to stop counting. 11 = Rising edge on TMx_EXT (x= 0~5) pin triggers 24-bit timer to start counting, while falling edge triggers 24-bit timer to stop counting. **Note:** For TIMER$_{x+1}$_CTL, if INTRTGEN (TIMER$_{x}$_CTL[24]) is set to 1, the CAPEDGE will be forced to 11 (where x = 0, 2, or 4).
[17]	**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= 0~5) pin is using to save the 24-bit timercounter value. **Note 2:** When CAPFUNCS is 1, transition on TMx_EXT (x= 0~5) pin is using to reset the 24-bit timer counter value. **Note 3:** For TIMERx+1_CTL(x = 0, 2, or 4), if INTRTGEN (TIMERx_CTL[24]) is set to 1, the CAPFUNCS will be forced to low.
[16]	**CAPEN**	**Timer External Capture Pin Enable Bit** This bit enables the TMx_EXT pin. 0 =TMx_EXT (x= 0~5) pin Disabled. 1 =TMx_EXT (x= 0~5) pin Enabled. **Note 1:** For TIMERx_CTL, if INTRTGEN (TIMERx_CTL[24]) is set to 1, the CAPEN will be forced to low and the TC pin transition is ignored (where x = 0, 2, or 4). **Note 2:** For TIMERx+1_CTL, if INTRTGEN (TIMERx_CTL[24]) is set to 1, the CAPEN will be forced to high (where x = 0, 2, or 4).
[15]	**Reserved**	Reserved.
[14]	**CNTDBEN**	**Timer Counter Pin De-bounce Enable Bit** 0 = TMx_CNT (x= 0~5) pin de-bounce Disabled. 1 = TMx_CNT (x= 0~5) pin de-bounce Enabled. **Note:** If this bit is set to 1, the edge detection of TMx_CNT pin is detected with de-bounce circuit.
[13]	**CNTPHASE**	**Timer External Count Phase** This bit indicates the detection phase of external counting pin TMx_CNT (x= 0~5). 0 = A Falling edge of external counting pin will be counted. 1 = A Rising edge of external counting pin will be counted.

Bits	Description	
[12]	**EXTCNTEN**	**Event Counter Mode Enable Bit** This bit is for external counting pin function enabled. 0 = Event counter mode Disabled. 1 = Event counter mode Enabled. **Note:** When timer is used as an event counter, this bit should be set to 1 and PCLK as timer clock source.
[11]	**TRGSSEL**	**Trigger Source Selection** If this bit is set to 1, capture interrupt can trigger ADC and PDMA. Otherwise, time-out interrupt can trigger ADC and PDMA. 0 = Time-out interrupt is used to trigger ADC and PDMA. 1 = Capture interrupt is used to trigger ADC and PDMA.
[10]	**TRGPDMA**	**Timer Trigger PDMA EnableBit** 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[11]) is set to 0, the time-out interrupt signal will trigger PDMA. If TRGSSEL (TIMERx_CTL[11]) is set to 1, the capture interrupt signal will trigger PDMA.
[9]	**Reserved**	Reserved.
[8]	**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[11]) is set to 0, the time-out interrupt signal will trigger ADC. If TRGSSEL (TIMERx_CTL[11]) is set to 1, the capture interrupt signal will trigger ADC.
[7]	**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.
[6]	**Reserved**	Reserved.
[5:4]	**OPMODE**	**Timer Counting Mode Selection** 00 = The Timer controller is operated in One-shot mode. 01 = The Timer controller is operated in Periodic mode. 10 = The Timer controller is operated in Toggle-output mode. 11 = The Timer controller is operated in Continuous Counting mode.
[3]	**ICEDEBUG**	**ICE Debug Mode Acknowledge Disable Bit** 0 = ICE debug mode acknowledgement affects TIMER counting. Timer counter will be held while CPU is held by ICE. 1 = ICE debug mode acknowledgement Disabled. Timer counter will keep going no matter CPU is held by ICE or not.

Bits	Description	
[2]	**WKEN**	**Wake-up Function Enable Bit** If this bit is set to 1, while CNTIF (TIMERx_INTSTS[0]) or CAPIF (TIMERx_INTSTS[1]) is 1, 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.
[1]	**RSTCNT**	**Timer Counter Reset Bit** Setting this bit will reset the internal 8-bit prescale counter, 24-bit up counter value CNT (TIMERx_CNT[23:0]) and also force CNTEN (TIMERx_CTL[0]) to 0. 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 and takes at least 3 TIMERx_CLK clock cycles.
[0]	**CNTEN**	**Timer Counting Enable Bit** 0 = Stops/Suspends counting. 1 = Starts counting. **Note 1:** In stop status, set CNTEN to 1 enables 24-bit counter keeps up counting from the last stop counting value. **Note 2:** This bit is auto-cleared by hardware in one-shot mode (OPMODE (TIMERx_CTL[5:4]) =00) when the timer interrupt flag TIF (TIMERx_INTSTS[0]) is generated. **Note 3:** Writing this bit 1 will not take any effect if RSTCNT (TIMERx_CTL[1]) is also set to 1 at the same time.

Timer Pre-scale Counter Register (TIMERx_PRECNT)

Register	Offset	R/W	Description	Reset Value
TIMER0_PRECNT	TMR_BA01+0x004	R/W	Timer 0 Pre-scale Counter Register	0x0000_0000
TIMER1_PRECNT	TMR_BA01+0x104	R/W	Timer 1 Pre-scale Counter Register	0x0000_0000
TIMER2_PRECNT	TMR_BA23+0x004	R/W	Timer 2 Pre-scale Counter Register	0x0000_0000
TIMER3_PRECNT	TMR_BA23+0x104	R/W	Timer 3 Pre-scale Counter Register	0x0000_0000
TIMER4_PRECNT	TMR_BA45+0x004	R/W	Timer 4 Pre-scale Counter Register	0x0000_0000
TIMER5_PRECNT	TMR_BA45+0x104	R/W	Timer 5 Pre-scale 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**
PSC							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**PSC**	**Prescale Counter** Timer input clock or event source is divided by (PSC+1) before it is fed to the timer up counter. If this field is 0 (PSC = 0), then there is no scaling. **Note:** If the PSC value is changed, CNT (TIMERx_CNT) is reset to 0 and prescale counter is reloaded.

Timer Compare Register (TIMERx_CMP)

Register	Offset	R/W	Description	Reset Value
TIMER0_CMP	TMR_BA01+0x008	R/W	Timer 0 Compare Register	0x0000_0000
TIMER1_CMP	TMR_BA01+0x108	R/W	Timer 1 Compare Register	0x0000_0000
TIMER2_CMP	TMR_BA23+0x008	R/W	Timer 2 Compare Register	0x0000_0000
TIMER3_CMP	TMR_BA23+0x108	R/W	Timer 3 Compare Register	0x0000_0000
TIMER4_CMP	TMR_BA45+0x008	R/W	Timer 4 Compare Register	0x0000_0000
TIMER5_CMP	TMR_BA45+0x108	R/W	Timer 5 Compare Register	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**CMPDAT**	**Timer Compared Value** CMPDAT is a 24-bit compared value register. When the internal 24-bit up counter value is equal to CMPDAT value, the CNTIF (TIMERx_INTSTS[0] Timer Interrupt Flag) will be 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, or the core will run into unknown state. **Note 2:**When the timer is operating in Continuous Counting mode (OPMODE (TIMERx_CTL[5:4] is 11), the 24-bit up counter will keep counting continuously even if user writes a new value into CMPDAT field. **Note 3:** Whenthe timer is not operating in Continuous Counting mode (OPMODE (TIMERx_CTL[5:4] is not 11), the 24-bit up counter will restart counting from 0 and usethe newest CMPDAT value as the timer compared value when user writes a new value into the CMPDAT field.In addition, the prescale counter will be reloaded.

Timer Interrupt Enable Register (TIMERx_INTEN)

Register	Offset	R/W	Description	Reset Value
TIMER0_INTEN	TMR_BA01+0x00C	R/W	Timer 0 Interrupt Enable Register	0x0000_0000
TIMER1_INTEN	TMR_BA01+0x10C	R/W	Timer 1 Interrupt Enable Register	0x0000_0000
TIMER2_INTEN	TMR_BA23+0x00C	R/W	Timer 2 Interrupt Enable Register	0x0000_0000
TIMER3_INTEN	TMR_BA23+0x10C	R/W	Timer 3 Interrupt Enable Register	0x0000_0000
TIMER4_INTEN	TMR_BA45+0x00C	R/W	Timer 4 Interrupt Enable Register	0x0000_0000
TIMER5_INTEN	TMR_BA45+0x10C	R/W	Timer 5 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						CAPIEN	CNTIEN

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**CAPIEN**	**Timer External Capture Interrupt Enable Bit** 0 = TMx_EXT (x= 0~5) pin detection Interrupt Disabled. 1 = TMx_EXT (x= 0~5) pin detection Interrupt Enabled. **Note:**CAPIEN is used to enable timer external interrupt. If CAPIEN is enabled, the timer will rise an interrupt when CAPIF (TIMERx_INTSTS[1]) is 1.
[0]	**CNTIEN**	**Timer Interrupt EnableBit** 0 = Timer Interrupt Disabled. 1 = Timer Interrupt Enabled. **Note:**If this bit is enabled, when the timer interrupt flag CNTIF(TIMERx_INTSTS[0]) is set to 1, the timer interrupt signal is generated and informed to CPU.

Timer Interrupt Status Register (TIMERx_INTSTS)

Register	Offset	R/W	Description	Reset Value
TIMER0_INTSTS	TMR_BA01+0x010	R/W	Timer 0 Interrupt Status Register	0x0000_0000
TIMER1_INTSTS	TMR_BA01+0x110	R/W	Timer 1 Interrupt Status Register	0x0000_0000
TIMER2_INTSTS	TMR_BA23+0x010	R/W	Timer 2 Interrupt Status Register	0x0000_0000
TIMER3_INTSTS	TMR_BA23+0x110	R/W	Timer 3 Interrupt Status Register	0x0000_0000
TIMER4_INTSTS	TMR_BA45+0x010	R/W	Timer 4 Interrupt Status Register	0x0000_0000
TIMER5_INTSTS	TMR_BA45+0x110	R/W	Timer 5 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	**CAPFEDF**	**CAPDATOF**	**TWKF**	**Reserved**		**CAPIF**	**CNTIF**

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**CAPFEDF**	**Capture Falling Edge Detected Flag** This flag indicates theedge detected on TMx_EXT pin is rising edge or falling edge. 0 = Rising edge detected on TMx_EXT (x= 0~5) pin. 1 = Falling edge detected on TMx_EXT (x= 0~5) pin. **Note 1:** The timer updates this flag when it updates the Timer Capture Data (TMR_CAP[23:0]) value. **Note 2:**When a new incoming capture event detected before CPU clearing the CAPIF (TIMERx_INTSTS[1]) status, Timer will keep this bit unchanged.

Bits	Description	
[5]	**CAPDATOF**	**Capture Data Overflow Flag** This status is to indicate there is a new incoming capture event detected before CPU clearing the CAPIF (TIMERx_INTSTS[1]) status. If the above condition occurred, the Timer will keep register TIMERx_CAP unchanged and drop the new capture value. 0 **=** New incoming capture event didn't detect before CPU clearing CAPIF (TIMERx_INTSTS[1]) status. 1 = New incoming capture event detected before CPU clearing CAPIF (TIMERx_INTSTS[1]) status. **Note:** This bit is cleared by writing 1 to it.
[4]	**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.
[3:2]	**Reserved**	Reserved.
[1]	**CAPIF**	**Timer External Capture Interrupt Flag** This bit indicates the timer external capture interrupt flag status. 0 = TMx_EXT (x= 0~5) pin interrupt did not occur. 1 = TMx_EXT (x= 0~5) pin interrupt occurred. **Note 1:** This bit is cleared by writing 1 to it. **Note 2:** When CAPEN (TIMERx_CTL[16]) bit is set, CAPFUNCS (TIMERx_CTL[17]) bit is 0, and a transition on TMx_EXT (x= 0~5) pin matched the CAPEDGE (TIMERx_CTL[19:18]) setting, this bit will set to 1 by hardware. **Note 3:** If a new incoming capture event detected before CPU clearing the CAPIF status, the Timer will keep register TIMERx_CAP unchanged and drop the new capture value.
[0]	**CNTIF**	**Timer Interrupt Status** This bit indicates the interrupt flag status of Timer while 24-bit timer up counter CNT (TIMERx_CNT[23:0]) value reaches to CMPDAT (TIMERx_CMP[23:0]) value. 0 = No effect. 1 = CNT (TIMERx_CNT[23:0])value matches the CMPDAT (TIMERx_CMP[23:0]) value. **Note:** This bit is cleared by writing 1 to it.

Timer Counter Data Register (TIMERx_CNT)

Register	Offset	R/W	Description	Reset Value
TIMER0_CNT	TMR_BA01+0x014	R/W	Timer 0 Counter Data Register	0x0000_0000
TIMER1_CNT	TMR_BA01+0x114	R/W	Timer 1 Counter Data Register	0x0000_0000
TIMER2_CNT	TMR_BA23+0x014	R/W	Timer 2 Counter Data Register	0x0000_0000
TIMER3_CNT	TMR_BA23+0x114	R/W	Timer 3 Counter Data Register	0x0000_0000
TIMER4_CNT	TMR_BA45+0x014	R/W	Timer 4 Counter Data Register	0x0000_0000
TIMER5_CNT	TMR_BA45+0x114	R/W	Timer 5 Counter Data Register	0x0000_0000

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

Bits	Description	
[31]	**RSTACT**	**Reset Active** This bit indicates if the counter reset operation active. When user write this register, timer starts to reset its internal 24-bit timer up-counter and 8-bit pre-scale counter to 0. At the same time, timer set this flag to 1 to indicate the counter reset operation is in progress. Once the counter reset operation done, timer clear this bit to 0 automatically. 0 = Reset operation is done. 1 = Reset operation triggered by writing TIMERx_CNT is in progress. **Note:** This bit is read only. Write operation wouldn't take any effect.
[30:24]	**Reserved**	Reserved.
[23:0]	**CNT**	**Timer Counter Data (Read)** This field can reflect the internal 24-bit timer counter value or external event input counter value from TMx_ECNT (x=0~5) pin. **Counter Reset (Write)** User can write any value to TIEMRx_CNT to reset internal 24-bit timer up-counter and 8-bit pre-scale counter. This reset operation wouldn't affect any other timer control registers and circuit. After reset completed, the 24-bit timer up-counter and 8-bit pre-scale counter restart the counting based on the TIMERx_CTL register setting.

Timer Capture Data Register (TIMERx_CAP)

Register	Offset	R/W	Description	Reset Value
TIMER0_CAP	TMR_BA01+0x018	R	Timer 0 Capture Data Register	0x0000_0000
TIMER1_CAP	TMR_BA01+0x118	R	Timer 1 Capture Data Register	0x0000_0000
TIMER2_CAP	TMR_BA23+0x018	R	Timer 2 Capture Data Register	0x0000_0000
TIMER3_CAP	TMR_BA23+0x118	R	Timer 3 Capture Data Register	0x0000_0000
TIMER4_CAP	TMR_BA45+0x018	R	Timer 4 Capture Data Register	0x0000_0000
TIMER5_CAP	TMR_BA45+0x118	R	Timer 5 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_CTL[16]) bit is set, CAPFUNCS (TIMERx_CTL[17]) bit is 0, CAPCNTMD (TIMERx_CTL[20]) bit is 0, and the transition on TMx_EXT pin matched the CAPEDGE (TIMERx_CTL[19:18]) setting, CAPIF (TIMERx_INTSTS[1]) will set to 1 and the current timer counter value CNT (TIMERx_CNT[23:0]) will be auto-loaded into this CAPDAT field. When CAPEN (TIMERx_CTL[16]) bit is set, CAPFUNCS (TIMERx_CTL[17]) bit is 0, CAPCNTMD (TIMERx_CTL[20]) bit is 1, and the transition on TMx_EXT pin matched the 2nd transition of CAPEDGE (TIMERx_CTL[19:18]) setting, CAPIF (TIMERx_INTSTS[1]) will set to 1 and the current timer counter value CNT (TIMERx_CNT[23:0]) will be auto-loaded into this CAPDAT field. **Note:**When edge transition defined by CAPEDGE (TIMERx_CTL[19:18]) is detected on TMx_EXT (x = 0 ~ 3) before CPU clears the CAPIF (TIMERxISR[1]) status, the timer keeps this value unchanged and CAPDATOF (TIMERx_INTSTS[5]) is set to 1.

Timer Extended Control Register (TIMERx_ECTL)

Register	Offset	R/W	Description	Reset Value
TIMER0_ECTL	TMR_BA01+0x020	R/W	Timer 0 Extended Control Register	0x0000_0000
TIMER1_ECTL	TMR_BA01+0x120	R/W	Timer 1 Extended Control Register	0x0000_0000
TIMER2_ECTL	TMR_BA23+0x020	R/W	Timer 2 Extended Control Register	0x0000_0000
TIMER3_ECTL	TMR_BA23+0x120	R/W	Timer 3 Extended Control Register	0x0000_0000
TIMER4_ECTL	TMR_BA45+0x020	R/W	Timer 4 Extended Control Register	0x0000_0000
TIMER5_ECTL	TMR_BA45+0x120	R/W	Timer 5 Extended Control Register	0x0000_0000

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

Bits	Description	
[31:24]	**EVNTDPCNT**	**Event Drop Count** This field indicates timer how many events dropped after inter-timer trigger function enable. For example, if user configured EVNTDPCNT to 7, timer would drop 7 first incoming events and starts the inter-timer trigger operation when it get 8th event. **Note:** ECNTDPCNT only takes effect when INTRTGEN (TIMERx_CTL[24]) INTRTGMD (TIMERx_CTL[25]) are both set to 1.
[23:17]	**Reserved**	Reserved.
[16]	**SRCSEL**	**Capture Source Select** 0 = Capture source is from TMx_EXT pin. 1 = Capture source is from RTC 1Hz signal for RTC clock calibration.
[15:0]	**Reserved**	Reserved.

6.10 Pulse Width Modulation (PWM)

6.10.1 Overview

This chip has 2 PWM controllers, PWM0 and PWM1. Each PWM controller has 4 independent PWM outputs.

PWM0 has 4 independent PWM outputs, CH0~CH3, or 2 complementary PWM pairs, (CH0, CH1), (CH2, CH3) with 2 programmable dead-zone generators. PWM1 has 4 independent PWM outputs, CH4~CH7, or 2 complementary PWM pairs, (CH4, CH5), (CH6, CH7) with 2 programmable dead-zone generators. Each PWM pair has one prescaler, one clock divider, two clock selectors, two 16-bit PWM counters, two 16-bit comparators, and one dead-zone e generator. They are all driven by APB system clock (PCLK) in chip. Each PWM channel can be used as a timer and issue interrupt independently.

Two channels PWM Timers in one pair share the same prescaler. The Clock divider provides each PWM channel with 5 divided clock sources (1, 1/2, 1/4, 1/8, 1/16). Each channel receives its own clock signal from clock divider which receives clock from 8-bit prescaler. The 16-bit down-counter in each channel receive clock signal from clock selector and can be used to handle one PWM period. The 16-bit comparator compares PWM counter value with threshold value in register CMR (PWM_CMR[15:0]) loaded previously to generate PWM duty cycle. The clock signal from clock divider is called PWM clock. The dead-zone generator utilizes PWM clock as clock source. Once the dead-zone generator is enabled, two outputs of the corresponding PWM channel pair will be replaced by the output of dead-zone generator. The dead-zone generator is used to control off-chip power device.

To prevent PWM driving output pin with unsteady waveform, 16-bit down-counter and 16-bit comparator are implemented with double buffering feature. User can feel free to write data to counter buffer register and comparator buffer register without generating glitch. When 16-bit down-counter reaches zero, the interrupt request is generated to inform CPU that time is up. When counter reaches zero, if counter is set as periodic mode, it is reloaded automatically and start to generate next cycle. User can set PWM counter as one-shot mode instead of periodic mode. If counter is set as one-shot mode, counter will stop and generate one interrupt request when it reaches zero. The value of comparator is used for pulse width modulation. The counter control logic changes the output level when down-counter value matches the value of compare register.

6.10.2 Features

- 8 PWM channels with a 16-bit down counter and an interrupt each
- 4 complementary PWM pairs, (CH0, CH1), (CH2, CH3), (CH4, CH5), (CH6, CH7), each with a programmable dead-zone generator
- Internal 8-bit prescaler and a clock divider for each PWM paired channel
- Independent clock source selection for each PWM channel
- Internal 16-bit down counter and 16-bit comparator for each independent PWM channel
- PWM down-counter supports One-shot or Periodic mode

6.10.3 Block Diagram

Figure 6.10-1 describes the architecture of one PWM pair.

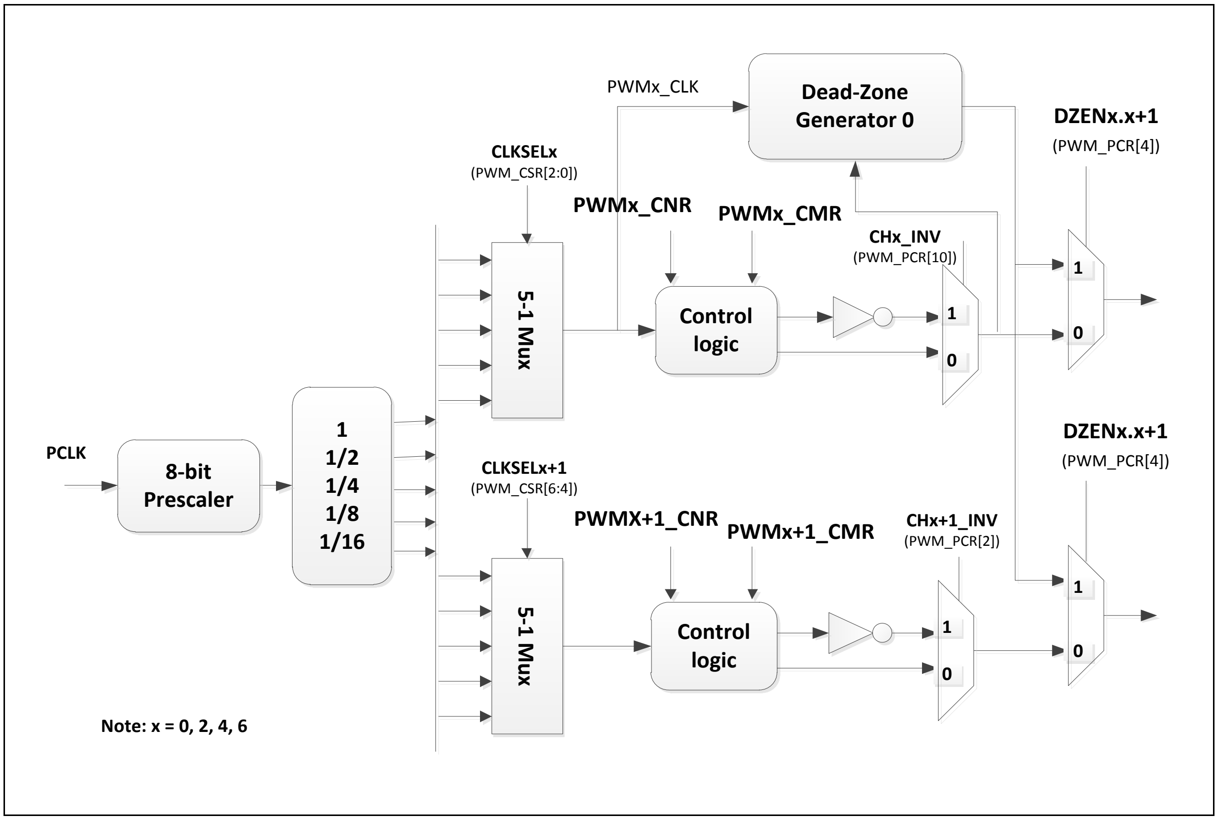


Figure 6.10-1 Two Channels of PWM in One Pair

6.10.4 Basic Configuration

6.10.4.1 PWM0 Basic Configurations

- Clock Source Configuration
  - Enable PWM0 controller clock in PWM0CKEN (CLK_PCLKEN1[26]).
- Reset Configuration
  - Reset PWM0 controller in PWM0RST (SYS_APBIPRST1[26]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
PWM00	PWM00	PF.5	MFP4
		PD.12, PG.0	MFP6
		PG.10	MFP7
PWM01	PWM01	PF.6	MFP4
		PD.13, PG.1	MFP6
		PA.15	MFP7

PWM02	PWM02	PB.13, PF.7	MFP4
		PD.14, PG.2	MFP6
		PA.14	MFP7
PWM03	PWM03	PF.8	MFP4
		PD.15, PG.3	MFP6
		PA.13	MFP7

6.10.4.2 PWM1 Basic Configurations

- Clock Source Configuration
  - Enable PWM1 controller clock in PWM1CKEN (CLK_PCLKEN1[27]).
- Reset Configuration
  - Reset PWM1 controller in PWM1RST (SYS_APBIPRST1[27]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
PWM10	PWM10	PB.12	MFP2
		PF.9	MFP4
		PG.6, PG.11	MFP6
PWM11	PWM11	PB.11	MFP2
		PF.10	MFP4
		PG.7, PG.12	MFP6
PWM12	PWM12	PB.10	MFP2
		PE.10	MFP4
		PG.8, PG.13	MFP6
PWM13	PWM13	PB.9	MFP2
		PE.12	MFP4
		PG.9, PG.14	MFP6

6.10.5 Functional Description

6.10.5.1 PWM Timer Operation

The PWM period and duty control are decided by register CNR (PWM_CNR[15:0]) and CMR (PWM_CMR[15:0]). The PWM-timer timing operation is shown in Figure 6.10-2. The pulse width modulation follows the formula below and the legend of PWM-Timer Comparator is shown in Figure 6.10-2.

PWM frequency = PCLK/(prescale+1)*(clock divider)/(CNR+1); depending on selected PWM channel.

Duty ratio = (CMR+1)/(CNR+1).

CMR >= CNR: PWM output is always high.

CMR < CNR: PWM low width = (CNR - CMR) unit1; PWM high width = (CMR+1) unit.

If CMR = 0: PWM low width = (CNR) unit; PWM high width = 1 unit.

Note: 1. Unit = one PWM clock cycle.

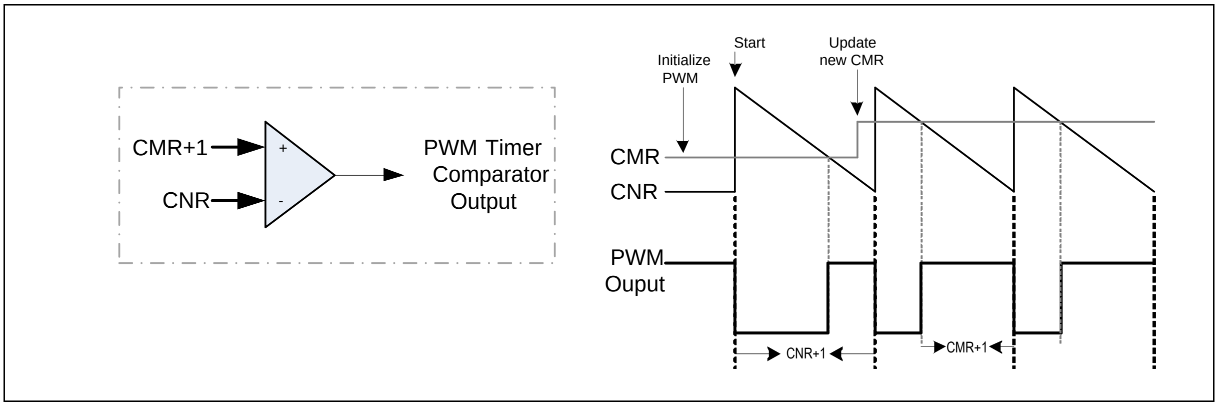


Figure 6.10-2 Legend of Internal Comparator Output of PWM-Timer

6.10.5.2 PWM Double Buffering, Periodic and One-shot Operation

The PWM timers have double buffering function; the reload value is updated at the start of next period without affecting current timer operation. The PWM counter value can be written into CNR (PWM_CNR[15:0]).

Channel0, For example, the bit CH0MOD (PWM_PCR[3]) defines PWM operation in Periodic or One-shot mode If CH0MOD (PWM_CTL[3]) is set to one (Periodic mode), the controller loads CNR (PWM_CNR[15:0]) to PWM counter when PWM counter reaches zero. If CNR (PWM_CNR[15:0]) is set to zero, PWM counter will be halt when PWM counter counts to zero.

In One-shot mode (CH0MOD=0; PWM_PCR[3]=0), the corresponding channel will output only one cycle of duty waveform and then PWM counter will be stopped if no further corresponding duty register updated. When PWM counter is running, updating corresponding duty register will engage the next cycle of duty waveform.

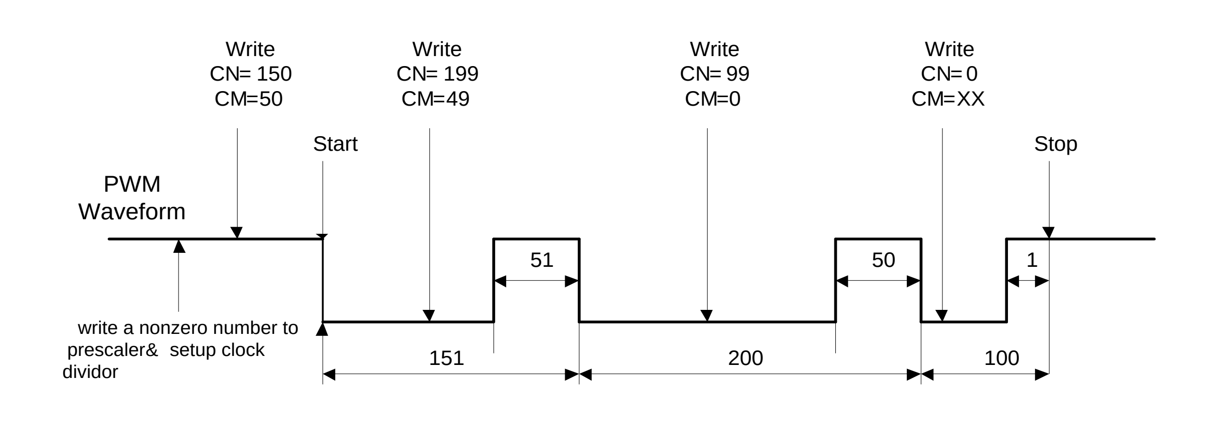


Figure 6.10-3 PWM Double Buffer Illustration

6.10.5.3 Modulate Duty Ratio

The double buffering function allows CMR (PWM_CMR[15:0]) to be written at any point in current cycle. The loaded value will take effect from next cycle.

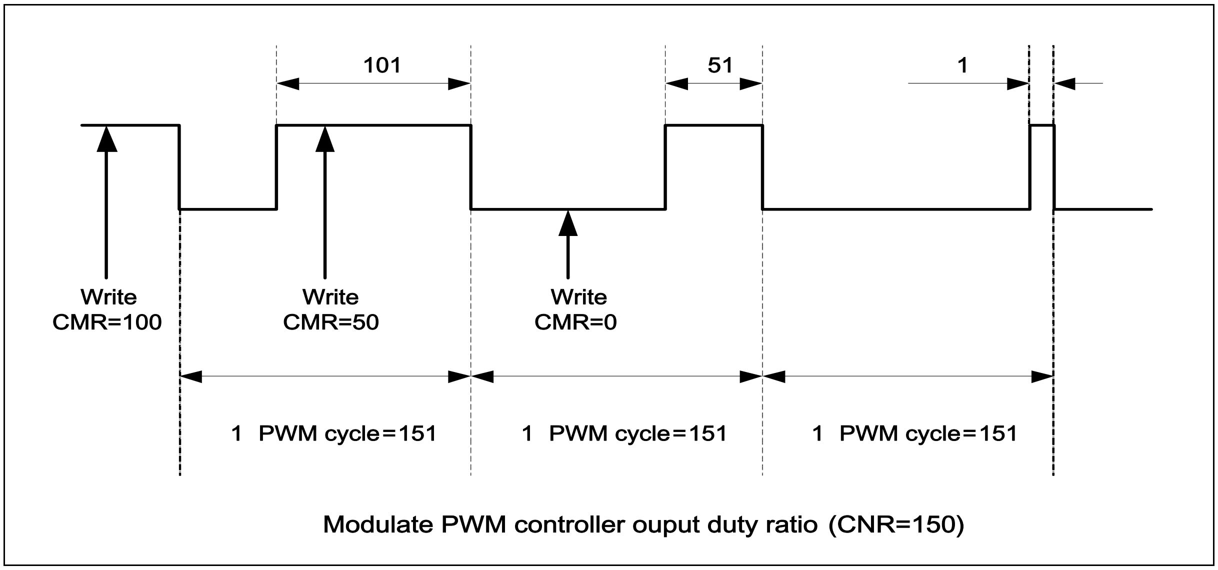


Figure 6.10-4 PWM Controller Output Duty Ratio

6.10.5.4 Dead-Zone Generator

The PWM implements dead-zone generator. They are built for power device protection. This function generates a programmable time gap called “dead-zone” to delay PWM rising output, and it is in order to prevent damage for the power switch devices that connected to the PWM output pins. User can program Dead-Zone counter to determine the dead-zone interval, for example, of channel 0, 1 pair with DZL01 (PWMx_PPR[23:16]). The dead-zone period of channel 0, 1 pair can be calculated by (PWM_CLK period x (DZL01 + 1)) and the enable bit is DZEN01 (PWM_PCR[4]).

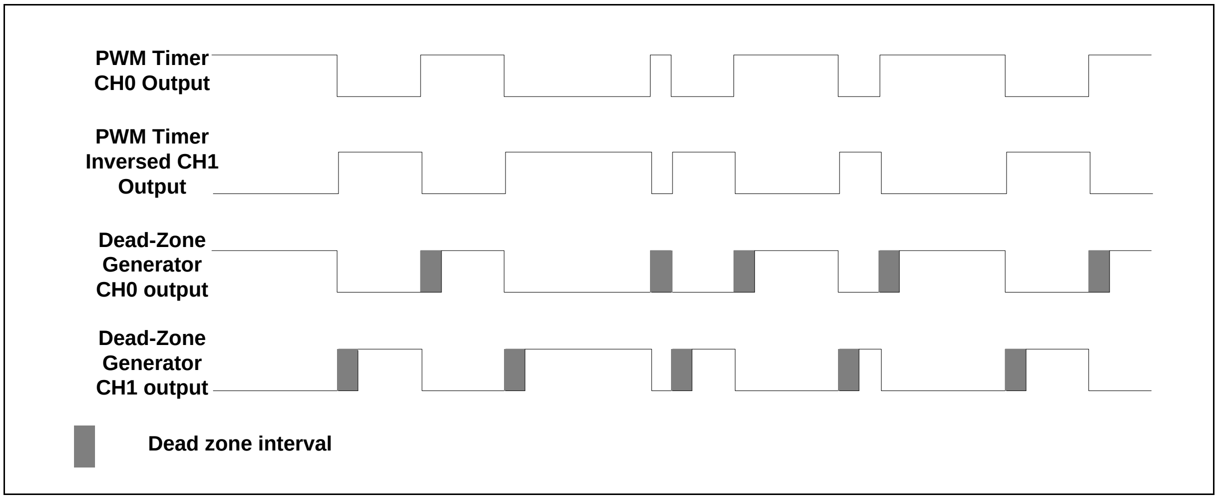


Figure 6.10-5 Paired PWM Output with Dead-Zone Generation Operation

6.10.5.5 PWM-Timer Start Procedure

Take PWM channel 0 for example, and the following procedure is for starting a PWM drive.

- Set clock selector CLKSEL0 (PWM_CSR[2:0])
- Set prescaler PRESCALE01 (PWM_PPR[7:0])
- Set inverter on/off, dead-zone generator on/off, Periodic/One-shot mode and stop PWM-

timer (PWM_PCR)

- Set interrupt enable register PIER0 (PWM_PIER[0])
- Set the corresponding GPI/O pins to PWM function
- Set PWM comparator register CMR (PWM_CMR[15:0]) and PWM counter register CNR (PWM_CNR[15:0]) for setting PWM period and duty length
- Enable PWM down-counter start running (Set CH0EN = 1 (PWM_PCR[0]))

The procedure mentioned above may be not set up in the order and PWM Timer can still work fine.

6.10.5.6 PWM-Timer Stop Procedure

Take PWM channel 0 for example.

Method 1: Set 16-bit down counter CNR (PWM_CNR[15:0]) as 0, and monitor data register PDR (PWM_PDR[15:0]). When PDR (PWM_PDR[15:0]) reaches to 0, disable PWM Timer by setting CH0EN = 0 (PWM_PCR[0] = 0). (Recommended)

Method 2: Set 16-bit down counter CNR (PWM_CNR[15:0]) as 0. When interrupt request happen, disable PWM Timer by setting CH0EN = 0 (PWM_PCR[0] = 0). (Recommended)

Method 3: Disable PWM Timer by setting CH0EN = 0 (PWM_PCR[0] = 0). (Not recommended)

6.10.6 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 = 0xB005_8000** **PWM1_BA = 0xB005_9000**				
PWM0_PPR	PWM0_BA+0x000	R/W	PWM0 Pre-scale Register	0x0000_0000
PWM1_PPR	PWM1_BA+0x000	R/W	PWM1 Pre-scale Register	0x0000_0000
PWM0_CSR	PWM0_BA+0x004	R/W	PWM0 Clock Select Register	0x0000_0000
PWM1_CSR	PWM1_BA+0x004	R/W	PWM1 Clock Select Register	0x0000_0000
PWM0_PCR	PWM0_BA+0x008	R/W	PWM0 Control Register	0x0000_0000
PWM1_PCR	PWM1_BA+0x008	R/W	PWM1 Control Register	0x0000_0000
PWM0_CNR	PWM0_BA+0x00C	R/W	PWM0 Counter Register 0	0x0000_0000
PWM0_CMR	PWM0_BA+0x010	R/W	PWM0 Comparator Register 0	0x0000_0000
PWM0_PDR	PWM0_BA+0x014	R	PWM0 Data Register 0	0x0000_0000
PWM1_CNR	PWM0_BA+0x018	R/W	PWM0 Counter Register 1	0x0000_0000
PWM1_CMR	PWM0_BA+0x01C	R/W	PWM0 Comparator Register 1	0x0000_0000
PWM1_PDR	PWM0_BA+0x020	R	PWM0 Data Register 1	0x0000_0000
PWM2_CNR	PWM0_BA+0x024	R/W	PWM0 Counter Register 2	0x0000_0000
PWM2_CMR	PWM0_BA+0x028	R/W	PWM0 Comparator Register 2	0x0000_0000
PWM2_PDR	PWM0_BA+0x02C	R	PWM0 Data Register 2	0x0000_0000
PWM3_CNR	PWM0_BA+0x030	R/W	PWM0 Counter Register 3	0x0000_0000
PWM3_CMR	PWM0_BA+0x034	R/W	PWM0 Comparator Register 3	0x0000_0000
PWM3_PDR	PWM0_BA+0x038	R	PWM0 Data Register 3	0x0000_0000
PWM4_CNR	PWM1_BA+0x00C	R/W	PWM1 Counter Register 4	0x0000_0000
PWM4_CMR	PWM1_BA+0x010	R/W	PWM1 Comparator Register 4	0x0000_0000
PWM4_PDR	PWM1_BA+0x014	R	PWM1 Data Register 4	0x0000_0000
PWM5_CNR	PWM1_BA+0x018	R/W	PWM1 Counter Register 5	0x0000_0000
PWM5_CMR	PWM1_BA+0x01C	R/W	PWM1 Comparator Register 5	0x0000_0000
PWM5_PDR	PWM1_BA+0x020	R	PWM1 Data Register 5	0x0000_0000
PWM6_CNR	PWM1_BA+0x024	R/W	PWM1 Counter Register 6	0x0000_0000

PWM6_CMR	PWM1_BA+0x028	R/W	PWM1 Comparator Register 6	0x0000_0000
PWM6_PDR	PWM1_BA+0x02C	R	PWM1 Data Register 6	0x0000_0000
PWM7_CNR	PWM1_BA+0x030	R/W	PWM1 Counter Register 7	0x0000_0000
PWM7_CMR	PWM1_BA+0x034	R/W	PWM1 Comparator Register 7	0x0000_0000
PWM7_PDR	PWM1_BA+0x038	R	PWM1 Data Register 7	0x0000_0000
PWM0_PIER	PWM0_BA+0x03C	R/W	PWM0 Timer Interrupt Enable Register	0x0000_0000
PWM1_PIER	PWM1_BA+0x03C	R/W	PWM1 Timer Interrupt Enable Register	0x0000_0000
PWM0_PIIR	PWM0_BA+0x040	R/W	PWM0 Timer Interrupt Indication Register	0x0000_0000
PWM1_PIIR	PWM1_BA+0x040	R/W	PWM1 Timer Interrupt Indication Register	0x0000_0000

6.10.7 Register Description

PWM0 Pre-scale Register (PWM0_PPR)

Register	Offset	R/W	Description	Reset Value
PWM0_PPR	PWM0_BA+0x000	R/W	PWM0 Pre-scale Register	0x0000_0000

31	30	29	28	27	26	25	24
DZL23							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DZL01							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PRESCALE23							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PRESCALE01							

Bits	Description	
[31:24]	**DZL23**	**Dead-zone Length Register 1** These 8 bits determine the dead-zone length of channel 2, 3 pair. The unit time of dead-zone length is received from clock selector A.
[23:16]	**DZL01**	**Dead-zone Length Register 0** These 8 bits determine the dead-zone length of channel 0, 1 pair. The unit time of dead-zone length is received from clock selector A.
[15:8]	**PRESCALE23**	**Prescale Register for Channel 2 & 3** Prescale output clock frequency = PCLK / (PRESCALE23 + 1). If PPR=0, then the prescale output clock will be stopped.
[7:0]	**PRESCALE01**	**Prescale Register for Channel 0 & 1** Prescale output clock frequency = PCLK / (PRESCALE01 + 1). If PPR=0, then the prescale output clock will be stopped.

PWM1 Pre-scale Register (PWM0_PPR)

Register	Offset	R/W	Description	Reset Value
PWM1_PPR	PWM1_BA+0x000	R/W	PWM1 Pre-scale Register	0x0000_0000

31	30	29	28	27	26	25	24
DZL67							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DZL45							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PRESCALE67							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PRESCALE45							

Bits	Description	
[31:24]	**DZL67**	**Dead-zone Length Register 3** These 8 bits determine the dead-zone length of channel 6, 7 pair. The unit time of dead-zone length is received from clock selector B.
[23:16]	**DZL45**	**Dead-zone Length Register 2** These 8 bits determine the dead-zone length of channel 4, 5 pair. The unit time of dead-zone length is received from clock selector B.
[15:8]	**PRESCALE67**	**Prescale Register for Channel 6 & 7** Prescale output clock frequency = PCLK / (PRESCALE67 + 1). If PPR=0, then the prescale output clock will be stopped.
[7:0]	**PRESCALE45**	**Prescale Register for Channel 4 & 5** Prescale output clock frequency = PCLK / (PRESCALE01 + 1). If PPR=0, then the prescale output clock will be stopped.

PWM0 Clock Select Register (PWM0_CSR)

Register	Offset	R/W	Description	Reset Value
PWM0_CSR	PWM0_BA+0x004	R/W	PWM0 Clock 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	**CLKSEL3**			**Reserved**	**CLKSEL2**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**CLKSEL1**			**Reserved**	**CLKSEL0**		

Bits	Description	
[31:15]	**Reserved**	Reserved.
[14:12]	**CLKSEL3**	**Channel 3 Clock Source Selection** Select PWM clock source for PWM timer channel 3 000 = Prescale output divided by 2. 001 = Prescale output divided by 4. 010 = Prescale output divided by 8. 011 = Prescale output divided by 16. 100 = Prescale output divided by 1.
[11]	**Reserved**	Reserved.
[10:8]	**CLKSEL2**	**Channel 2 Clock Source Selection** Select PWM clock source for PWM timer channel 2 (Table is the same as CH3)
[7]	**Reserved**	Reserved.
[6:4]	**CLKSEL1**	**Channel 1 Clock Source Selection** Select PWM clock source for PWM timer channel 1 (Table is the same as CH3)
[3]	**Reserved**	Reserved.
[2:0]	**CLKSEL0**	**Channel 0 Clock Source Selection** Select PWM clock source for PWM timer channel 0 (Table is the same as CH3)

PWM1 Clock Select Register (PWM1_CSR)

Register	Offset	R/W	Description	Reset Value
PWM1_CSR	PWM1_BA+0x004	R/W	PWM1 Clock 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	**CLKSEL7**			**Reserved**	**CLKSEL6**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**CLKSEL5**			**Reserved**	**CLKSEL4**		

Bits	Description	
[31:15]	**Reserved**	Reserved.
[14:12]	**CLKSEL7**	**Channel 7 Clock Source Selection** Select PWM clock source for PWM timer channel 7 000 = Prescale output divided by 2. 001 = Prescale output divided by 4. 010 = Prescale output divided by 8. 011 = Prescale output divided by 16. 100 = Prescale output divided by 1.
[11]	**Reserved**	Reserved.
[10:8]	**CLKSEL6**	**Channel 6 Clock Source Selection** Select PWM clock source for PWM timer channel 6 (Table is the same as CH7)
[7]	**Reserved**	Reserved.
[6:4]	**CLKSEL5**	**Channel 6 Clock Source Selection** Select PWM clock source for PWM timer channel 5 (Table is the same as CH7)
[3]	**Reserved**	Reserved.
[2:0]	**CLKSEL4**	**Channel 4 Clock Source Selection** Select PWM clock source for PWM timer channel 4 (Table is the same as CH7)

PWM0 Control Register (PWM0_PCR)

Register	Offset	R/W	Description	Reset Value
PWM0_PCR	PWM0_BA+0x008	R/W	PWM0 Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved				CH3MOD	CH3INV	Reserved	CH3EN
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CH2MOD	CH2INV	Reserved	CH2EN	CH1MOD	CH1INV	Reserved	CH1EN
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		DZEN23	DZEN01	CH0MOD	CH0INV	Reserved	CH0EN

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19]	**CH3MOD**	**Channel 3 Periodic/One-shot Mode** 0 = PWM counter of channel 3 operates as One-shot mode. 1 = PWM counter of channel 3 operates as Periodic mode.
[18]	**CH3INV**	**Channel 3 Inverter Switch** 0 = Inverter OFF. The output polarity of PWM channel 3 will be kept as usual. 1 = Inverter ON. The output polarity of PWM channel 3 will be inverted.
[17]	**Reserved**	Reserved.
[16]	**CH3EN**	**Channel 3 Enable Bit** 0 = Output of PWM channel 3 Disabled. 1 = Output of PWM channel 3 Enabled.
[15]	**CH2MOD**	**Channel 2 Periodic/One-shot Mode** 0 = PWM counter of channel 2 operates as One-shot mode. 1 = PWM counter of channel 2 operates as Periodic mode.
[14]	**CH2INV**	**Channel 2 Inverter Switch** 0 = Inverter OFF. The output polarity of PWM channel 2 will be kept as usual. 1 = Inverter ON. The output polarity of PWM channel 2 will be inverted.
[13]	**Reserved**	Reserved.
[12]	**CH2EN**	**Channel 2 Enable Bit** 0 = Output of PWM channel 2 Disabled. 1 = Output of PWM channel 2 Enabled.
[11]	**CH1MOD**	**Channel 1 Periodic/One-shot Mode** 0 = PWM counter of channel 1 operates as One-shot mode. 1 = PWM counter of channel 1 operates as Periodic mode.

[10]	**CH1INV**	**Channel 1 Inverter Switch** 0 = Inverter OFF. The output polarity of PWM channel 1 will be kept as usual. 1 = Inverter ON. The output polarity of PWM channel 1 will be inverted.
[9]	**Reserved**	Reserved.
[8]	**CH1EN**	**Channel 1 Enable Bit** 0 = Output of PWM channel 1 Disabled. 1 = Output of PWM channel 1 Enabled.
[7:6]	**Reserved**	Reserved.
[5]	**DZEN23**	**Dead-zone Generator 1 Enable Bit** 0 = Dead-Zone output of PWM channel 2, 3 Disabled. 1 = Dead-Zone output of PWM channel 2, 3 Enabled.
[4]	**DZEN01**	**Dead-zone Generator 0 Enable Bit** 0 = Dead-Zone output of PWM channel 0, 1 Disabled. 1 = Dead-Zone output of PWM channel 0, 1 Enabled.
[3]	**CH0MOD**	**Channel 0 Periodic/One-shot Mode** 0 = PWM counter of channel 0 operates as One-shot mode. 1 = PWM counter of channel 0 operates as Periodic mode.
[2]	**CH0INV**	**Channel 0 Inverter Switch** 0 = Inverter OFF. The output polarity of PWM channel 0 will be kept as usual 1 = Inverter ON. The output polarity of PWM channel 0 will be inverted
[1]	**Reserved**	Reserved.
[0]	**CH0EN**	**Channel 0 Enable Bit** 0 = Output of PWM channel 0 Disabled. 1 = Output of PWM channel 0 Enabled.

PWM1 Control Register (PWM1_PCR)

Register	Offset	R/W	Description	Reset Value
PWM1_PCR	PWM1_BA+0x008	R/W	PWM1 Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved				CH7MOD	CH7INV	Reserved	CH7EN
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CH6MOD	CH6INV	Reserved	CH6EN	CH5MOD	CH5INV	Reserved	CH5EN
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		DZEN67	DZEN45	CH4MOD	CH4INV	Reserved	CH4EN

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19]	**CH7MOD**	**Channel 7 Periodic/One-shot Mode** 0 = PWM counter of channel 7 operates as One-shot mode. 1 = PWM counter of channel 7 operates as Periodic mode.
[18]	**CH7INV**	**Channel 7 Inverter Switch** 0 = Inverter OFF. The output polarity of PWM channel 7 will be kept as usual. 1 = Inverter ON. The output polarity of PWM channel 7 will be inverted.
[17]	**Reserved**	Reserved.
[16]	**CH7EN**	**Channel 7 Enable Bit** 0 = Output of PWM channel 7 Disabled. 1 = Output of PWM channel 7 Enabled.
[15]	**CH6MOD**	**Channel 6 Periodic/One-shot Mode** 0 = PWM counter of channel 6 operates as One-shot mode. 1 = PWM counter of channel 6 operates as Periodic mode.
[14]	**CH6INV**	**Channel 6 Inverter Switch** 0 = Inverter OFF. The output polarity of PWM channel 6 will be kept as usual. 1 = Inverter ON. The output polarity of PWM channel 6 will be inverted.
[13]	**Reserved**	Reserved.
[12]	**CH6EN**	**Channel 6 Enable Bit** 0 = Output of PWM channel 6 Disabled. 1 = Output of PWM channel 6 Enabled.
[11]	**CH5MOD**	**Channel 5 Periodic/One-shot Mode** 0 = PWM counter of channel 5 operates as One-shot mode. 1 = PWM counter of channel 5 operates as Periodic mode.

[10]	**CH5INV**	**Channel 5 Inverter Switch** 0 = Inverter OFF. The output polarity of PWM channel 5 will be kept as usual. 1 = Inverter ON. The output polarity of PWM channel 5 will be inverted.
[9]	**Reserved**	Reserved.
[8]	**CH5EN**	**Channel 5 Enable Bit** 0 = Output of PWM channel 5 Disabled. 1 = Output of PWM channel 5 Enabled.
[7:6]	**Reserved**	Reserved.
[5]	**DZEN67**	**Dead-zone Generator 3 Enable Bit** 0 = Dead-Zone output of PWM channel 6, 7 Disabled. 1 = Dead-Zone output of PWM channel 6, 7 Enabled.
[4]	**DZEN45**	**Dead-zone Generator 2 Enable Bit** 0 = Dead-Zone output of PWM channel 4, 5 Disabled. 1 = Dead-Zone output of PWM channel 4, 5 Enabled.
[3]	**CH4MOD**	**Channel 4 Periodic/One-shot Mode** 0 = PWM counter of channel 4 operates as One-shot mode. 1 = PWM counter of channel 4 operates as Periodic mode.
[2]	**CH4INV**	**Channel 4 Inverter Switch** 0 = Inverter OFF. The output polarity of PWM channel 4 will be kept as usual. 1 = Inverter ON. The output polarity of PWM channel 4 will be inverted.
[1]	**Reserved**	Reserved.
[0]	**CH4EN**	**Channel 4 Enable Bit** 0 = Output of PWM channel 4 Disabled. 1 = Output of PWM channel 4 Enabled.

PWM Counter Register (PWM_CNR)

Register	Offset	R/W	Description	Reset Value
PWM0_CNR	PWM0_BA+0x00C	R/W	PWM0 Counter Register 0	0x0000_0000
PWM1_CNR	PWM0_BA+0x018	R/W	PWM0 Counter Register 1	0x0000_0000
PWM2_CNR	PWM0_BA+0x024	R/W	PWM0 Counter Register 2	0x0000_0000
PWM3_CNR	PWM0_BA+0x030	R/W	PWM0 Counter Register 3	0x0000_0000
PWM4_CNR	PWM1_BA+0x00C	R/W	PWM1 Counter Register 4	0x0000_0000
PWM5_CNR	PWM1_BA+0x018	R/W	PWM1 Counter Register 5	0x0000_0000
PWM6_CNR	PWM1_BA+0x024	R/W	PWM1 Counter Register 6	0x0000_0000
PWM7_CNR	PWM1_BA+0x030	R/W	PWM1 Counter Register 7	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CNR**	**PWM Counter Reload Value** The PWM period = CNR + 1, and if CNR is set to zero, PWM down-counting will be stopped. **Note:** Software can write a value to CNR at any time, and it will take effect in next PWM period.

PWM Comparator Register (PWM_CMR)

Register	Offset	R/W	Description	Reset Value
PWM0_CMR	PWM0_BA+0x010	R/W	PWM0 Comparator Register 0	0x0000_0000
PWM1_CMR	PWM0_BA+0x01C	R/W	PWM0 Comparator Register 1	0x0000_0000
PWM2_CMR	PWM0_BA+0x028	R/W	PWM0 Comparator Register 2	0x0000_0000
PWM3_CMR	PWM0_BA+0x034	R/W	PWM0 Comparator Register 3	0x0000_0000
PWM4_CMR	PWM1_BA+0x010	R/W	PWM1 Comparator Register 4	0x0000_0000
PWM5_CMR	PWM1_BA+0x01C	R/W	PWM1 Comparator Register 5	0x0000_0000
PWM6_CMR	PWM1_BA+0x028	R/W	PWM1 Comparator Register 6	0x0000_0000
PWM7_CMR	PWM1_BA+0x034	R/W	PWM1 Comparator Register 7	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CMR**	**PWM Comparator Register** PWM duty length follows the description below: CMR >= CNR : PWM output is always high. CMR < CNR : PWM output high for (CMR + 1) unit CMR = 0 : PWM output high for 1 unit. (Unit : 1 PWM clock cycle)

PWM Data Register (PWM_PDR)

Register	Offset	R/W	Description	Reset Value
PWM0_PDR	PWM0_BA+0x014	R	PWM0 Data Register 0	0x0000_0000
PWM1_PDR	PWM0_BA+0x020	R	PWM0 Data Register 1	0x0000_0000
PWM2_PDR	PWM0_BA+0x02C	R	PWM0 Data Register 2	0x0000_0000
PWM3_PDR	PWM0_BA+0x038	R	PWM0 Data Register 3	0x0000_0000
PWM4_PDR	PWM1_BA+0x014	R	PWM1 Data Register 4	0x0000_0000
PWM5_PDR	PWM1_BA+0x020	R	PWM1 Data Register 5	0x0000_0000
PWM6_PDR	PWM1_BA+0x02C	R	PWM1 Data Register 6	0x0000_0000
PWM7_PDR	PWM1_BA+0x038	R	PWM1 Data Register 7	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PDR**	**PWM Data Register** It indicates the current value of the PWM down-counter.

PWM0 Timer Interrupt Enable Register (PWM0_PIER)

Register	Offset	R/W	Description	Reset Value
PWM0_PIER	PWM0_BA+0x03C	R/W	PWM0 Timer 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				**PIER3**	**PIER2**	**PIER1**	**PIER0**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**PIER3**	**PWM Timer Channel 3 Interrupt Enable Bit** 0 = PWM interrupt function of channel 3 Disabled. 1 = PWM interrupt function of channel 3 Enabled.
[2]	**PIER2**	**PWM Timer Channel 2 Interrupt Enable Bit** 0 = PWM interrupt function of channel 2 Disabled. 1 = PWM interrupt function of channel 2 Enabled.
[1]	**PIER1**	**PWM Timer Channel 1 Interrupt Enable Bit** 0 = PWM interrupt function of channel 1 Disabled. 1 = PWM interrupt function of channel 1 Enabled.
[0]	**PIER0**	**PWM Timer Channel 0 Interrupt Enable Bit** 0 = PWM interrupt function of channel 0 Disabled. 1 = PWM interrupt function of channel 0 Enabled.

PWM1 Timer Interrupt Enable Register (PWM1_PIER)

Register	Offset	R/W	Description	Reset Value
PWM1_PIER	PWM1_BA+0x03C	R/W	PWM1 Timer 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				PIER7	PIER6	PIER5	PIER4

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**PIER7**	**PWM Timer Channel 7 Interrupt Enable Bit** 0 = PWM interrupt function of channel 7 Disabled. 1 = PWM interrupt function of channel 7 Enabled.
[2]	**PIER6**	**PWM Timer Channel 6 Interrupt Enable Bit** 0 = PWM interrupt function of channel 6 Disabled. 1 = PWM interrupt function of channel 6 Enabled.
[1]	**PIER5**	**PWM Timer Channel 5 Interrupt Enable Bit** 0 = PWM interrupt function of channel 5 Disabled. 1 = PWM interrupt function of channel 5 Enabled.
[0]	**PIER4**	**PWM Timer Channel 4 Interrupt Enable Bit** 0 = PWM interrupt function of channel 4 Disabled. 1 = PWM interrupt function of channel 4 Enabled.

PWM0 Timer Interrupt Indication Register (PWM0_PIIR)

Register	Offset	R/W	Description	Reset Value
PWM0_PIIR	PWM0_BA+0x040	R/W	PWM0 Timer Interrupt Indication Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				PIIR3	PIIR2	PIIR1	PIIR0

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**PIIR3**	**PWM Timer Channel 3 Interrupt Flag** This bit is set by hardware when PWM channel 3 counter reaches zero. **Note:** This bit is cleared by writing 1 to it.
[2]	**PIIR2**	**PWM Timer Channel 2 Interrupt Flag** This bit is set by hardware when PWM channel 2 counter reaches zero. **Note:** This bit is cleared by writing 1 to it.
[1]	**PIIR1**	**PWM Timer Channel 1 Interrupt Flag** This bit is set by hardware when PWM channel 1 counter reaches zero. **Note:** This bit is cleared by writing 1 to it.
[0]	**PIIR0**	**PWM Timer Channel 0 Interrupt Flag** This bit is set by hardware when PWM channel 0 counter reaches zero. **Note:** This bit is cleared by writing 1 to it.

PWM1 Timer Interrupt Indication Register (PWM1_PIIR)

Register	Offset	R/W	Description	Reset Value
PWM1_PIIR	PWM1_BA+0x040	R/W	PWM1 Timer Interrupt Indication Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				PIIR7	PIIR6	PIIR5	PIIR4

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**PIIR7**	**PWM Timer Channel 7 Interrupt Flag** This bit is set by hardware when PWM channel 7 counter reaches zero. **Note:** This bit is cleared by writing 1 to it.
[2]	**PIIR6**	**PWM Timer Channel 6 Interrupt Flag** This bit is set by hardware when PWM channel 6 counter reaches zero. **Note:** This bit is cleared by writing 1 to it.
[1]	**PIIR5**	**PWM Timer Channel 5 Interrupt Flag** This bit is set by hardware when PWM channel 5 counter reaches zero. **Note:** This bit is cleared by writing 1 to it.
[0]	**PIIR4**	**PWM Timer Channel 4 Interrupt Flag** This bit is set by hardware when PWM channel 4 counter reaches zero. **Note:** This bit is cleared by writing 1 to it.

6.11 Watchdog Timer (WDT)

6.11.1 Overview

The purpose of Watchdog Timer (WDT) is to perform a system reset when system runs into an unknown state. This prevents system from hanging for an infinite period of time. Besides, this Watchdog Timer supports the function to wake-up system from Idle/Power-down mode.

6.11.2 Features

- 20-bit free running up counter for WDT time-out interval
- Selectable time-out interval (2^4 ~ 2^{20}) and the time-out interval is 0.48828125 ms ~ 32 s if WDT_CLK = 32.768 kHz
- System kept in reset state for a period of (1 / WDT_CLK) * 63
- Supports selectable WDT reset delay period, including 1026、130、18 or 3 WDT_CLK reset delay period
- Supports to force WDT enabled after chip powered on or reset by setting WDTON (SYS_PWRON [3])
- Supports WDT time-out wake-up function only if WDT clock source is selected as LXT.

6.11.3 Block Diagram

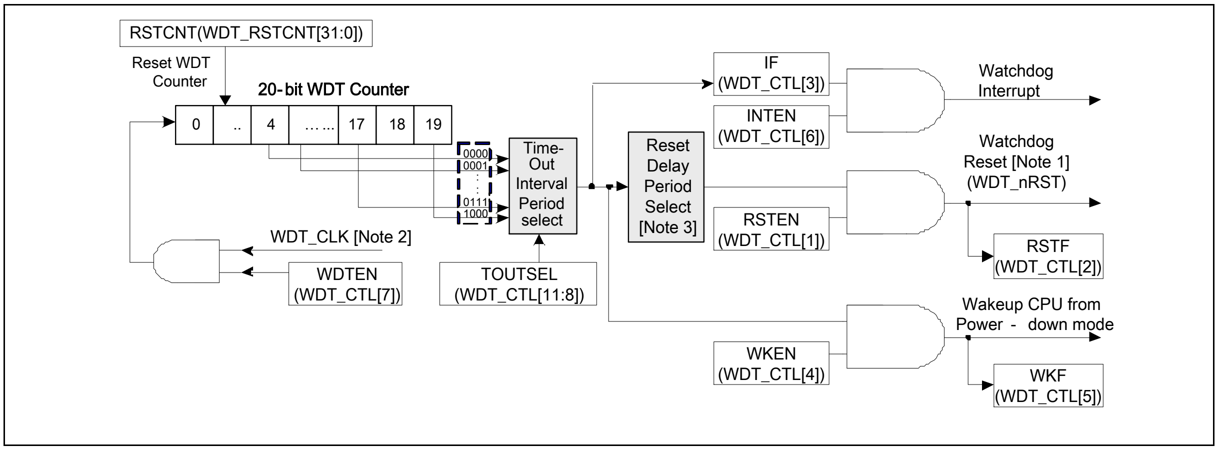


Figure 6.11-1 Watchdog Timer Block Diagram

Note 1: WDT resets CPU and lasts 63 WDT_CLK.

Note 2: Chip can be woken up by WDT time-out interrupt signal generated only if WDT clock source is selected to LXT.

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

6.11.4 Basic Configuration

- Clock source configuration
  - Enable WDT peripheral clock in WDTCKEN (CLK_PCLKEN0[0]).
- Reset Configuration
  - Reset WDT in WDTRST (SYS_APBIPRST0[0]).

The WDT clock control and block diagram are shown as Figure 6.11-2.

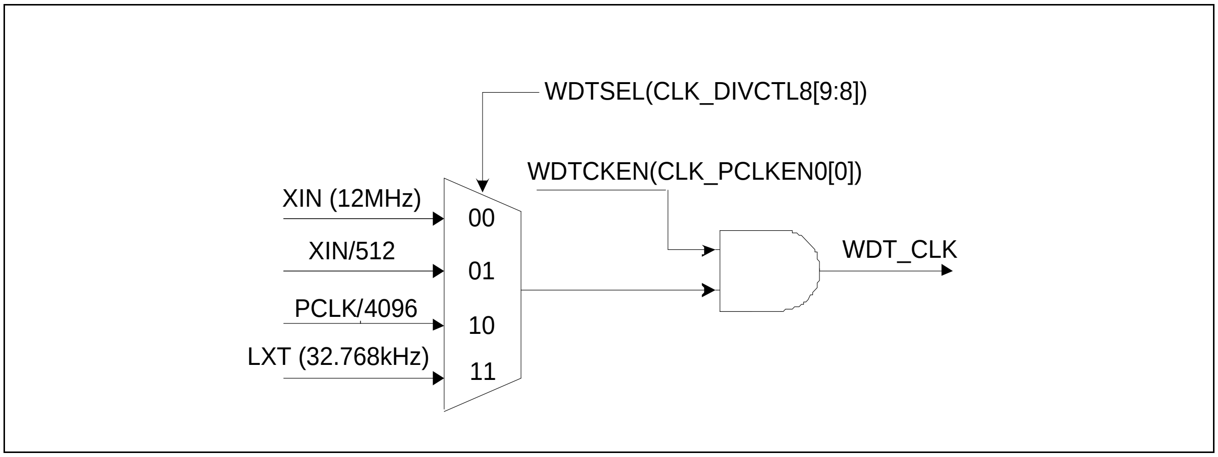


Figure 6.11-2 Watchdog Timer Clock Control

- Pin configuration

Group	Pin Name	GPIO	MFP
WDT	WDT_nRST	nRESET	MFP1

6.11.5 Functional Description

The WDT includes an 20-bit free running up counter with programmable time-out intervals. Table 6.11-1 shows the WDT time-out interval period selection and Figure 6.11-3 shows the WDT reset period timing.

6.11.5.1 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[10:8]) settings, WDT time-out interrupt will occur then WDT time-out interrupt flag IF (WDT_CTL[3]) will be set to 1 immediately. If INTEN (WDT_CTL[6]) is enabled, WDT time-out interrupt will inform to CPU.

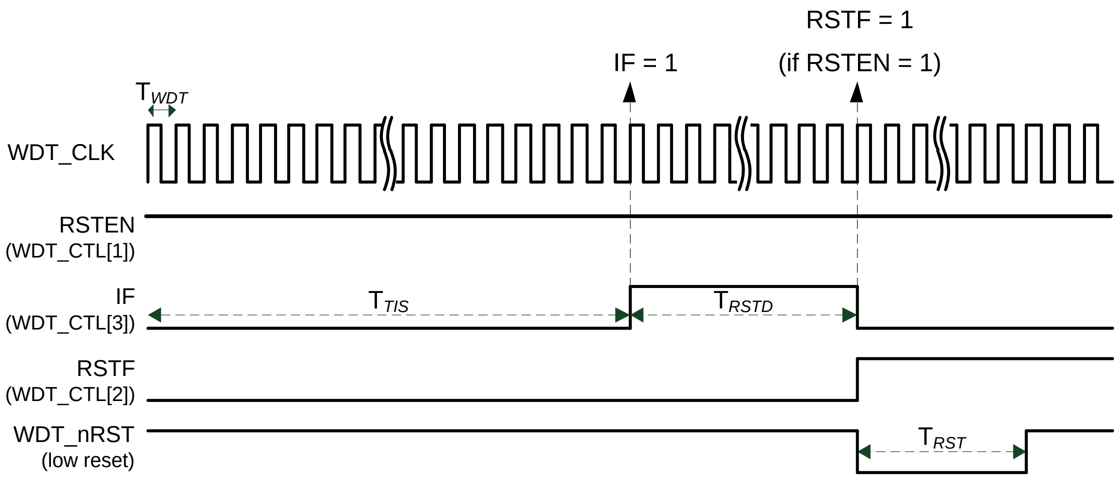
6.11.5.2 WDT Reset Delay Period and Reset System

There is a specified T_{RSTD} reset delay period follows the IF (WDT_CTL[3]) is setting to 1. User should write 0x5aa5 to WDT_RSTCNT[31:0] to reset the 20-bit WDT up counter value to avoid generate WDT time-out reset signal before the T_{RSTD} reset delay period expires. Moreover, user should set RSTDSEL (WDT_ALTCTL [1:0]) to select reset delay period to clear WDT counter. If the WDT up counter value has not been cleared after the specific T_{RSTD} delay period expires, the WDT control will set RSTF (WDT_CTL[2]) to 1 if RSTEN (WDT_CTL[1]) bit is enabled, then chip enters to reset state immediately. Refer to Figure 6.11-3, T_{RST} reset period will keep last 63 WDT clocks then chip restart executing program from reset vector (0x0000_0000). The RSTF (WDT_CTL[2]) will keep 1 after WDT time-out reset the chip, user can check RSTF (WDT_CTL[2]) by software to recognize the system has been reset by WDT time-out reset or not.

TOUTSEL	Time-Out Interval Period T_{TIS}	Reset Delay Period (T_{RSTD})

0000	$2^{4} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
0001	$2^{6} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
0010	$2^{8} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
0011	$2^{10} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
0100	$2^{12} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
0101	$2^{14} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
0110	$2^{16} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
0111	$2^{18} * T_{WDT}$	(3/18/130/1026) * T_{WDT}
1000	$2^{20} * T_{WDT}$	(3/18/130/1026) * T_{WDT}

Table 6.11-1 Watchdog Time-out Interval Period Selection



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

Figure 6.11-3 Watchdog Timer Time-out Interval and Reset Period Timing

6.11.5.3 WDT Wake-up

If WDT clock source is selected to 32 kHz, system can be waken-up from Power-down mode while WDT time-out interrupt signal is generated and WKEN (WDT_CTL[4]) enabled. Notice that user should set XTAL_EN (CLK_PMCON[0]) to enable crystal clock source before system enters power down mode because the system peripheral clock are disabled when system is power down mode. In the meanwhile, the WKF (WDT_CTL[5]) will set to 1 automatically, user can check WKF (WDT_CTL[5]) status by software to recognize the system has been waken-up by WDT time-out interrupt or not.

6.11.5.4 WDT ICE Debug

When ICE is connected to MCU, the 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.

6.11.6 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 = 0xB004_0000**				
WDT_CTL	WDT_BA+0x00	R/W	WDT Control Register	0x0000_0XXX
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

6.11.7 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_0XXX

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 Control (Write Protect)** 0 = ICE debug mode acknowledgement affects WDT counting. WDT up counter will be held while CPU is held by ICE. 1 = ICE debug mode acknowledgement Disabled. WDT up counter will keep going no matter CPU is held by ICE or not. **Note:** This bit is write protected. Refer to the REGWRPROT register.
[30]	**SYNC**	**WDT Enable Control SYNC Flag Indicator (Read Only)** If user execute 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:** Perform enable or disable WDTEN bit needs 2 * WDT_CLK period to become active.
[29:12]	**Reserved**	Reserved.

NUC980 SERIES TECHNICAL REFERENCE MANUAL

[11:8]	**TOUTSEL**	**WDT Time-out Interval Selection (Write Protect)** These three bits select the time-out interval period for the WDT. 0000 = 2^{4} *T_{WDT}. 0001 = 2^{6} *T_{WDT}. 0010 = 2^{8} *T_{WDT}. 0011 = 2^{10} *T_{WDT}. 0100 = 2^{12} *T_{WDT}. 0101 = 2^{14} *T_{WDT}. 0110 = 2^{16} *T_{WDT}. 0111 = 2^{18} *T_{WDT}. 1000 = 2^{20} *T_{WDT}. **Note:** This bit is write protected. Refer to the REGWRPROT register.
[7]	**WDTEN**	**WDT Enable Control (Write Protect)** 0 = WDT Disabled (This action will reset the internal up counter value). 1 = WDT Enabled. **Note1:** This bit is write protected. Refer to the SYS_REGLCTL register. **Note2:** If CWDTEN[2:0] (combined by Config0[31] and Config0[4:3]) bits is not configured to 111, this bit is forced as 1 and user cannot change this bit to 0.
[6]	**INTEN**	**WDT Time-out Interrupt Enable Control (Write Protect)** If this bit is enabled, the WDT time-out interrupt signal is generated and inform to CPU. 0 = WDT time-out interrupt Disabled. 1 = WDT time-out interrupt Enabled. **Note:** This bit is write protected. Refer to the REGWRPROT register.
[5]	**WKF**	**WDT Time-out Wake-up Flag** 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:** 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 REGWRPROT register. **Note 2:** Chip can be woken up by WDT time-out interrupt signal generated only if WDT clock source is selected to 32 kHz oscillator.
[3]	**IF**	**WDT Time-out Interrupt Flag** This bit will set to 1 while WDT up counter value reaches the selected WDT time-out interval 0 = WDT time-out interrupt did not occur. 1 = WDT time-out interrupt occurred. **Note:** This bit is cleared by writing 1 to it.

[2]	**RSTF**	**WDT Time-out Reset Flag** This bit indicates the system has been reset by WDT time-out reset or not. 0 = WDT time-out reset did not occur. 1 = WDT time-out reset occurred. **Note:** This bit is cleared by writing 1 to it.
[1]	**RSTEN**	**WDT Time-out Reset Enable Control (Write Protect)** Setting this bit will enable the WDT time-out reset function If the WDT up counter value has not been cleared after the specific WDT reset delay period expires. 0 = WDT time-out reset function Disabled. 1 = WDT time-out reset function Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[0]	**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:** Perform RSTCNT to reset counter needs 2 * WDT_CLK period to become active. **Note:** RSTCNT (WDT_RSTCNT[31:0]) bits are not write protected.

6.12 Window Watchdog Timer (WWDT)

6.12.1 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.

6.12.2 Features

- 6-bit down counter value (CNTDAT) and 6-bit compare value (CMPDAT) to make the WWDT time-out window period flexible.
- Supports 4-bit value (PSCSEL) to programmable maximum 11-bit prescale counter period of WWDT counter.

6.12.3 Block Diagram

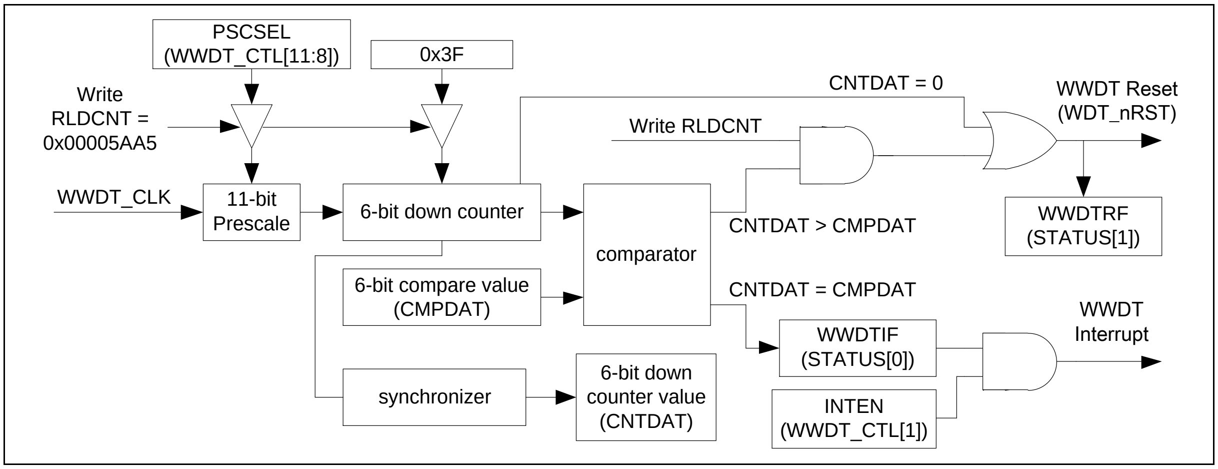


Figure 6.12-1 WWDT Block Diagram

6.12.4 Basic Configuration

- Clock source configuration
  - Enable WWDT peripheral clock in WWDTCKEN (CLK_PCLKEN0[1]).

The WWDT clock control and block diagram are shown as Figure 6.12-2.

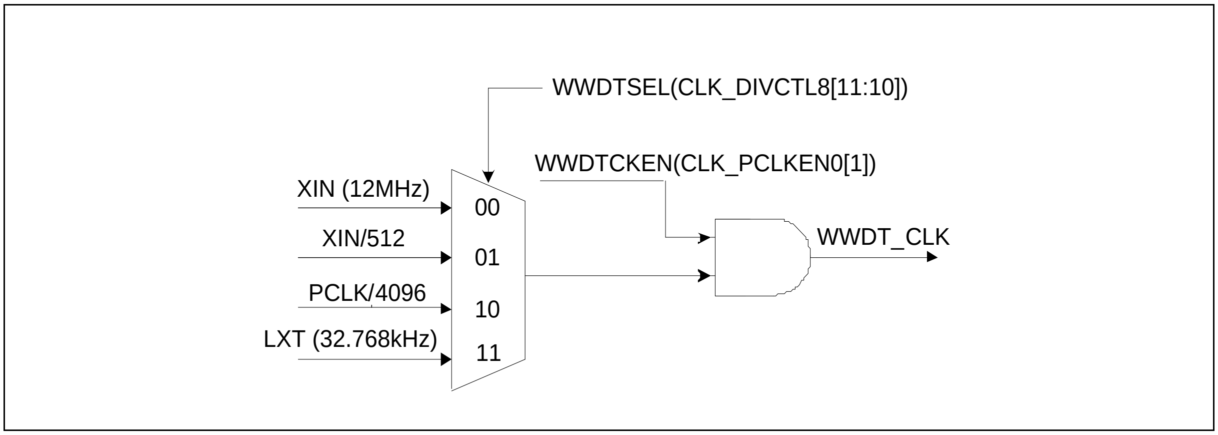


Figure 6.12-2 WWDT Clock Control

- Pin configuration

Group	Pin Name	GPIO	MFP
WDT	WDT_nRST	nRESET	MFP1

6.12.5 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 4096 (PCLK/4096), external 12 MHz oscillator or internal 32 kHz oscillator with a programmable 11-bit prescale counter value which is controlled by PSCSEL (WWDT_CTL[11:8]). Also, the correlate of PSCSEL (WWDT_CTL[11:8]) and prescale value are listed in Table 6.12-1

PSCSEL	Prescaler Value	Max. Time-Out Period	Max. Time-Out Interval (WWDT_CLK=32768 Hz)
0000	1	1 * 64 * T_{WWDT}	1.9 ms
0001	2	2 * 64 * T_{WWDT}	3.9 ms
0010	4	4 * 64 * T_{WWDT}	7.8 ms
0011	8	8 * 64 * T_{WWDT}	15.6 ms
0100	16	16 * 64 * T_{WWDT}	31.3 ms
0101	32	32 * 64 * T_{WWDT}	62.5 ms
0110	64	64 * 64 * T_{WWDT}	125 ms
0111	128	128 * 64 * T_{WWDT}	250 ms
1000	192	192 * 64 * T_{WWDT}	375 ms
1001	256	256 * 64 * T_{WWDT}	500 ms
1010	384	384 * 64 * T_{WWDT}	750 ms
1011	512	512 * 64 * T_{WWDT}	1 s
1100	768	768 * 64 * T_{WWDT}	1.5 s
1101	1024	1024 * 64 * T_{WWDT}	2 s
1110	1536	1536 * 64 * T_{WWDT}	3 s
1111	2048	2048 * 64 * T_{WWDT}	4 s

Table 6.12-1 Window Watchdog Prescaler Value Selection

6.12.5.1 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[0]), 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.

Please note that to avoid resetting the system while CPU clock is disabled, the WWDT counter will stop counting when CPU enters Idle/Power-down mode. After CPU enters normal mode, the WWDT counter will start down counting.

6.12.5.2 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.

6.12.5.3 WWDT Reset System

Figure 6.12-3 shows three cases of WWDT reset and reload behavior.

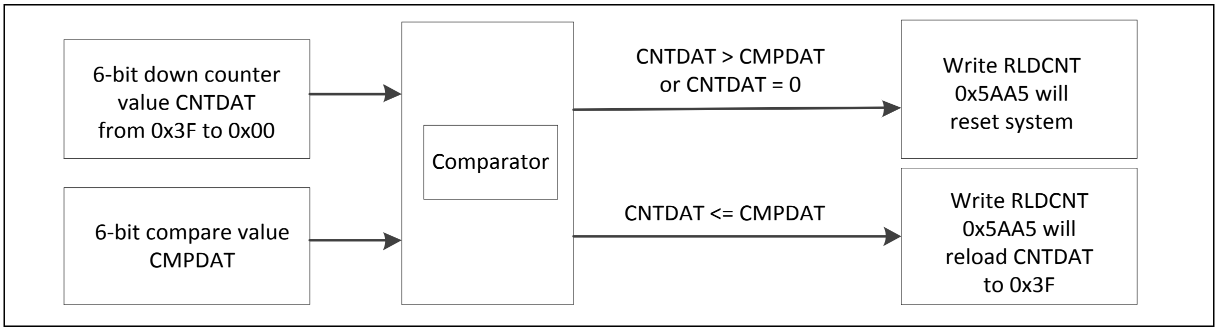


Figure 6.12-3 WWDT Reset and Reload Behavior

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 6.12-4.

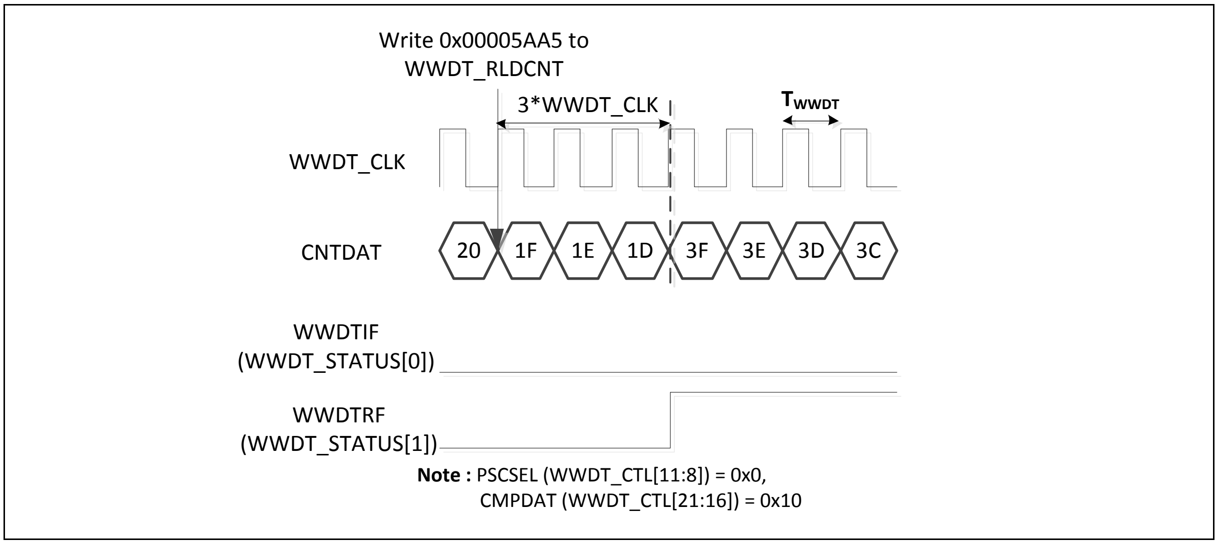


Figure 6.12-4 WWDT Reload Counter When CNTDAT > CMPDAT

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 6.12-5 shows the waveform of WWDT reload counter when CNTDAT < CMPDAT and Figure 6.12-6 shows WWDT generate reset system signal (WWDTRF) if user doesn’t write 0x00005AA5 to WWDT_RLDCNT before WWDT counter value reach to 0.

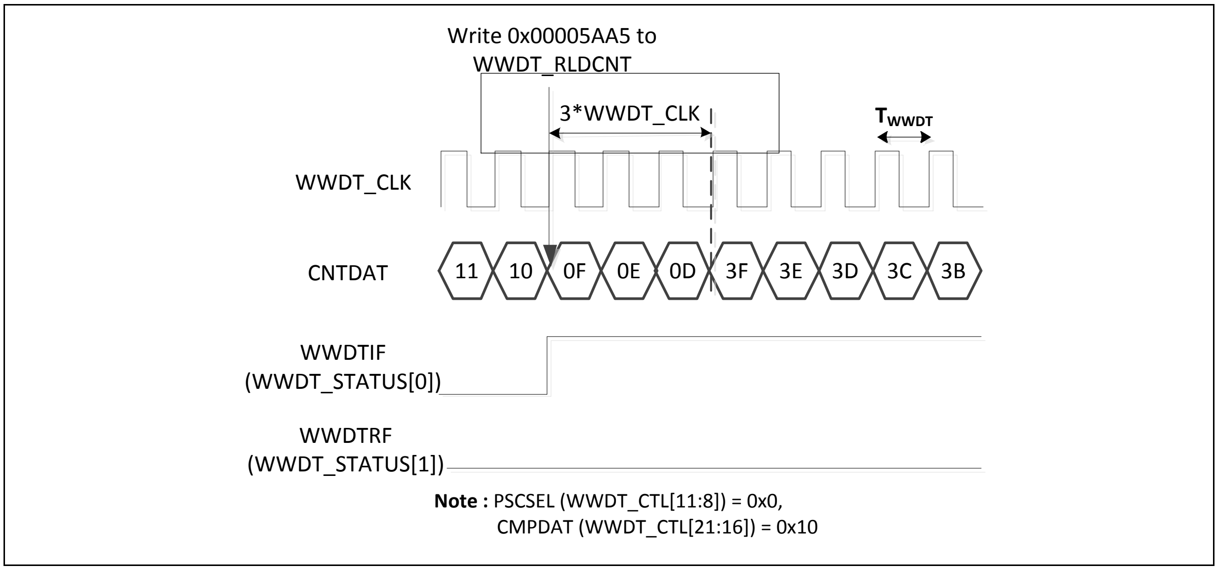


Figure 6.12-5 WWDT Reload Counter When CNTDAT < CMPDAT

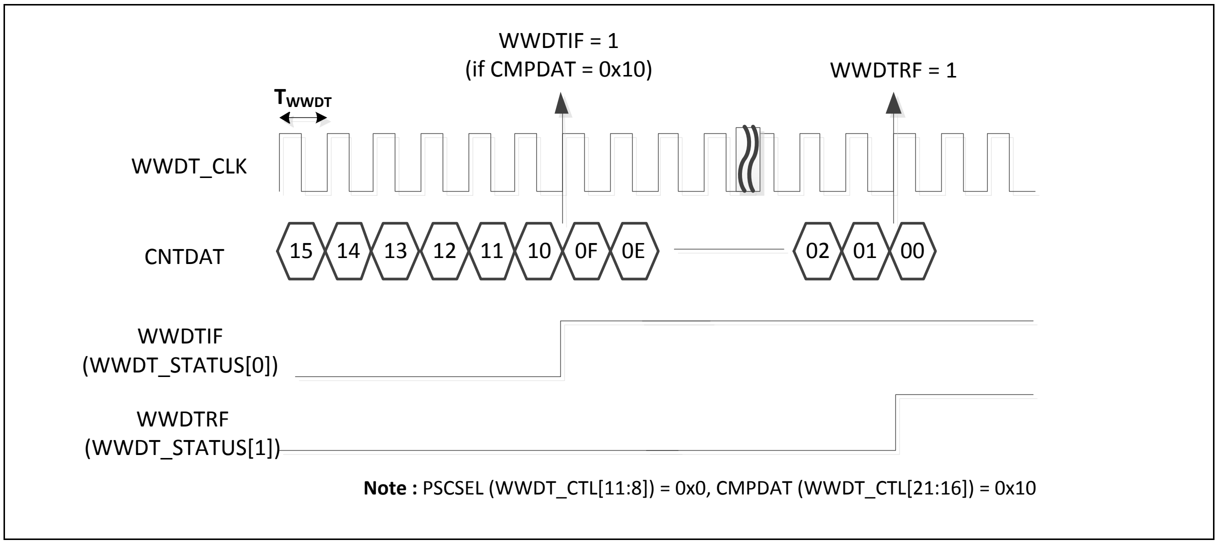


Figure 6.12-6 WWDT Interrupt and Reset Signals

6.12.5.4 WWDT Window Setting Limitation

When user writes 0x00005AA5 to WWDT_RLDCNT register to reload WWDT counter value to 0x3F, it needs 3 WWDT clocks to sync the reload command to actually perform reload action. Notice that if user set PSCSEL (WWDT_CTL[11:8]) to 0000, the counter prescale value should be as 1, and the CMPDAT (WWDT_CTL[21:16]) must be larger than 2. Otherwise, writing WWDT_RLDCNT register to reload WWDT counter value to 0x3F is unavailable, WWDTIF (WWDT_STATUS[0]) is generated, and WWDT reset system event always happened.

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 6.12-2 CMPDA Setting Limitation

6.12.5.5 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.

6.12.6 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 = 0xB004_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

6.12.7 Register Description

WWDT Reload Counter Register (WWDT_RLDCNT)

Register	Offset	R/W	Description	Reset Value
WWDT_RLDCNT	WWDT_BA+0x00	W	WWDT Reload Counter Register	0x0000_0000

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

Bits	Description	
[31:0]	**RLDCNT**	**WWDT Reload Counter Register** Writing 0x00005AA5 to this register will reload the WWDT counter value to 0x3F. **Note:** User can only write WWDT_RLDCNT register to reload WWDT counter value when current WWDT counter value between 0 and CMPDAT (WWDT_CTL[21:16]). If user writes WWDT_RLDCNT when current WWDT counter value is larger than CMPDAT, WWDT reset signal will generate immediately.

WWDT Control Register (WWDT_CTL)

Register	Offset	R/W	Description	Reset Value
WWDT_CTL	WWDT_BA+0x04	R/W	WWDT Control Register	0x003F_0800

Note: This register can be write only one time after chip is powered on or reset.

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

Bits	Description	
[31]	**ICEDEBUG**	**ICE Debug Mode Acknowledge Disable Control** 0 = ICE debug mode acknowledgement effects WWDT counting. WWDT down counter will be held while CPU is held by ICE. 1 = ICE debug mode acknowledgement Disabled. WWDT down counter will keep going no matter CPU is held by ICE or not.
[30:22]	**Reserved**	Reserved.
[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 * T_{WWDT}. 0001 = Pre-scale is 2; Max time-out period is 2 * 64 * T_{WWDT}. 0010 = Pre-scale is 4; Max time-out period is 4 * 64 * T_{WWDT}. 0011 = Pre-scale is 8; Max time-out period is 8 * 64 * T_{WWDT}. 0100 = Pre-scale is 16; Max time-out period is 16 * 64 * T_{WWDT}. 0101 = Pre-scale is 32; Max time-out period is 32 * 64 * T_{WWDT}. 0110 = Pre-scale is 64; Max time-out period is 64 * 64 * T_{WWDT}. 0111 = Pre-scale is 128; Max time-out period is 128 * 64 * T_{WWDT}. 1000 = Pre-scale is 192; Max time-out period is 192 * 64 * T_{WWDT}. 1001 = Pre-scale is 256; Max time-out period is 256 * 64 * T_{WWDT}. 1010 = Pre-scale is 384; Max time-out period is 384 * 64 * T_{WWDT}. 1011 = Pre-scale is 512; Max time-out period is 512 * 64 * T_{WWDT}.

		1100 = Pre-scale is 768; Max time-out period is 768 * 64 * T_{WWDT}. 1101 = Pre-scale is 1024; Max time-out period is 1024 * 64 * T_{WWDT}. 1110 = Pre-scale is 1536; Max time-out period is 1536 * 64 * T_{WWDT}. 1111 = Pre-scale is 2048; Max time-out period is 2048 * 64 * T_{WWDT}.
[7:2]	**Reserved**	Reserved.
[1]	**INTEN**	**WWDT Interrupt Enable Control Bit** If this bit is enabled, the WWDT counter compare match interrupt signal is generated and inform to CPU. 0 = WWDT counter compare match interrupt Disabled. 1 = WWDT counter compare match interrupt Enabled.
[0]	**WWDTEN**	**WWDT Enable Control Bit** Set this bit to enable WWDT counter counting. 0 = WWDT counter is stopped. 1 = WWDT counter is starting counting.

WWDT Status Register (WWDT_STATUS)

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

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

Bits	Description	
[31:2]	**Reserved**	Reserved.
[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.

6.13 Real Time Clock (RTC)

6.13.1 Overview

The Real Time Clock (RTC) controller provides the real time clock and calendar information. The clock source of RTC controller is from an external 32.768 kHz low-speed crystal which connected at pins X32_IN and X32_OUT (refer to pin Description). The RTC controller provides the real time clock (hour, minute, second) in RTC_TIME (RTC Time Loading Register) as well as calendar information (year, month, day) in RTC_CAL (RTC Calendar Loading Register). It also offers RTC alarm function that user can preset the alarm time in RTC_TALM (RTC Time Alarm Register) and alarm calendar in RTC_CALM (RTC Calendar Alarm Register). The data format of RTC time and calendar message are all expressed in BCD (Binary Coded Decimal) format.

The RTC controller supports periodic RTC Time Tick and Alarm Match interrupts. The periodic RTC 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 which are selected by RTC_TICK (RTC_TICK[2:0] Time Tick Register). When real time and calendar message in RTC_TIME and RTC_CAL are equal to alarm time and calendar settings in RTC_TALM and RTC_CALM, the ALMIF (RTC_INTSTS [0] RTC Alarm Interrupt Flag) is set to 1 and the RTC alarm interrupt signal is generated if the ALMIEN (RTC_INTEN [0] Alarm Interrupt Enable) is enabled.

Both RTC Time Tick and Alarm Match interrupt signal can cause chip to wake-up from Idle or Power-down mode if the corresponding interrupt enable bit (ALMIEN or TICKIEN) is set to 1 before chip enters Idle or Power-down mode.

Real Time Clock (RTC) block can operate with independent power supply (RTC_V_{DD}) while the system power is off.

6.13.2 Features

- Supports real time counter and calendar counter for RTC time and calendar check.
- Supports time (hour, minute, second) and calendar (year, month, day) alarm and alarm mask settings.
- Selectable 12-hour or 24-hour time scale.
- Supports Leap Year indication.
- Supports Day of the Week counter.
- Supports frequency compensation mechanism for 32.768 kHz clock source.
- All time and calendar message expressed in BCD format.
- Supports periodic RTC Time Tick interrupt with 8 period interval options 1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2 and 1 second.
- Supports RTC Time Tick and Alarm match interrupt.
- Supports chip wake-up from Idle or Power-down mode while alarm or relative alarm interrupt is generated.
- Supports 64 bytes spare registers to store user’s important information.

6.13.3 Block Diagram

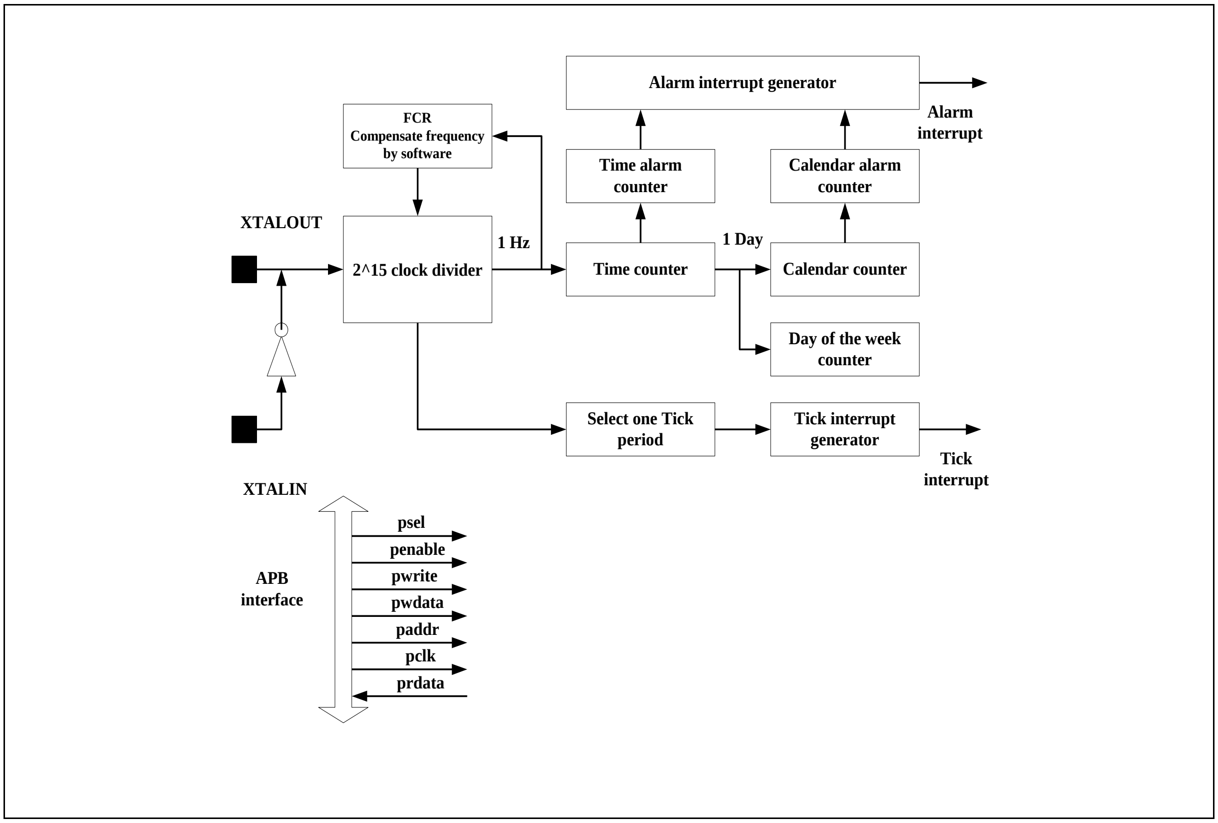


Figure 6.13-1 RTC Functional Block Diagram

6.13.4 Basic Configuration

6.13.4.1 RTC Basic Configuration

- Clock source Configuration

- Enable RTC peripheral clock in RTCCKEN (CLK_PCLKEN[2]).

- Reset Configuration

- It only can be reset by power on reset and watch-dog reset.

6.13.5 Functional Description

6.13.5.1 RTC Initiation

When RTC block is power on, programmer has to write a number (0xa5eb1357) to RTC_INIT to reset all logic. RTC_INIT act as hardware reset circuit. Once RTC_INIT has been set as 0xa5eb1357, user cannot reload any other value.

6.13.5.2 RTC write enable

Register RTC_RWEN bit 15~0 is RTC read /write password. It is used to avoid signal interference from system during system power off. RTC_RWEN bit 15~0 has to be set as 0xa965 before user want to write new data into all registers besides RTC_INIT. If user set RTC_RWEN as 0xa965, RWENF will be raised high. Then user can feel free to write data into register. RWENF will keep high for a short period (about 24ms) and it will be pull low by internal state machine automatically.

6.13.5.3 Frequency Compensation

The RTC_FREQADJ allows software control digital compensation of a 32.768 kHz crystal oscillator. User can utilize a frequency counter to measure RTC tick output in pin PH.4 and PI.3 during manufacture, and store the value in Flash memory for retrieval when the product is first power on.

Frequency Compensation	**Example 1**	**Frequency Counter Measurement: 32773.65Hz** **Integer Part: 32773 => RTC_FREQADJ[11:8] = 0xc** **Fraction Part: 0.65 X 60 = 39(0x27) => RTC_FREQADJ[5:0]=0x27**
	Example 2	Frequency counter measurement: 32765.27Hz Integer part: 32765=> RTC_FREQADJ[11:8] = 0x4 Fraction part: 0.27 X 60 = 16.2(0x10) => RTC_FREQADJ[5:0] = 0x10

Table 6.13-1 Frequency Compenstation Example

6.13.5.4 Time and Calendar counter

RTC_TIME and RTC_CAL are used to read the time and calendar. RTC_TALM and RTC_CALM are used as alarm. They are all BCD counters.

6.13.5.5 12/24 hour Time Scale Selection

The 12/24 hour time scale selection decided by 24HEN (RTC_TIMEFMT[0]).

24HEN (RTC_TIMEFMT[0])=1	24HEN (RTC_TIMEFMT[0])=0	24HEN (RTC_TIMEFMT[0])=1	24HEN (RTC_TIMEFMT[0])=0
24-hour time scale	12-hour time scale	24-hour time scale	12-hour time scale
00	12(AM12)	12	32(PM12)
01	01(AM01)	13	21(PM01)
02	02(AM02)	14	22(PM02)
03	03(AM03)	15	23(PM03)
04	04(AM04)	16	24(PM04)
05	05(AM05)	17	25(PM05)
06	06(AM06)	18	26(PM06)
07	07(AM07)	19	27(PM07)
08	08(AM08)	20	28(PM08)
09	09(AM09)	21	29(PM09)
10	10(AM10)	22	30(PM10)
11	11(AM11)	23	31(PM11)

Table 6.13-2 The 12/24 hour time scale selection table

6.13.5.6 Day of the Week Counter

Count from Sunday to Saturday

6.13.5.7 Tick Time Interrupt

RTC block use a counter to calibrate the tick time count value. When the value in counter reaches zero, RTC will issue an interrupt.

6.13.5.8 RTC Register Property

When system power is off but RTC power is on, data stored in RTC registers will not be lost except RTC_TSSR, RTC_INTEN and RTC_INTSTS. Because of difference between RTC clock and system clock, every time user write new data to any one register, the register will be updated until 2 RTC clock later (60us).

In addition, user must be aware that RTC block does not check whether loaded data is out of bounds. RTC does not check rationality between RTC_WEEKDAY and RTC_CLR either.

Note:

RTC_TALM, RTC_CALM, RTC_TIME and RTC_CAL are all BCD counter, but RTC_FREQADJ is not a BCD counter.

Programmer must be aware that the RTC block does not check whether the loaded value is reasonable. For example, Load RTC_CAL as 201a (year), 13 (month), 00 (day), or RTC_CAL does not match with RTC_WEEKDAY, etc.

In RTC_TIME and RTC_TALM, only 2 BCD digits are used to express “year”. We assume 2 BCD digits of XY denote 20XY, but not 19XY or 21XY.

6.13.6 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 = 0xB004_1000**				
RTC_INIT	RTC_BA+0x000	R/W	RTC Initiation Register	0x0000_0000
RTC_RWEN	RTC_BA+0x004	R/W	RTC Access Enable Register	0x0000_0000
RTC_FREQADJ	RTC_BA+0x008	R/W	RTC Frequency Compensation Register	0x0000_0700
RTC_TIME	RTC_BA+0x00C	R/W	RTC Time Counter Register	0x0000_0000
RTC_CAL	RTC_BA+0x010	R/W	RTC Calendar Counter Register	0x0005_0101
RTC_TIMEFMT	RTC_BA+0x014	R/W	RTC Time Format Selection Register	0x0000_0001
RTC_WEEKDAY	RTC_BA+0x018	R/W	RTC Day of the Week Register	0x0000_0006
RTC_TALM	RTC_BA+0x01C	R/W	RTC Time Alarm Register	0x0000_0000
RTC_CALM	RTC_BA+0x020	R/W	RTC Calendar Alarm Register	0x0000_0000
RTC_LEAPYEAR	RTC_BA+0x024	R	RTC Leap Year Indicator Register	0x0000_0000
RTC_INTEN	RTC_BA+0x028	R/W	RTC Interrupt Enable Register	0x0000_0000
RTC_INTSTS	RTC_BA+0x02C	R/W	RTC Interrupt Status Register	0x0000_0000
RTC_TICK	RTC_BA+0x030	R/W	RTC Time Tick Register	0x0000_0000
RTC_PWRCTL	RTC_BA+0x034	R/W	RTC Power Control Register	0x0000_7000
RTC_PWRCNT	RTC_BA+0x038	R	RTC Power Control Counter Register	0x0000_0000
RTC_CLKCTL	RTC_BA+0x03C	R/W	RTC 32.768 kHz Clock Control Register	0x0000_0001
RTC_SPR0	RTC_BA+0x040	R/W	RTC Spare Register 0	0x0000_0000
RTC_SPR1	RTC_BA+0x044	R/W	RTC Spare Register 1	0x0000_0000
RTC_SPR2	RTC_BA+0x048	R/W	RTC Spare Register 2	0x0000_0000
RTC_SPR3	RTC_BA+0x04C	R/W	RTC Spare Register 3	0x0000_0000
RTC_SPR4	RTC_BA+0x050	R/W	RTC Spare Register 4	0x0000_0000
RTC_SPR5	RTC_BA+0x054	R/W	RTC Spare Register 5	0x0000_0000
RTC_SPR6	RTC_BA+0x058	R/W	RTC Spare Register 6	0x0000_0000
RTC_SPR7	RTC_BA+0x05C	R/W	RTC Spare Register 7	0x0000_0000
RTC_SPR8	RTC_BA+0x060	R/W	RTC Spare Register 8	0x0000_0000
RTC_SPR9	RTC_BA+0x064	R/W	RTC Spare Register 9	0x0000_0000
RTC_SPR10	RTC_BA+0x068	R/W	RTC Spare Register 10	0x0000_0000
RTC_SPR11	RTC_BA+0x06C	R/W	RTC Spare Register 11	0x0000_0000
RTC_SPR12	RTC_BA+0x070	R/W	RTC Spare Register 12	0x0000_0000

RTC_SPR13	RTC_BA+0x074	R/W	RTC Spare Register 13	0x0000_0000
RTC_SPR14	RTC_BA+0x078	R/W	RTC Spare Register 14	0x0000_0000
RTC_SPR15	RTC_BA+0x07C	R/W	RTC Spare Register 15	0x0000_0000

6.13.7 Register Description

RTC Initiation Register (RTC_INIT)

Register	Offset	R/W	Description	Reset Value
RTC_INIT	RTC_BA+0x000	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**	**RTC Initiation** After RTC block is powered on, RTC is at reset state. User has to write a number (0x a5eb1357) to INIT to make RTC leaving reset state. Once the INIT is written as 0xa5eb1357, the RTC will be in un-reset state permanently. The INIT is a write-only field and read value will be always “0”.
[0]	**INIT/Active**	**RTC Active Status (Read Only)** 0 = RTC is at reset state. 1 = RTC is at normal active state.

RTC Access Enable Register (RTC_RWEN)

Register	Offset	R/W	Description	Reset Value
RTC_RWEN	RTC_BA+0x004	R/W	RTC Access Enable Register	0x0000_0000

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

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**RWENF**	**RTC Register Access Enable Flag (Read Only)** 0 = RTC register read/write Disabled. 1 = RTC register read/write Enabled. **Note:** This bit will be set after RWENPASSWD (RTC_RWEN[15:0]) register is load a 0xA965, and will be cleared automatically after 1024 RTC clock.
[15:0]	**RWENPASSWD**	**RTC Register Access Enable Password (Write Only)** Writing 0xA965 to this field will enable RTC access and keep 1024 RTC clock.

RTC Frequency Compensation Register (RTC_FREQADJ)

Register	Offset	R/W	Description	Reset Value
RTC_FREQADJ	RTC_BA+0x008	R/W	RTC Frequency Compensation Register	0x0000_0700

31	30	29	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	
[31:12]	**Reserved**	Reserved.
[11:8]	**INTEGER**	**Integer Part** 0000 = Integer part of detected value is 32761. 0001 = Integer part of detected value is 32762. 0010 = Integer part of detected value is 32763. 0011 = Integer part of detected value is 32764. 0100 = Integer part of detected value is 32765. 0101 = Integer part of detected value is 32766. 0110 = Integer part of detected value is 32767. 0111 = Integer part of detected value is 32768. 1000 = Integer part of detected value is 32769. 1001 = Integer part of detected value is 32770. 1010 = Integer part of detected value is 32771. 1011 = Integer part of detected value is 32772. 1100 = Integer part of detected value is 32773.
[7:6]	**Reserved**	Reserved.
[5:0]	**FRACTION**	**Fraction Part** Formula: FRACTION = (fraction part of detected value) X 60. **Note:** Digit in FCR must be expressed as hexadecimal number.

RTC Time Counter Register (RTC_TIME)

Register	Offset	R/W	Description	Reset Value
RTC_TIME	RTC_BA+0x00C	R/W	RTC Time Counter Register	0x0000_0000

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

Bits	Description	
[31:22]	**Reserved**	Reserved.
[21:20]	**TENHOUR**	10 Hour Time Digit (0 ~ 2)
[19:16]	**HOUR**	1 Hour Time Digit (0 ~ 9)
[15]	**Reserved**	Reserved.
[14:12]	**TENMINUTE**	10 Min Time Digit (0 ~ 5)
[11:8]	**MINUTE**	1 Min Time Digit (0 ~ 9)
[7]	**Reserved**	Reserved.
[6:4]	**TENSECOND**	10 Sec Time Digit (0 ~ 5)
[3:0]	**SECOND**	1 Sec Time Digit (0 ~ 9)

Note: RTC_TIME is a BCD digit counter and RTC will not check loaded data.

RTC Calendar Counter Register (RTC_CAL)

Register	Offset	R/W	Description	Reset Value
RTC_CAL	RTC_BA+0x010	R/W	RTC Calendar Counter Register	0x0005_0101

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			**TENMONTH**	**MONTH**			
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]	**TENMONTH**	10-Month Calendar Digit (0 ~ 1)
[11:8]	**MONTH**	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: RTC_CAL is a BCD digit counter and RTC will not check loaded data.

RTC Time Format Selection Register (RTC_TIMEFMT)

Register	Offset	R/W	Description	Reset Value
RTC_TIMEFMT	RTC_BA+0x014	R/W	RTC Time Format 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 Mode Selection** It indicate that TLR and TAR are in 24-hour mode or 12-hour mode 0 = 12-hour time format with am and pm indication selected. 1 = 24-hour time format selected.

RTC Day of the Week Register (RTC_WEEKDAY)

Register	Offset	R/W	Description	Reset Value
RTC_WEEKDAY	RTC_BA+0x018	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** 0x0 = Sunday. 0x1 = Monday. 0x2 = Tuesday. 0x3 = Wednesday. 0x4 = Thursday. 0x5 = Friday. 0x6 = Saturday. Others = Reserved.

RTC Time Alarm Register (RTC_TALM)

Register	Offset	R/W	Description	Reset Value
RTC_TALM	RTC_BA+0x01C	R/W	RTC Time Alarm Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	**HRALM_MSK**	**MINALM_MSK**	**SECALM_MSK**	**Reserved**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		**TENHOUR**		**HOUR**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	**TENMINUTE**			**MINUTE**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**TENSECOND**			**SECOND**			

Bits	Description	
[31]	**Reserved**	Reserved.
[30]	**HRALM_MSK**	**Hour Alarm Mask** This bit controls if TENHOUR (RTC_TALM[21:20] and HOUR (RTC_TALM[19:16]) could trigger RTC timer alarm. 0 = TENHOUR (RTC_TALM[21:20] and HOUR (RTC_TALM[19:16]) could trigger RTC time alarm. 1 = TENHOUR (RTC_TALM[21:20] and HOUR (RTC_TALM[19:16]) couldn't trigger RTC time alarm.
[29]	**MINALM_MSK**	**Minute Alarm Mask** This bit controls if TENMINUTE (RTC_TALM[14:12] and MINUTE (RTC_TALM[11:8]) could trigger RTC timer alarm. 0 = TENMINUTE (RTC_TALM[14:12] and MINUTE (RTC_TALM[11:8]) could trigger RTC time alarm. 1 = TENMINUTE (RTC_TALM[14:12] and MINUTE (RTC_TALM[11:8]) couldn't trigger RTC time alarm.
[28]	**SECALM_MSK**	**Second Alarm Mask** This bit controls if TENSECOND (RTC_TALM[6:4] and SECOND (RTC_TALM[3:0]) could trigger RTC timer alarm. 0 = TENSECOND (RTC_TALM[6:4] and SECOND (RTC_TALM[3:0]) could trigger RTC time alarm. 1 = TENSECOND (RTC_TALM[6:4] and SECOND (RTC_TALM[3:0]) couldn't trigger RTC time alarm.
[27:22]	**Reserved**	Reserved.
[21:20]	**TENHOUR**	10 Hour Time Digit (0 ~ 2)
[19:16]	**HOUR**	1 Hour Time Digit (0 ~ 9)
[15]	**Reserved**	Reserved.
[14:12]	**TENMINUTE**	10 Min Time Digit (0 ~ 5)

[11:8]	**MINUTE**	1 Min Time Digit (0 ~ 9)
[7]	**Reserved**	Reserved.
[6:4]	**TENSECOND**	10 Sec Time Digit (0 ~ 5)
[3:0]	**SECOND**	1 Sec Time Digit (0 ~ 9)

Note 1: RTC_TALM is a BCD digit counter and RTC will not check loaded data.

Note 2: Set all MSK bits high would disable calendar alarm functionality.

RTC Calendar Alarm Register (RTC_CALM)

Register	Offset	R/W	Description	Reset Value
RTC_CALM	RTC_BA+0x020	R/W	RTC Calendar Alarm Register	0x0000_0000

31	30	29	28	27	26	25	24
WKDALM_MSK	YRALM_MSK	MONALM_MSK	DAYALM_MSK	Reserved	WEEKDAY		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TENYEAR				YEAR			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			TENMONTH	MONTH			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		TENDAY		DAY			

Bits	Description	
[31]	**WKDALM_MSK**	**Day of Week Alarm Mask** This bit controls if WEEKDAY (RTC_CALM[26:24]) could trigger RTC timer alarm. 0 = WEEKDAY (RTC_CALM[26:24]) could trigger RTC time alarm. 1 = WEEKDAY (RTC_CALM[26:24]) couldn't trigger RTC time alarm.
[30]	**YRALM_MSK**	**Year Alarm Mask** This bit controls if TENYEAR (RTC_CALM[23:20]) and YEAR (RTC_CALM[19:16]) could trigger RTC timer alarm. 0 = TENYEAR (RTC_CALM[23:20]) and YEAR (RTC_CALM[19:16]) could trigger RTC time alarm. 1 = TENYEAR (RTC_CALM[23:20]) and YEAR (RTC_CALM[19:16]) couldn't trigger RTC time alarm.
[29]	**MONALM_MSK**	**Month Alarm Mask** This bit controls if TENMONTH (RTC_CALM[12]) and MONTH (RTC_CALM[11:8]) could trigger RTC timer alarm. 0 = TENMONTH (RTC_CALM[12]) and MONTH (RTC_CALM[11:8]) could trigger RTC time alarm. 1 = TENMONTH (RTC_CALM[12]) and MONTH (RTC_CALM[11:8]) couldn't trigger RTC time alarm.
[28]	**DAYALM_MSK**	**Day Alarm Mask** This bit controls if TENDAY (RTC_CALM[5:4]) and DAY (RTC_CALM[3:0]) could trigger RTC timer alarm. 0 = TENDAY (RTC_CALM[5:4]) and DAY (RTC_CALM[3:0]) could trigger RTC time alarm. 1 = TENDAY (RTC_CALM[5:4]) and DAY (RTC_CALM[3:0]) couldn't trigger RTC time alarm.
[27]	**Reserved**	Reserved.

[26:24]	**WEEKDAY**	**Day of the Week** 000 = Sunday. 001 = Monday. 010 = Tuesday. 011 = Wednesday. 100 = Thursday. 101 = Friday. 110 = Saturday. 111 = Reserved.
[23:20]	**TENYEAR**	10-Year Calendar Digit (0 ~ 9)
[19:16]	**YEAR**	1-Year Calendar Digit (0 ~ 9)
[15:13]	**Reserved**	Reserved.
[12]	**TENMONTH**	10-Month Calendar Digit (0 ~ 1)
[11:8]	**MONTH**	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_CALM is a BCD digit counter and RTC will not check loaded data.

Note 2: Set all MSK bits high would disable calendar alarm functionality.

RTC Leap Year Indicator Register (RTC_LEAPYEAR)

Register	Offset	R/W	Description	Reset Value
RTC_LEAPYEAR	RTC_BA+0x024	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 Indicator (Read Only)** 0 = It indicates that this year is not a leap year. 1 = It indicates that this year is leap year.

RTC Interrupt Enable Register (RTC_INTEN)

Register	Offset	R/W	Description	Reset Value
RTC_INTEN	RTC_BA+0x028	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			**RELALMIEN**	**Reserved**	**WAKEUPIEN**	**TICKIEN**	**ALMIEN**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**RELALMIEN**	**Relative Alarm Interrupt Enable Bit** 0 = RTC Relative Alarm interrupt Disabled. 1 = RTC Relative Alarm interrupt Enabled.
[3]	**Reserved**	Reserved.
[2]	**WAKEUPIEN**	**Wakeup Interrupt Enable Bit** 0 = RTC Power Down wakeup interrupt Disabled. 1 = RTC Power Down wakeup interrupt Enabled.
[1]	**TICKIEN**	**Tick Interrupt Enable Bit** 0 = RTC Time Tick Interrupt and counter Disabled. 1 = RTC Time Tick Interrupt and counter Enabled.
[0]	**ALMIEN**	**Alarm Interrupt Enable Bit** 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+0x02C	R/W	RTC Interrupt Status Register	0x0000_0000

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

Bits	Description	
[31]	**REGWRBUSY**	**Register Write Operation Busy** 0 = The new register write operation is acceptable. 1 = The last write operation is in progress and new register write operation prohibited.
[30:5]	**Reserved**	Reserved.
[4]	**RELALMINT**	**Relative Alarm Interrupt Status** This bit high indicates the related timer have counted down to zero. User can write 1 to clear this bit. 0 = Related timer didn’t count down to zero. 1 = Related timer counts down to zero.
[3]	**Reserved**	Reserved.
[2]	**WAKEUPINT**	**Wakeup Interrupt Status** This bit indicates the RTC generates a wakeup event to wakeup system from power down mode. In RTC, the wakeup source includes the RTC alarm and RTC related alarm. User can write 1 to clear this bit. 0 = RTC event to wakeup system has never occurred. 1 = RTC event to wakeup system occurred.
[1]	**TICKINT**	**RTC Time Tick Interrupt Indication REGISTER** This bit indicates the RTC timer tick value configured in RTC_TICK has reached. User can write 1 to clear this bit. 0 = RTC timer tick value configured in RTC_TICK didn’t reach. 1 = RTC timer tick value configured in RTC_TICK has reached.

[0]	**ALMINT**	**RTC Alarm Interrupt Indication REGISTER** This bit indicates the RTC_TIME and RTC_CAL counter have counted to value configured in RTC_TALM and RTC_CALM. User can write 1 to clear this bit. 0 = It indicates that alarm interrupt has never occurred. 1 = It indicates that RTC_TIME and RTC_CAL counter have counted to a specified time configured in RTC_TALM and RTC_CALM.

RTC Time Tick Register (RTC_TICK)

Register	Offset	R/W	Description	Reset Value
RTC_TICK	RTC_BA+0x030	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					**TTR**		

Bits	Description	
[2:0]	**TTR**	**RTC Tick Time Interrupt Request Interval** The TTR [2:0] is used to select tick time interrupt request interval. The period of tick time interrupt is as follow: 000 = 1 second. 001 = 1/2 second. 010 = 1/4 second. 011 = 1/8 second. 100 = 1/16 second. 101 = 1/32 second. 110 = 1/64 second. 111 = 1/128 second.

RTC Power Control Register (RTC_PWRCTL)

Register	Offset	R/W	Description	Reset Value
RTC_PWRCTL	RTC_BA+0x034	R/W	RTC Power Control Register	0x0000_7000

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

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:16]	**RELALM_TIME**	**Relative Alarm Time** This field defines the relative alarm time period by unit second. The maximum value is 12’d1800.
[15:5]	**Reserved**	Reserved.
[4]	**REL_ALArm_EN**	**Relative Alarm Function Enable Bit** Set this bit high would enable the relative alarm function. When the relative alarm condition met, RTC would set RELALMINT (RTC_INTSTS[4]) interrupt status to high. 0 = Relative alarm function Disabled. 1 = Relative alarm function Enabled.
[3]	**ALArm_EN**	**Alarm Function Enable Bit** Set this bit high would enable the alarm function. When the alarm condition met, RTC would set ALMINT (RTC_INTSTS[0]) interrupt status to high. 0 = Alarm function Disabled. 1 = Alarm function Enabled.
[2:0]	**Reserved**	Reserved.

RTC Power Control Counter Register (RTC_PWRCNT)

Register	Offset	R/W	Description	Reset Value
RTC_PWRCNT	RTC_BA+0x038	R	RTC Power Control 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				**RELArm_CNT**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RELArm_CNT							

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:0]	**RELArm_CNT**	**Relative Alarm Counter Current Value (Read Only)** This field shows the current value of relative alarm counter.

RTC 32.768 kHz Clock Control Register (RTC_CLKCTL)

Register	Offset	R/W	Description	Reset Value
RTC_CLKCTL	RTC_BA+0x03C	R/W	RTC 32.768 kHz Clock 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						**CLKMOD**	**CLKEN**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**CLKMOD**	**32.768 kHz Clock Mode Selection** 0 = The 32.768 kHz clock macro is operating in strong mode (8uA). 1 = The 32.768 kHz clock macro is operating in weak mode (5uA).
[0]	**CLKEN**	**32.768 kHz Clock Enable Bit** 0 = The 32.768 kHz clock macro Disabled. 1 = The 32.768 kHz clock macro Enabled.

RTC Spare Register (RTC_SPRn, n = 0, 1, .., 15)

Register	Offset	R/W	Description	Reset Value
RTC_SPR0	RTC_BA+0x040	R/W	RTC Spare Register 0	0x0000_0000
RTC_SPR1	RTC_BA+0x044	R/W	RTC Spare Register 1	0x0000_0000
RTC_SPR2	RTC_BA+0x048	R/W	RTC Spare Register 2	0x0000_0000
RTC_SPR3	RTC_BA+0x04C	R/W	RTC Spare Register 3	0x0000_0000
RTC_SPR4	RTC_BA+0x050	R/W	RTC Spare Register 4	0x0000_0000
RTC_SPR5	RTC_BA+0x054	R/W	RTC Spare Register 5	0x0000_0000
RTC_SPR6	RTC_BA+0x058	R/W	RTC Spare Register 6	0x0000_0000
RTC_SPR7	RTC_BA+0x05C	R/W	RTC Spare Register 7	0x0000_0000
RTC_SPR8	RTC_BA+0x060	R/W	RTC Spare Register 8	0x0000_0000
RTC_SPR9	RTC_BA+0x064	R/W	RTC Spare Register 9	0x0000_0000
RTC_SPR10	RTC_BA+0x068	R/W	RTC Spare Register 10	0x0000_0000
RTC_SPR11	RTC_BA+0x06C	R/W	RTC Spare Register 11	0x0000_0000
RTC_SPR12	RTC_BA+0x070	R/W	RTC Spare Register 12	0x0000_0000
RTC_SPR13	RTC_BA+0x074	R/W	RTC Spare Register 13	0x0000_0000
RTC_SPR14	RTC_BA+0x078	R/W	RTC Spare Register 14	0x0000_0000
RTC_SPR15	RTC_BA+0x07C	R/W	RTC Spare Register 15	0x0000_0000

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

Bits	Description	
[31:0]	**DATA**	**Data** This register is used to keep information written by user.

6.14 UART Interface Controller (UART)

6.14.1 Overview

The chip provides ten 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, LIN and RS-485 function modes and auto-baud rate measuring function.

6.14.2 Features

- Full-duplex asynchronous communications
- Separates receive and transmit 16/16 bytes entry FIFO for data payloads
- Supports hardware auto-flow control
- Programmable receiver buffer trigger level
- Supports programmable baud rate generator for each channel individually
- Supports nCTS, incoming data, Received Data FIFO reached threshold and RS-485 Address Match (AAD mode) wake-up function
- Supports 8-bit receiver buffer time-out detection function
- Programmable transmitting data delay time between the last stop and the next start bit by setting DLY (UART_TOUT [15:8])
- Supports Auto-Baud Rate measurement and baud rate compensation function
- Support 9600 bps for UART_CLK is selected LXT.
- Supports break error, frame error, parity error and receive/transmit buffer overflow detection function
- Fully programmable serial-interface characteristics
- Programmable number of data bit, 5-, 6-, 7-, 8- bit character
- Programmable parity bit, even, odd, no parity or stick parity bit generation and detection
- Programmable stop bit, 1, 1.5, or 2 stop bit generation
- Supports IrDA SIR function mode
- Supports for 3/16 bit duration for normal mode
- Supports LIN function mode (Only UART1 /UART2 with LIN function)
- Supports LIN master/slave mode
- Supports programmable break generation function for transmitter
- Supports break detection function for receiver
- Supports RS-485 function mode
- Supports RS-485 9-bit mode
- Supports hardware or software enables to program nRTS pin to control RS-485

transmission direction

- Supports PDMA transfer function

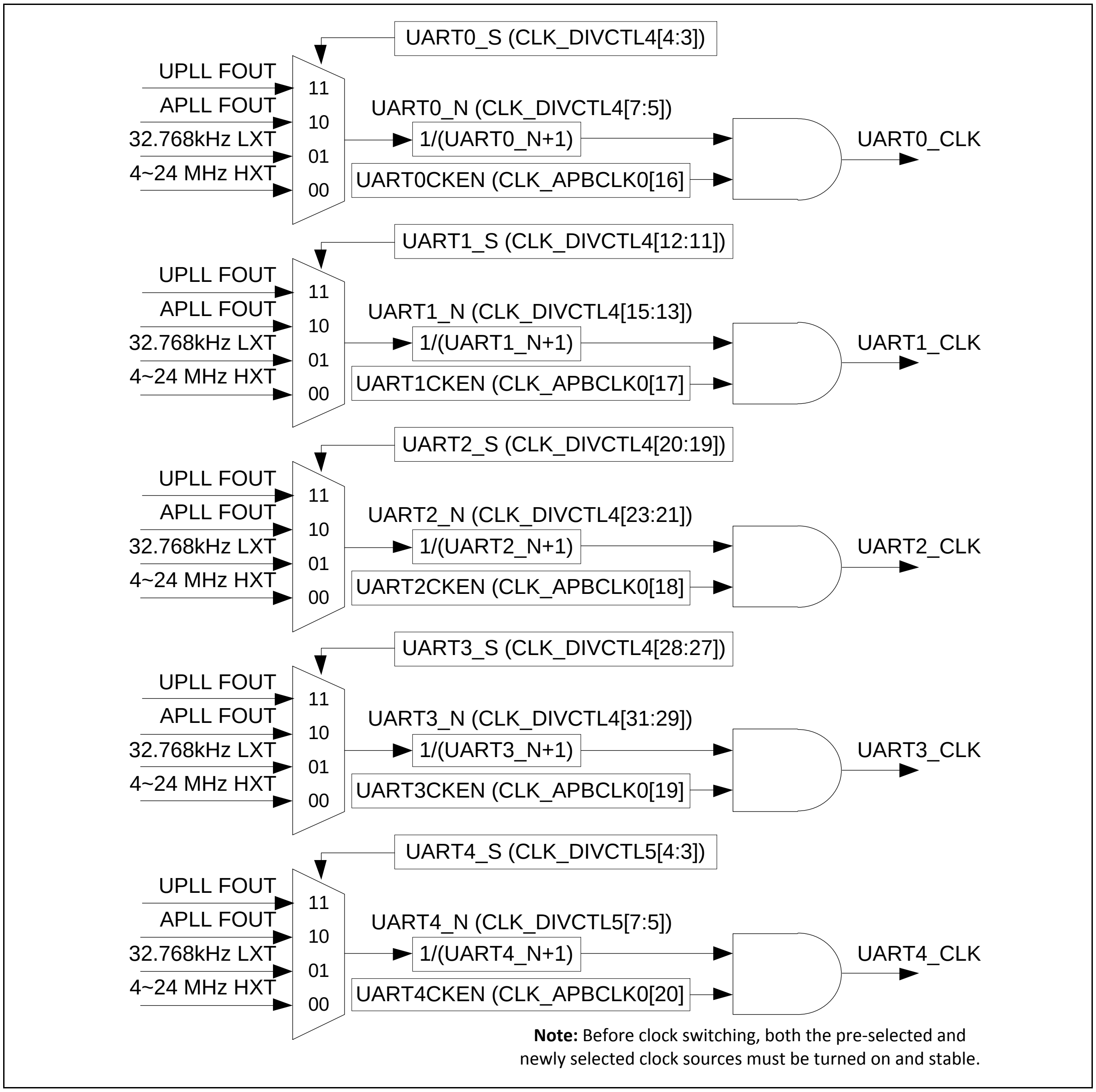
UART Feature	UART0	UART1/ UART2	UART3~UART9
FIFO	16 Bytes	16 Bytes	16 Bytes
Auto Flow Control (CTS/RTS)	-	√	√
IrDA	√	√	√
LIN	-	√	-
RS-485 Function Mode	√	√	√
nCTS Wake-up	-	√	√
Imcoming Data Wake-up	√	√	√
Received Data FIFO reached threshold Wake-up	√	√	√
RS-485 Address Match (AAD mode) Wake-up	√	√	√
Auto-Baud Rate Measurement	√	√	√
STOP Bit Length	1, 1.5, 2 bit	1, 1.5, 2 bit	1, 1.5, 2 bit
Word Length	5, 6, 7, 8 bits	5, 6, 7, 8 bits	5, 6, 7, 8 bits
Even / Odd Parity	√	√	√
Stick Bit	√	√	√
	Note: √= Supported		

Table 6.14-1 NUC980 Series UART Features

6.14.3 Block Diagram

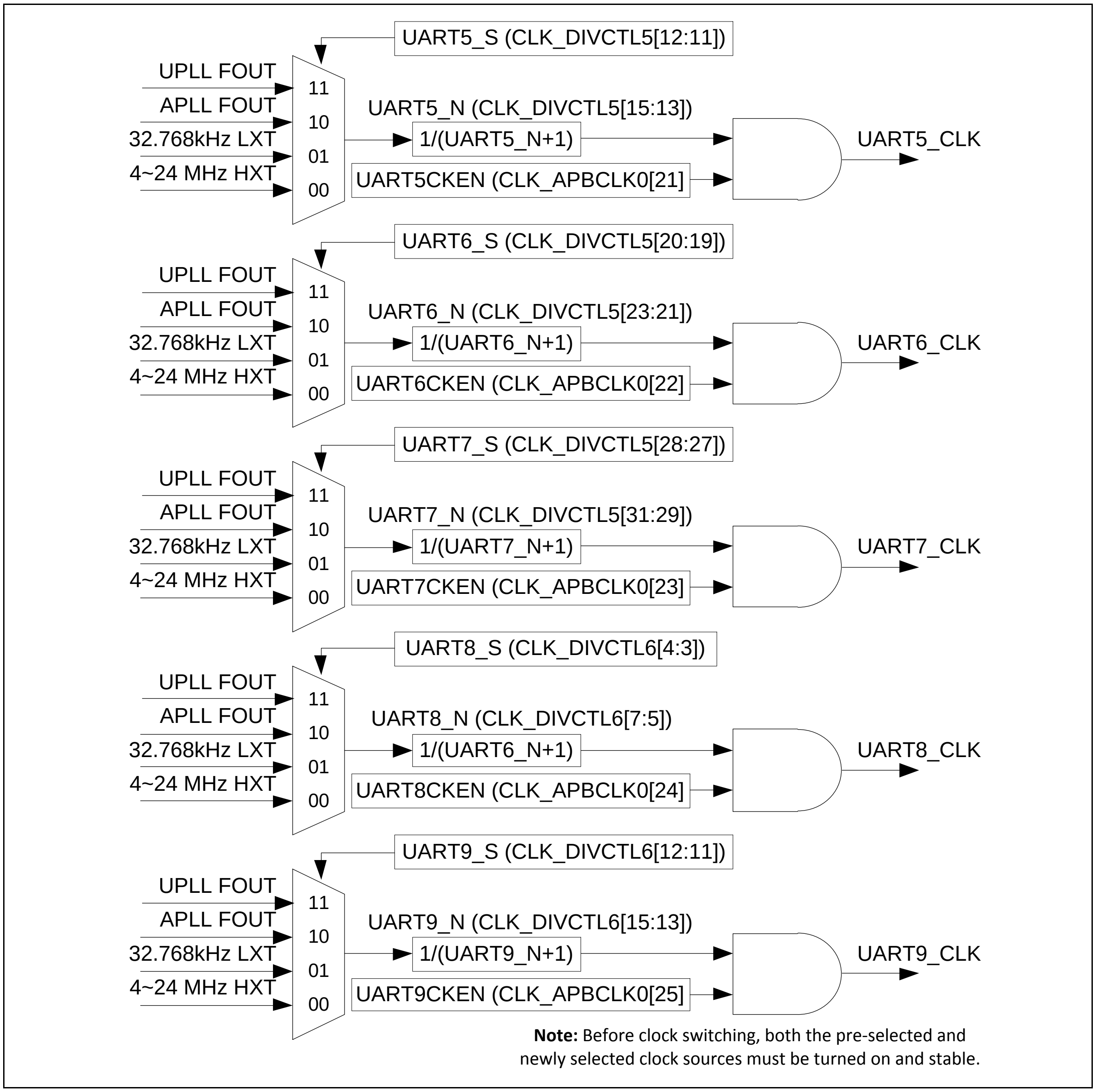
The UART clock control and block diagram are shown in Figure 6.14-1, Figure 6.14-2 and Figure 6.14-3 respectively.

Note: The frequency of UARTx_CLK should not be greater than 30 times HCLK.


UART0_S (CLK_DIVCTL4[4:3])
UPLL FOUT
APLL FOUT
32.768kHz LXT
4~24 MHz HXT
11
10
01
00
UART0_N (CLK_DIVCTL4[7:5])
1/(UART0_N+1)
UART0CKEN (CLK_APBCLK0[16]
UART0_CLK
UART1_S (CLK_DIVCTL4[12:11])
UPLL FOUT
APLL FOUT
32.768kHz LXT
4~24 MHz HXT
11
10
01
00
UART1_N (CLK_DIVCTL4[15:13])
1/(UART1_N+1)
UART1CKEN (CLK_APBCLK0[17]
UART1_CLK
UART2_S (CLK_DIVCTL4[20:19])
UPLL FOUT
APLL FOUT
32.768kHz LXT
4~24 MHz HXT
11
10
01
00
UART2_N (CLK_DIVCTL4[23:21])
1/(UART2_N+1)
UART2CKEN (CLK_APBCLK0[18]
UART2_CLK
UART3_S (CLK_DIVCTL4[28:27])
UPLL FOUT
APLL FOUT
32.768kHz LXT
4~24 MHz HXT
11
10
01
00
UART3_N (CLK_DIVCTL4[31:29])
1/(UART3_N+1)
UART3CKEN (CLK_APBCLK0[19]
UART3_CLK
UART4_S (CLK_DIVCTL5[4:3])
UPLL FOUT
APLL FOUT
32.768kHz LXT
4~24 MHz HXT
11
10
01
00
UART4_N (CLK_DIVCTL5[7:5])
1/(UART4_N+1)
UART4CKEN (CLK_APBCLK0[20]
UART4_CLK

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

Figure 6.14-1 UART0-UART4 Clock Control Diagram


UART5_S (CLK_DIVCTL5[12:11])
UPLL FOUT
APLL FOUT
32.768kHz LXT
4~24 MHz HXT
11
10
01
00
UART5_N (CLK_DIVCTL5[15:13])
1/(UART5_N+1)
UART5CKEN (CLK_APBCLK0[21]
UART5_CLK
UART6_S (CLK_DIVCTL5[20:19])
UPLL FOUT
APLL FOUT
32.768kHz LXT
4~24 MHz HXT
11
10
01
00
UART6_N (CLK_DIVCTL5[23:21])
1/(UART6_N+1)
UART6CKEN (CLK_APBCLK0[22]
UART6_CLK
UART7_S (CLK_DIVCTL5[28:27])
UPLL FOUT
APLL FOUT
32.768kHz LXT
4~24 MHz HXT
11
10
01
00
UART7_N (CLK_DIVCTL5[31:29])
1/(UART7_N+1)
UART7CKEN (CLK_APBCLK0[23]
UART7_CLK
UART8_S (CLK_DIVCTL6[4:3])
UPLL FOUT
APLL FOUT
32.768kHz LXT
4~24 MHz HXT
11
10
01
00
UART8_N (CLK_DIVCTL6[7:5])
1/(UART6_N+1)
UART8CKEN (CLK_APBCLK0[24]
UART8_CLK
UART9_S (CLK_DIVCTL6[12:11])
UPLL FOUT
APLL FOUT
32.768kHz LXT
4~24 MHz HXT
11
10
01
00
UART9_N (CLK_DIVCTL6[15:13])
1/(UART9_N+1)
UART9CKEN (CLK_APBCLK0[25]
UART9_CLK

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

Figure 6.14-2 UART5-UART9 Clock Control Diagram

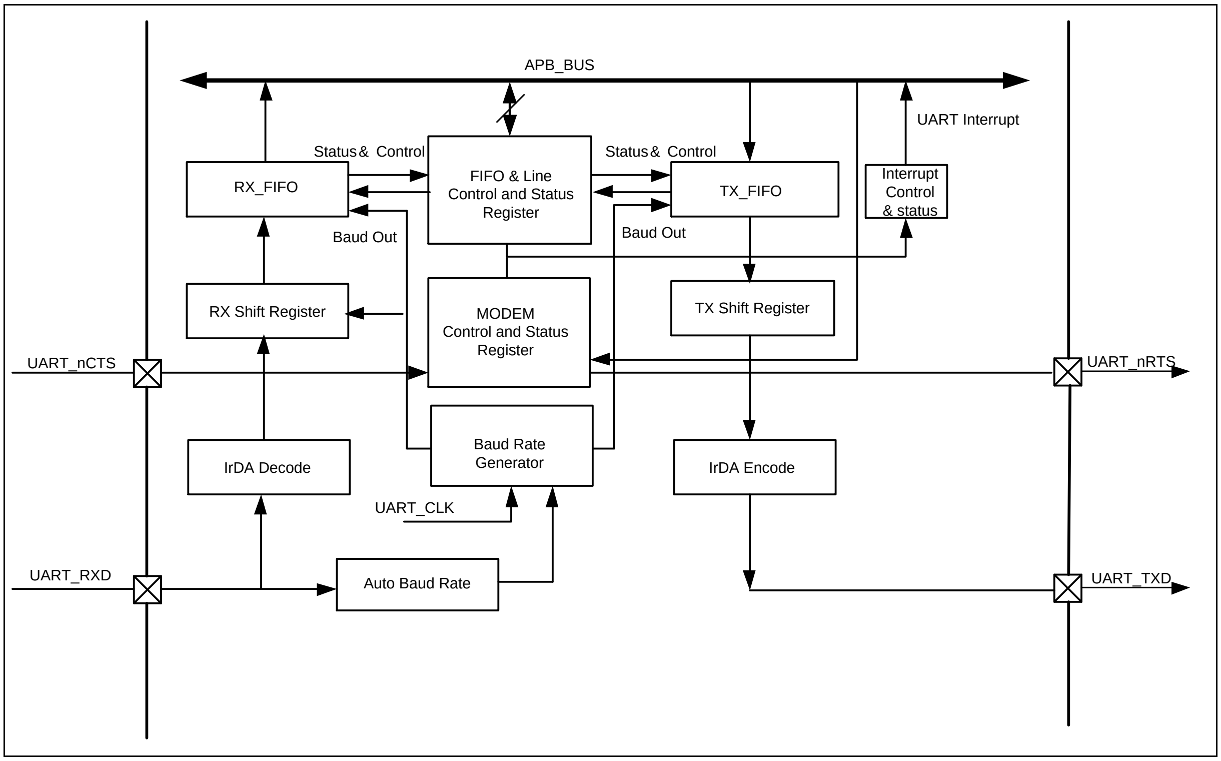


Figure 6.14-3 UART Block Diagram

Each block is described in detail as follows:

TX_FIFO

The transmitter is buffered with a 16 bytes FIFO to reduce the number of interrupts presented to the CPU.

RX_FIFO

The receiver is buffered with a 16 bytes FIFO (plus three error bits, BIF (UART_FIFOSTS[6]), FEF (UART_FIFOSTS[5]), PEF (UART_FIFOSTS[4])) to reduce the number of interrupts presented to the CPU.

TX Shift Register

This block is responsible for shifting out the transmitting data serially.

RX Shift Register

This block is responsible for shifting in the receiving data serially.

Modem Control and Status Register

This register controls the interface to the MODEM or data set (or a peripheral device emulating a MODEM).

Baud Rate Generator

Divide the external clock by the divisor to get the desired baud rate clock. Refer to baud rate equation.

IrDA Encode

This block is IrDA encoding control block.

IrDA Decode

This block is IrDA decoding control block.

FIFO & Line Control and Status Register

This field is register set that including the FIFO control register (UART_FIFO), FIFO status register (UART_FIFOSTS), and line control register (UART_LINE) for transmitter and receiver. The time-out register (UART_TOUT) identifies the condition of time-out interrupt.

Auto-Baud Rate Measurement

This block is responsible for auto-baud rate measurement.

Interrupt Control and Status Register

There are ten types of interrupts, Receive Data Available Interrupt (RDAINT), Transmit Holding Register Empty Interrupt (THERINT), Transmitter Empty Interrupt (TXENDINT), Receive Line Status Interrupt (parity error or framing error or break interrupt) (RLSINT), MODEM Status Interrupt (MODEMINT), Receiver Buffer Time-out Interrupt (RXTOINT), Buffer Error Interrupt (BUFERRINT), LIN Bus Interrupt (LININT), Wake-up Interrupt (WKINT) and Auto-Baud Rate Interrupt (ABRINT). Interrupt enable register (UART_INTEN) enable or disable the responding interrupt and interrupt status register (UART_INTSTS) identifying the occurrence of the responding interrupt.

Interrupt	Description
RDAINT	Receive Data Available Interrupt.
THERINT	Transmit Holding Register Empty Interrupt.
TXENDINT	Transmitter Empty Interrupt.
RLSINT	Receive Line Status Interrupt (parity error or frame error or break error).
MODEMINT	MODEM Status Interrupt.
RXTOINT	Receiver Buffer Time-out Interrupt.
BUFERRINT	Buffer Error Interrupt.
LININT	LIN Bus Interrupt.
WKINT	Wake-up Interrupt.
ABRINT	Auto-Baud Rate Interrupt.

Table 6.14-2 UART Interrupt

6.14.4 Basic Configuration

The basic configurations of UART0 are as follows:

- Clock Source Configuration
  - Select the source of UART0 peripheral clock on UART0_S (CLK_DIVCTL4[4:3]).
  - Select the clock divider number of UART0 peripheral clock on UART0_N (CLK_DIVCTL4[7:5]).
  - Enable UART0 peripheral clock in UART0CKEN (CLK_APBCLK0[16]).
- Reset Configuration
  - Reset UART0 controller in UART0RST (SYS_APBIPRST0[16]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
UART0	UART0_RXD	PF.11	MFP1

	UART0_TXD	PF.12	MFP1

The basic configurations of UART1 are as follows:

- Clock Source Configuration
  - Select the source of UART1 peripheral clock on UART1_S (CLK_DIVCTL4[12:11]).
  - Select the clock divider number of UART1 peripheral clock on UART1_N (CLK_DIVCTL4[15:13]).
  - Enable UART1 peripheral clock in UART1CKEN (CLK_APBCLK0[17]).
- Reset Configuration
  - Reset UART1 controller in UART1RST (SYS_APBIPRST0[17]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
UART1	UART1_RXD	PF.9	MFP2
		PA.0	MFP4
		PC.6	MFP7
	UART1_TXD	PF.10	MFP2
		PA.1	MFP4
		PC.5	MFP7
	UART1_nCTS	PF.7	MFP2
		PC.8	MFP7
	UART1_nRTS	PF.8	MFP2
		PC.7	MFP7

The basic configurations of UART2 are as follows:

- Clock Source Configuration
  - Select the source of UART2 peripheral clock on UART2_S (CLK_DIVCTL4[20:19]).
  - Select the clock divider number of UART2 peripheral clock on UART2_N (CLK_DIVCTL4[23:21]).
  - Enable UART2 peripheral clock in UART2CKEN (CLK_APBCLK0[18]).
- Reset Configuration
  - Reset UART2 controller in UART2RST (SYS_APBIPRST0[18]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
UART2	UART2_RXD	PA.9, PD.7, PG.0	MFP2
	UART2_TXD	PA.10, PD.6, PG.1	MFP2
	UART2_nCTS	PA.7, PB.0, PG.2	MFP2
	UART2_nRTS	PA.8, PG.3	MFP2

The basic configurations of UART3 are as follows:

- Clock Source Configuration
  - Select the source of UART3 peripheral clock on UART3_S (CLK_DIVCTL4[28:27]).
  - Select the clock divider number of UART3 peripheral clock on UART3_N (CLK_DIVCTL4[31:29]).
  - Enable UART3 peripheral clock in UART3CKEN (CLK_APBCLK0[19]).
- Reset Configuration
  - Reset UART3 controller in UART3RST (SYS_APBIPRST0[19]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
UART3	UART3_RXD	PB.10	MFP1
		PD.3	MFP2
		PC.4, PF.6	MFP5
	UART3_TXD	PB.9	MFP1
		PD.2	MFP2
		PB.13, PC.3, PF.7	MFP5
	UART3_nCTS	PB.12	MFP1
		PD.5	MFP2
		PF.4	MFP5
	UART3_nRTS	PB.11	MFP1
		PD.4	MFP2
		PF.5	MFP5

The basic configurations of UART4 are as follows:

- Clock Source Configuration
  - Select the source of UART4 peripheral clock on UART4_S (CLK_DIVCTL5[4:3]).
  - Select the clock divider number of UART4 peripheral clock on UART4_N (CLK_DIVCTL5[7:5]).
  - Enable UART4 peripheral clock in UART4CKEN (CLK_APBCLK0[20]).
- Reset Configuration
  - Reset UART4 controller in UART4RST (SYS_APBIPRST0[20]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
UART4	UART4_RXD	PD.13	MFP1
		PE.2	MFP5
		PC.10	MFP7

	UART4_TXD	PD.12	MFP1
		PE.3	MFP5
		PC.9	MFP7
	UART4_nCTS	PD.15	MFP1
		PE.0	MFP5
	UART4_nRTS	PD.14	MFP1
		PE.1	MFP5

The basic configurations of UART5 are as follows:

- Clock Source Configuration
  - Select the source of UART5 peripheral clock on UART5_S (CLK_DIVCTL5[12:11]).
  - Select the clock divider number of UART5 peripheral clock on UART5_N (CLK_DIVCTL5[15:13]).
  - Enable UART5 peripheral clock in UART5CKEN (CLK_APBCLK0[21]).
- Reset Configuration
  - Reset UART5 controller in UART5RST (SYS_APBIPRST0[21]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
UART5	UART5_RXD	PD.1, PG.6	MFP2
		PG.13	MFP5
	UART5_TXD	PD.0, PG.7	MFP2
		PG.14	MFP5
	UART5_nCTS	PG.4	MFP2
		PG.11	MFP5
	UART5_nRTS	PG.5	MFP2
		PG.12	MFP5

The basic configurations of UART6 are as follows:

- Clock Source Configuration
  - Select the source of UART6 peripheral clock on UART6_S (CLK_DIVCTL5[20:19]).
  - Select the clock divider number of UART6 peripheral clock on UART6_N (CLK_DIVCTL5[23:21]).
  - Enable UART6 peripheral clock in UART6CKEN (CLK_APBCLK0[22]).
- Reset Configuration
  - Reset UART6 controller in UART6RST (SYS_APBIPRST0[22]).
- Pin Configuration

Group	Pin Name	GPIO	MFP

UART6	UART6_RXD	PA.4	MFP1
		PD.11	MFP2
		PE.8	MFP5
	UART6_TXD	PA.5	MFP1
		PD.10	MFP2
		PE.9	MFP5
	UART6_nCTS	PA.2	MFP1
		PD.8	MFP2
	UART6_nRTS	PA.3	MFP1
		PD.9	MFP2

The basic configurations of UART7 are as follows:

- Clock Source Configuration
  - Select the source of UART7 peripheral clock on UART7_S (CLK_DIVCTL5[28:27]).
  - Select the clock divider number of UART7 peripheral clock on UART7_N (CLK_DIVCTL5[31:29]).
  - Enable UART7 peripheral clock in UART7CKEN (CLK_APBCLK0[23]).
- Reset Configuration
  - Reset UART7 controller in UART7RST (SYS_APBIPRST0[23]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
UART7	UART7_RXD	PC.2	MFP4
		PB.4, PF.2	MFP5
		PA.14	MFP6
	UART7_TXD	PC.1	MFP4
		PB.6, PF.3	MFP5
		PA.13	MFP6
	UART7_nCTS	PB.7, PF.0	MFP5
	UART7_nRTS	PB.5, PF.1	MFP5

The basic configurations of UART8 are as follows:

- Clock Source Configuration
  - Select the source of UART8 peripheral clock on UART8_S (CLK_DIVCTL6[4:3]).
  - Select the clock divider number of UART8 peripheral clock on UART8_N (CLK_DIVCTL6[7:5]).
  - Enable UART8 peripheral clock in UART8CKEN (CLK_APBCLK0[24]).
- Reset Configuration

- Reset UART8 controller in UART8RST (SYS_APBIPRST0[24]).

- Pin Configuration

Group	Pin Name	GPIO	MFP
UART8	UART8_RXD	PA.11	MFP2
		PC.0	MFP4
		PC.13	MFP7
	UART8_TXD	PA.12	MFP2
		PB.8	MFP4
		PC.12	MFP7
	UART8_nCTS	PG.9	MFP2
		PC.15	MFP7
	UART8_nRTS	PG.8	MFP2
		PC.14	MFP7

The basic configurations of UART9 are as follows:

- Clock Source Configuration
  - Select the source of UART9 peripheral clock on UART9_S (CLK_DIVCTL6[12:11]).
  - Select the clock divider number of UART9 peripheral clock on UART9_N (CLK_DIVCTL6[15:13]).
  - Enable UART9 peripheral clock in UART9CKEN (CLK_APBCLK0[25]).
- Reset Configuration
  - Reset UART9 controller in UART9RST (SYS_APBIPRST0[25]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
UART9	UART9_RXD	PE.10	MFP3
		PE.6	MFP5
		PB.3	MFP7
	UART9_TXD	PE.12	MFP3
		PE.7	MFP5
		PB.1	MFP7
	UART9_nCTS	PE.4	MFP5
	UART9_nRTS	PE.5	MFP5
		PB.2	MFP7

UART Interface Controller Pin description is shown in Table 6.14-3:

Pin	Type	Description

UARTx_TXD	Output	UARTx transmit
UARTx_RXD	Input	UARTx receive
UARTx_nCTS	Input	UARTx modem clear to send
UARTx_nRTS	Output	UARTx modem request to send

Table 6.14-3 UART Interface Controller Pin

6.14.5 Functional Description

The UART controller supports four function modes including UART, IrDA, LIN and RS-485 mode. User can select a function by setting the UART_FUNCSEL register. The four function modes will be described in following section.

6.14.5.1 UART Controller Baud Rate Generator

The UART controller includes a programmable baud rate generator capable of dividing clock input by divisors to produce the serial clock that transmitter and receiver need. Table 6.14-4 list the UART baud rate equations in the various conditions. Table 6.14-5 and Table 6.14-6 list the UART baud rate parameter and register setting example. In IrDA function mode, the baud rate generator must be set in mode 0. More detail register description is shown in UART_BAUD register. There are three setting mode. Mode 0 is set by UART_BAUD[29:28] with 00. Mode 1 is set by UART_BAUD[29:28] with 10. Mode 2 is set by UART_BAUD[29:28] with 11.

Mode	**BAUDM1**	**BAUDM0**	**Baud Rate Equation**
Mode 0	0	0	UART_CLK / [16 * (BRD+2)].
Mode 1	1	0	UART_CLK / [(EDIVM1+1) * (BRD+2)], EDIVM1 must >= 8.
Mode 2	1	1	UART_CLK / (BRD+2) If UART_CLK <= 3*HCLK, BRD must >= 9. If UART_CLK > 3*HCLK, BRD must >= 3*N – 1. N is the smallest integer larger than or equal to the ratio of UART_CLK /HCLK. For example, if 3*HCLK < UART_CLK =< 4*HCLK, BRD must >=11. if 4*HCLK < UART_CLK =< 5*HCLK, BRD must >=14. (If the UART_CLK is selected from LXT, BRD can be greater than or equal to 1)

Table 6.14-4 UART Controller Baud Rate Equation Table

UART Peripheral Clock = 12 MHz			
Baud Rate	Mode 0	Mode 1	Mode 2
921600	Not support	Not recommended	BRD=11
460800	Not recommended	BRD=0, EDIVM1 =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 6.14-5 UART Controller Baud Rate Parameter Setting Example Table

UART Peripheral Clock = 12 MHz			
Baud Rate	UART_BAUD Value		
	Mode 0	Mode 1	Mode 2
921600	Not support	Not recommended	0x3000_000B
460800	Not recommended	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 6.14-6 UART controller Baud Rate Register Setting Example Table

6.14.5.2 UART Controller Baud Rate Compensation

The UART controller supports baud rate compensation function. It is used to optimize the precision in each bit. The precision of the compensation is half of UART module clock because there is 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 6.14-7 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 6.14-8 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 1 st rising edge is detected. Then, auto-baud rate counter value divided by ABRDBITS (UART_ALTCTL[20:19]) is loaded to BRD (UART_BAUD[15:0]) automatically. ABRDEN (UART_ALTCTL[18]) is cleared. The Auto-Baud is shown in Figure 6.14-4. 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.

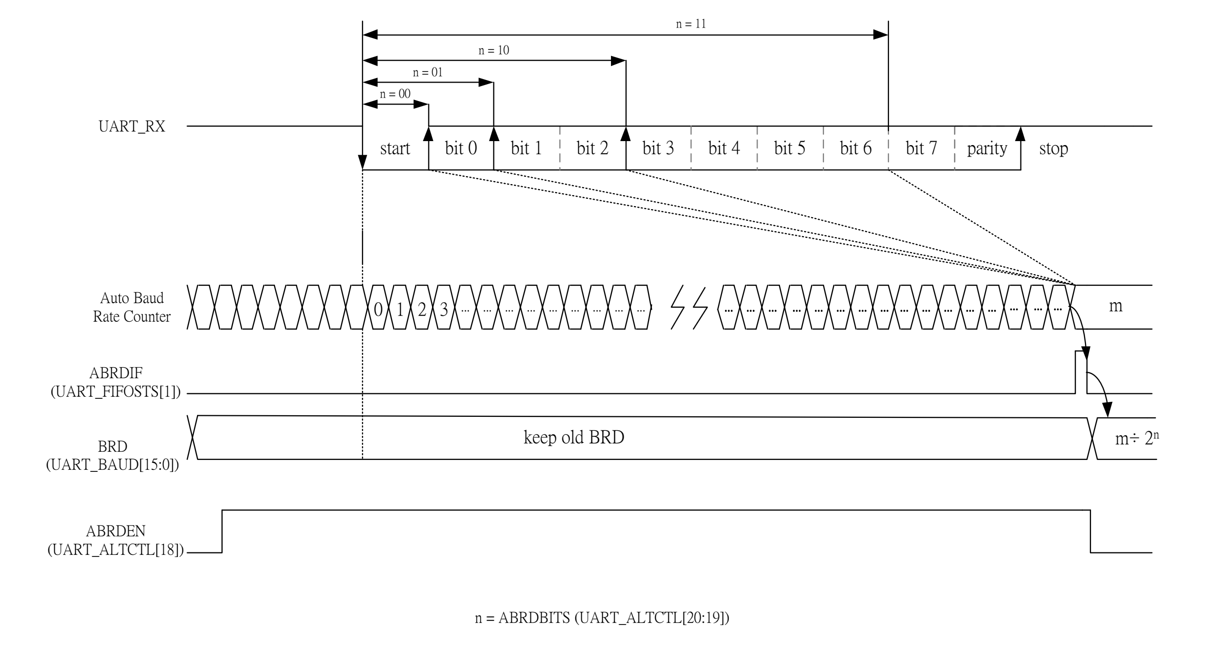


Figure 6.14-4 Auto-Baud Rate Measurement

6.14.5.3 Programming Sequence Example:

1. Program ABRDBITS (UART_ALTCTL[20:19]) to determines UART RX data 1st rising edge time from Start by 2 ABRDBITS bit time.
2. Set ABRIEN (UART_INTEN[18]) to enable auto-baud rate function interrupt.
3. Set ABRDEN (UART_ALTCTL[18]) to enable auto-baud rate function.
4. ABRDIF (UART_FIFOSTS[1]) is set, the auto-baud rate measurement is finished.
5. Operate UART transmit and receive action.
6. ABRDTOIF (UART_FIFOSTS[2]) is set, if auto-baud rate counter is overflow.
7. Go to Step 3.

6.14.5.4 UART Controller Transmit Delay Time Value

The UART controller programs DLY (UART_TOUT [15:8]) to control the transfer delay time between the last stop bit and next start bit in transmission. The unit is baud. The operation is shown in Figure 6.14-5.

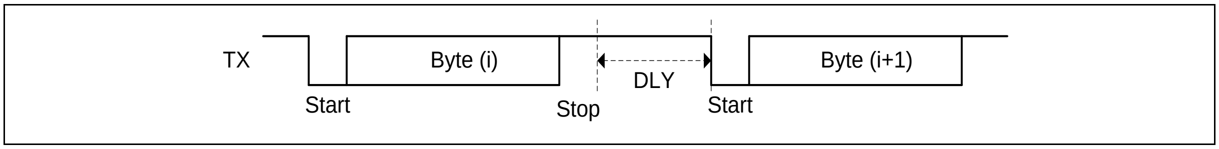


Figure 6.14-5 Transmit Delay Time Operation

6.14.5.5 UART Controller FIFO Control and Status

The UART controller is built-in with a 16 bytes transmitter FIFO (TX_FIFO) and a 16 bytes receiver FIFO (RX_FIFO) that reduces the number of interrupts presented to the CPU. The CPU can read the status of the UART at any time during operation. The reported status information includes condition of the transfer operations being performed by the UART, as well as 3 error conditions (parity error, framing error, break interrupt) occur if receiving data has parity, frame or break error. UART, IrDA, LIN and RS-485 mode support FIFO control and status function.

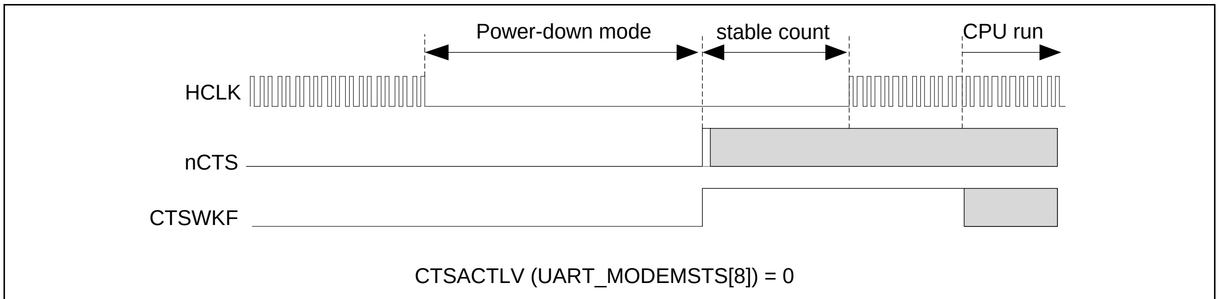
6.14.5.6 UART Controller Wake-up Function

The UART controller supports wake-up system function. The wake-up function includes nCTS pin, incoming data wake-up, Received Data FIFO reached threshold wake-up, RS-485 Address Match (AAD mode) wake-up and Received Data FIFO threshold time-out wake-up function. CTSWKF (UART_WKSTS[0]), DATWKF (UART_WKSTS[1]), RFRTWKF (UART_WKSTS[2]), RS485WKF (UART_WKSTS[3]) or TOUTWKF (UART_WKSTS[4]) cause the wake-up interrupt flag WKIF(UART_INTSTS[6]) 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 6.14-6 and Figure 6.14-7.

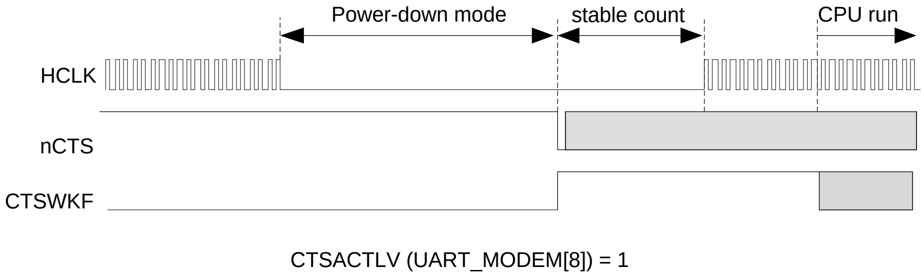
<u>nCTS Wake-up Case 1 (nCTS transition from low to high)</u>



Note: Stable count means HCLK source recovery stable count.

Figure 6.14-6 UART nCTS Wake-up Case1

<u>nCTS Wake-up Case 2 (nCTS transition from high to low)</u>



Note: Stable count means HCLK source recovery stable count.

Figure 6.14-7 UART nCTS Wake-up Case2

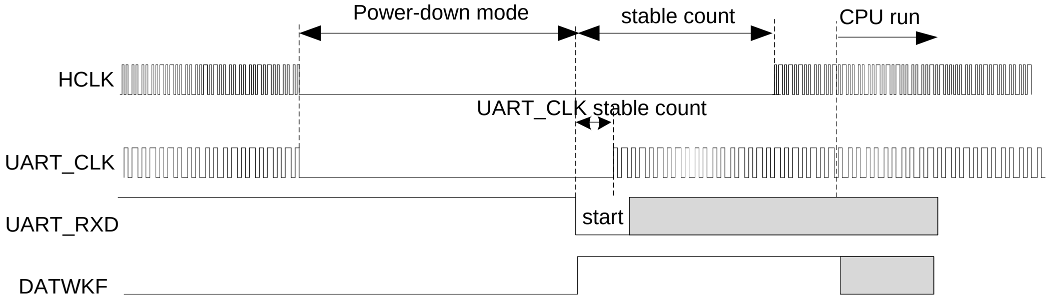
Incoming Data Wake-up

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 imcoing data wake-up is shown in Figure 6.14-8.

Note1: The UART controller clock source should be selected as HIRC and the compensation time for start bit is about 10.865us. It means that the value of STCOMP (UART_DWKCOMP[15:0]) can be set as 0x207.

Note2: The value of BRD(UART_BAUD[15:0]) should be greater than STCOMP (UART_DWKCOMP[15:0]).



Note1: Stable count means HCLK source recovery stable count.

Note2: UART_CLK stable count means UART clock source recovery stable count.

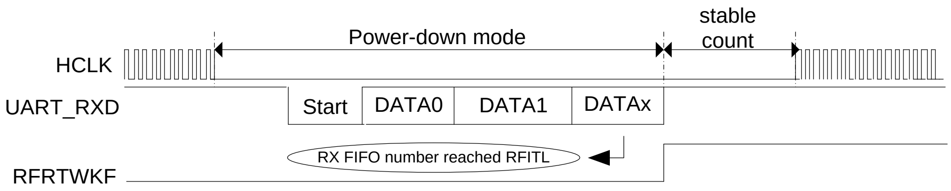
Figure 6.14-8 UART Data Wake-up

Received Data FIFO Reached Threshold Wake-up

The received data FIFO threshold reached wake-up function is enabled by setting WKRFRTEN (UART_WKCTL[2]). In Power-down mode, when the number of received data in RX FIFO reaches the threshold value RFITL (UART_FIFO[7:4]), it can wake-up the system. If the WKRFRTEN (UART_WKCTL[2]) is enabled, the number of received data in RX FIFO reaches the threshold value RFITL (UART_FIFO[7:4]) cause the received data FIFO reached threshold wake-up flag RFRTWKF (UART_WKSTS[2]) is generated. The Received Data FIFO reached threshold wake-up is shown in

Figure 6.14-9.

Note: The UART controller clock source should be selected as LXT in Power-down mode to receive data.



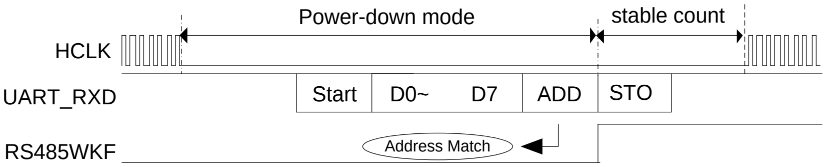
Note: Stable count means HCLK source recovery stable count.

Figure 6.14-9 UART 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 6.14-10.

Note: The UART controller clock source should be selected as LXT in Power-down mode to receive data.



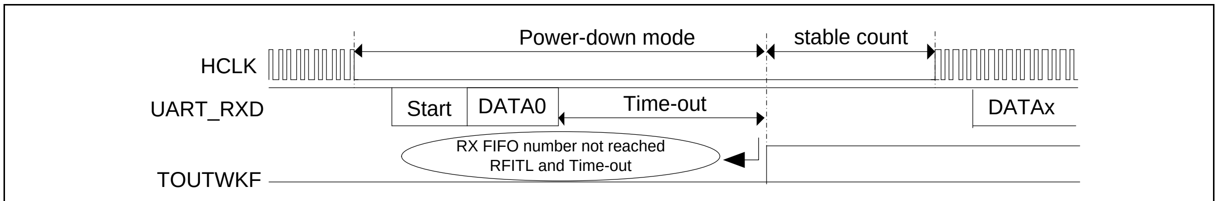
Note: Stable count means HCLK source recovery stable count.

Figure 6.14-10 UART 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 6.14-11.

Note: The UART controller clock source should be selected as LXT in Power-down mode to receive data.



Note: Stable count means HCLK source recovery stable count.

Figure 6.14-11 UART Received Data FIFO threshold time-out wake-up

6.14.5.7 UART Controller Interrupt and Status

Each UART controller supports ten types of interrupts including:

- Receive Data Available Interrupt (RDAINT)
- Transmit Holding Register Empty Interrupt (THERINT)
- Transmitter Empty Interrupt (TXENDIF)
- Receive Line Status Interrupt (RLSINT)
  - Break Interrupt Flag (BIF)
  - Framing Error Flag (FEF)
  - Parity Error Flag (PEF)
  - RS-485 Address Byte Detect Flag (ADDRDETF)
- MODEM Status Interrupt (MODEMINT)
  - Detect nCTS State Change Flag (CTSDETF)
- Receiver Buffer Time-out Interrupt (RXTOINT)
- Buffer Error Interrupt (BUFERRINT)
  - TX Overflow Error Interrupt Flag (TXOVIF)
  - RX Overflow Error Interrupt Flag (RXOVIF)
- LIN Bus Interrupt (LININT)
  - LIN Break Detection Flag (BRKDETF)
  - Bit Error Detect Status Flag (BITEF)
  - LIN Slave ID Parity Error Flag (SLVIDPEF)
  - LIN Slave Header Error Flag (SLVHEF)
  - LIN Slave Header Detection Flag (SLVHDETF)
- Wake-up Interrupt (WKINT)
  - nCTS Wake-up Flag (CTSWKF)
  - Incoming Data Wake-up Flag (DATWKF)
  - Received Data FIFO Reached Threshold Wake-up Flag (RFRTWKF)

- RS-485 Address Match (AAD mode) Wake-up Flag (RS485WKF)
- Received Data FIFO Threshold Time-out Wake-up Flag (TOUTWKF)

- Auto-Baud Rate Interrupt (ABRINT)
  - Auto-baud Rate Detect Interrupt Flag (ABRDIF)
  - Auto-baud Rate Detect Time-out Interrupt Flag (ABRDTOIF)

Table 6.14-9 describes the interrupt sources and flags. The interrupt is generated when the interrupt flag is generated and the interrupt enable bit is set. User must clear the interrupt flag after the interrupt is generated.

Interrupt Source	Interrupt Indicator	Interrupt Enable Bit	Interrupt Flag	Flag Caused By	Flag Cleared By
Receive Data Available Interrupt	RDAINT	RDAIEN	RDAIF	N/A	Read UART_DAT
Transmit Holding Register Empty Interrupt	THERINT	THREIEN	THREIF	N/A	Write UART_DAT
Transmitter Empty Interrupt	TXENDINT	TXENDIEN	TXENDIF	N/A	Write UART_DAT
Receive Line Status Interrupt	RLSINT	RLSIEN	RLSIF	RLSIF = BIF	Write ‘1’ to BIF
				RLSIF = FEF	Write ‘1’ to FEF
				RLSIF = PEF	Write ‘1’ to PEF
				RLSIF = ADDRDETF	Write ‘1’ to ADDRDETF
Modem Status Interrupt	MODEMINT	MODEMIEN	MODEMIF	MODEMIF = CTSDETF	Write ‘1’ to CTSDETF
Receiver Buffer Time-out Interrupt	RXTOINT	RXTOIEN	RXTOIF	N/A	Read UART_DAT
Buffer Error Interrupt	BUFERRINT	BUFERRIEN	BUFERRIF	BUFERRIF = TXOVIF	Write ‘1’ to TXOVIF
				BUFERRIF = RXOVIF	Write ‘1’ to RXOVIF
LIN Bus Interrupt	LININT	LINIEN	LINIF	LINIF = BRKDETF	Write ‘1’ to LINIF and Write ‘1’ to BRKDETF
				LINIF = BITEF	Write ‘1’ to LINIF and Write ‘1’ to BITEF
				LINIF = SLVIDPEF	Write ‘1’ to LINIF and Write ‘1’ to SLVIDPEF

				LINIF = SLVHEF	Write ‘1’ to LINIF and Write ‘1’ to SLVHEF
				LINIF = SLVHDETF	Write ‘1’ to LINIF and Write ‘1’ to SLVHDETF
Wake-up Interrupt	WKINT	WKIEN	WKIF	WKIF = CTSWKF	Write ‘1’ to CTSWKF
				WKIF = DATWKF	Write ‘1’ to DATWKF
				WKIF = RFRTWKF	Write ‘1’ to RFRTWKF
				WKIF = RS485WKF	Write ‘1’ to RS485WKF
				WKIF = TOUTWKF	Write ‘1’ to TOUTWKF
Auto-Baud Rate Interrupt	ABRINT	ABRIEN	ABRIF	ABRIF = ABRDIF	Write ‘1’ to ABRDIF
				ABRIF = ABRDTOIF	Write ‘1’ to ABRDTOIF

Table 6.14-9 UART controller Interrupt Source and Flag List

6.14.5.8 UART Function Mode

The UART controller provides UART function (Setting FUNCSEL (UART_FUNCSEL [1:0]) to ’00’ to enable UART function mode). The UART baud rate is up to 1 Mbps.

The UART provides full-duplex and asynchronous communications. The transmitter and receiver contain 16 bytes FIFO for payloads. User can program receiver buffer trigger level and receiver buffer time-out detection for receiver. The transmitting data delay time between the last stop and the next start bit can be programed by setting DLY (UART_TOUT [15:8]) register. The UART supports hardware auto-flow control that provides programmable nRTS flow control trigger level. The number of data bytes in RX FIFO is equal to or greater than RTSTRGLV (UART_FIFO[19:16]), and the nRTS is de-asserted.

UART Line Control Function

The UART controller supports fully programmable serial-interface characteristics by setting the UART_LINE register. User can program UART_LINE register for the word length, stop bit and parity bit setting. Table 6.14-10 and Table 6.14-11 list the UART word, stop bit length and the parity bit settings.

NSB (UART_LINE[2])	**WLS (UART_LINE[1:0])**	**Word Length (Bit)**	**Stop Length (Bit)**
0	00	5	1
0	01	6	1
0	10	7	1
0	11	8	1
1	00	5	1.5
1	01	6	2
1	10	7	2

1	11	8	2

Table 6.14-10 UART Line Control of Word and Stop Length Setting

Parity Type	SPE (UART_LINE[5])	EPE (UART_LINE[4])	PSS (UART_LINE[7])	PBE (UART_LINE[3])	Description
No Parity	x	x	x	0	No parity bit output.
Parity source from UART_DAT	x	x	1	1	Parity bit is generated and checked by software.
Odd Parity	0	0	0	1	Odd Parity is calculated by adding all the “1’s” in a data stream and adding a parity bit to the total bits, to make the total count an odd number.
Even Parity	0	1	0	1	Even Parity is calculated by adding all the “1’s” in a data stream and adding a parity bit to the total bits, to make the count an even number.
Forced Mask Parity	1	0	0	1	Parity bit always logic 1. Parity bit on the serial byte is set to “1” regardless of total number of “1’s” (even or odd counts).
Forced Space Parity	1	1	0	1	Parity bit always logic 0. Parity bit on the serial byte is set to “0” regardless of total number of “1’s” (even or odd counts).

Table 6.14-11 UART Line Control of Parity Bit Setting

UART Auto-Flow Control Function

The UART supports auto-flow control function that uses two signals, nCTS (clear-to-send) and nRTS (request-to-send), to control the flow of data transfer between the UART and external devices (e.g. Modem). When auto-flow is enabled, the UART is not allowed to receive data until the UART asserts nRTS to external device. When the number of bytes stored in the RX FIFO equals the value of RTSTRGLV (UART_FIFO [19:16]), the nRTS is de-asserted. The UART sends data out when UART detects nCTS is asserted from external device. If the valid asserted nCTS is not detected, the UART will not send data out. The auto flow control block diagram is shown in Figure 6.14-12.

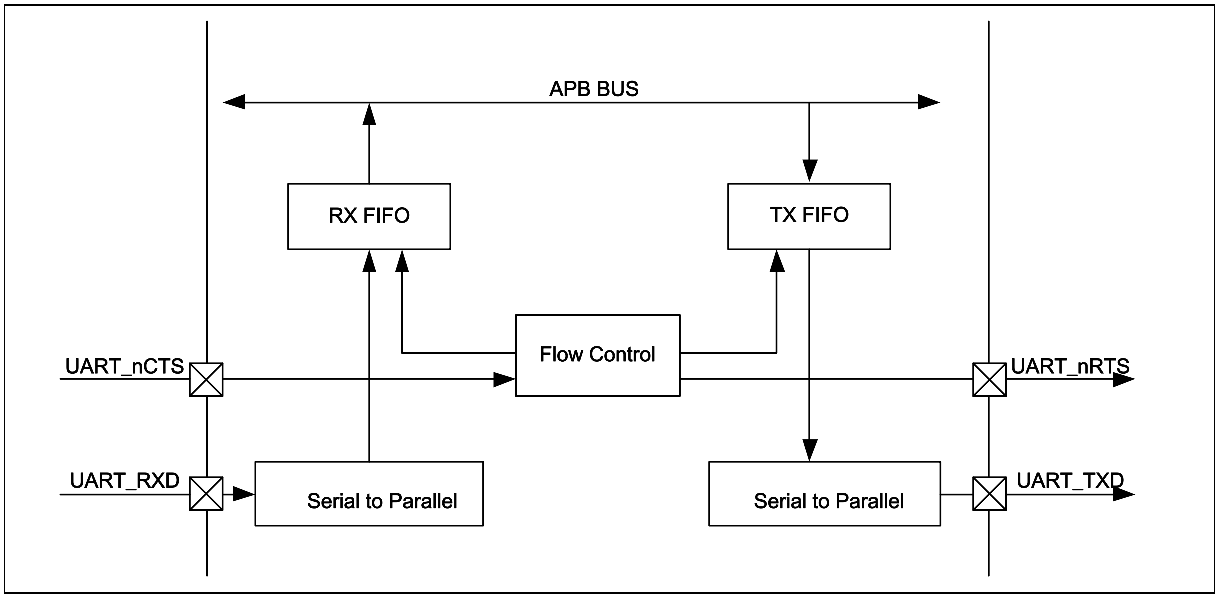


Figure 6.14-12 Auto-Flow Control Block Diagram

Figure 6.14-13 demonstrates the nCTS auto-flow control of UART function mode. User must set ATOCTSEN (UART_INTEN [13]) to enable nCTS auto-flow control function. The CTSACTLV (UART_MODEMSTS [8]) can set nCTS pin input active state. The CTSDETF (UART_MODEMSTS[0]) is set when any state change of nCTS pin input has occurred, and then TX data will be automatically transmitted from TX FIFO.

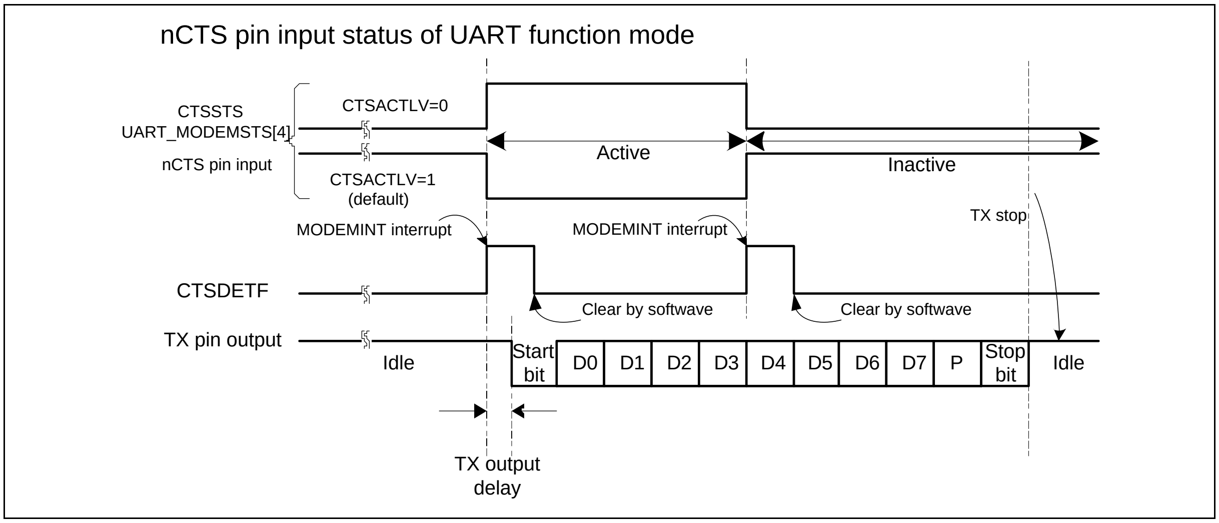


Figure 6.14-13 UART nCTS Auto-Flow Control Enabled

As shown in Figure 6.14-14, in UART nRTS auto-flow control mode (ATORTSEN(UART_INTEN[12])=1), the nRTS internal signal is controlled by UART FIFO controller with RTSTRGLV(UART_FIFO[19:16]) trigger level.

Setting RTSACTLV(UART_MODEM[9]) can control the nRTS pin output is inverse or non-inverse from nRTS signal. User can read the RTSSTS (UART_MODEM[13]) bit to get real nRTS pin output voltage logic status.

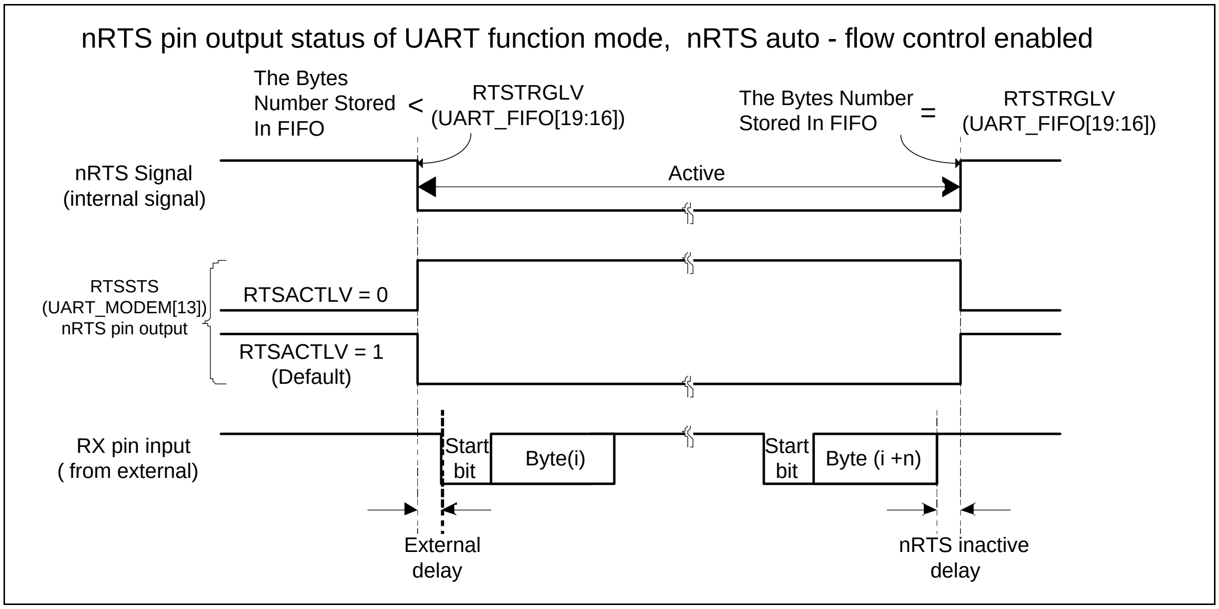

nRTS pin output status of UART function mode, nRTS auto - flow control enabled
The Bytes Number Stored In FIFO < RTSTRGLV (UART_FIFO[19:16])
The Bytes Number Stored In FIFO = RTSTRGLV (UART_FIFO[19:16])
nRTS Signal (internal signal)
Active
RTSSTS (UART_MODEM[13]) nRTS pin output
RTSACTLV = 0
RTSACTLV = 1 (Default)
RX pin input (from external)
Start bit
Byte(i)
Start bit
Byte (i +n)
External delay
nRTS inactive delay

Figure 6.14-14 UART nRTS Auto-Flow Control Enabled

As shown in Figure 6.14-15, 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.

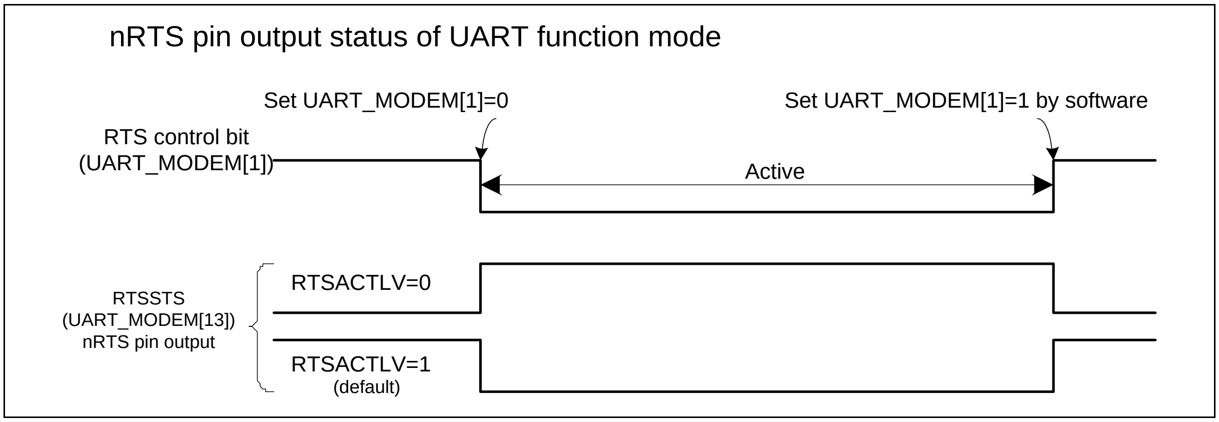

nRTS pin output status of UART function mode
Set UART_MODEM[1]=0
Set UART_MODEM[1]=1 by software
RTS control bit (UART_MODEM[1])
Active
RTSSTS (UART_MODEM[13]) nRTS pin output
RTSACTLV=0
RTSACTLV=1 (default)

Figure 6.14-15 UART nRTS Auto-Flow with Software Control

6.14.5.9 IrDA Function Mode

The UART controller also provides Serial IrDA (SIR, Serial Infrared) function (Setting UART_FUNCSEL [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 6.14-16.

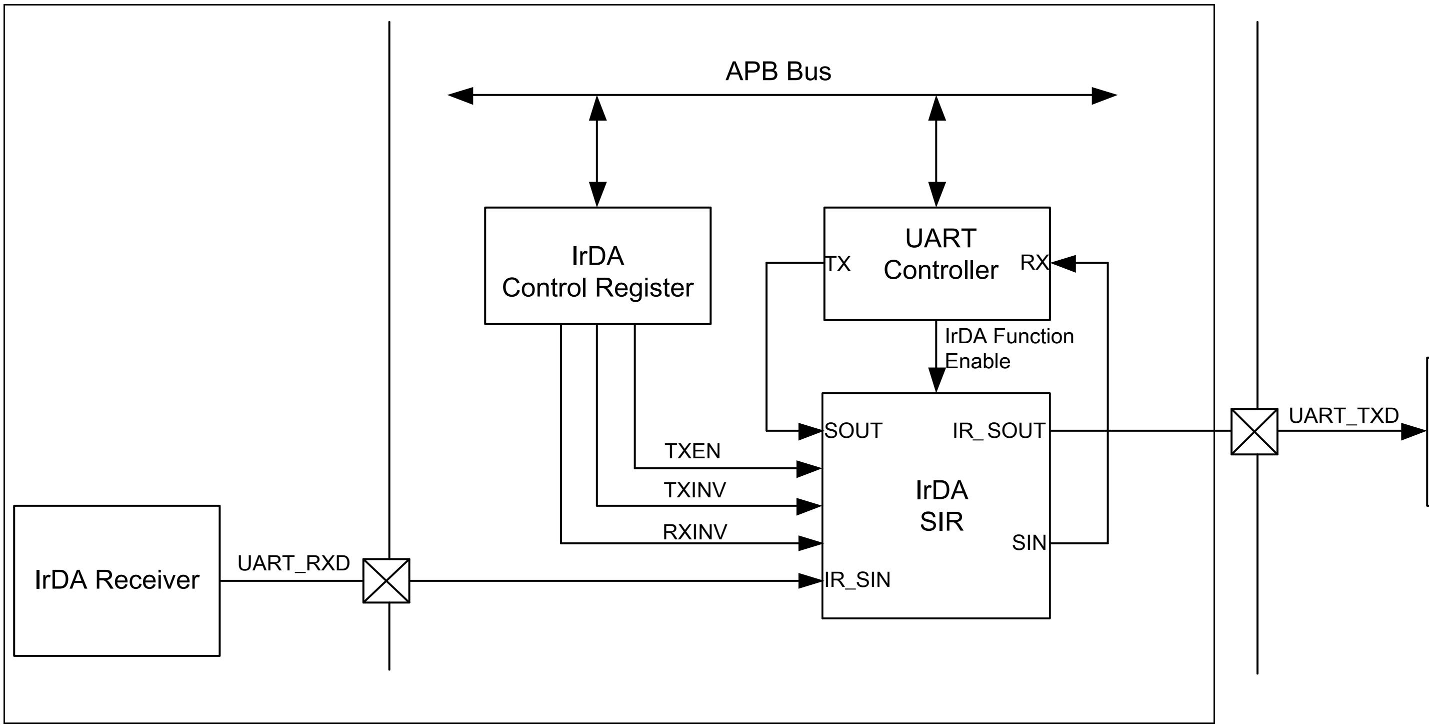


Figure 6.14-16 IrDA Control Block Diagram

IrDA SIR Transmit Encoder

The IrDA SIR Transmit Encoder modulates Non-Return-to-Zero (NRZ) transmit bit stream output from UART. The IrDA SIR physical layer specifies the use of Return-to-Zero, Inverted (RZI) modulation scheme which represents logic 0 as an infra light pulse. The modulated output pulse stream is transmitted to an external output driver and infrared light emitting diode.

The transmitted pulse width is specified as 3/16 period of baud rate.

IrDA SIR Receive Decoder

The IrDA SIR Receive Decoder demodulates the Return-to-Zero bit stream from the input detector and outputs the NRZ serial bits stream to the UART received data input.

In idle state, the decoder input is high. A start bit is detected when the decoder input is LOW. In normal operation, the RXINV (UART_IRDA[6]) is set to ‘1’ and TXINV (UART_IRDA[5]) is set to ‘0’.

IrDA SIR Operation

The IrDA SIR encoder/decoder provides functionality which converts between UART data stream and half-duplex serial SIR interface. Figure 6.14-17 is IrDA encoder/decoder waveform.

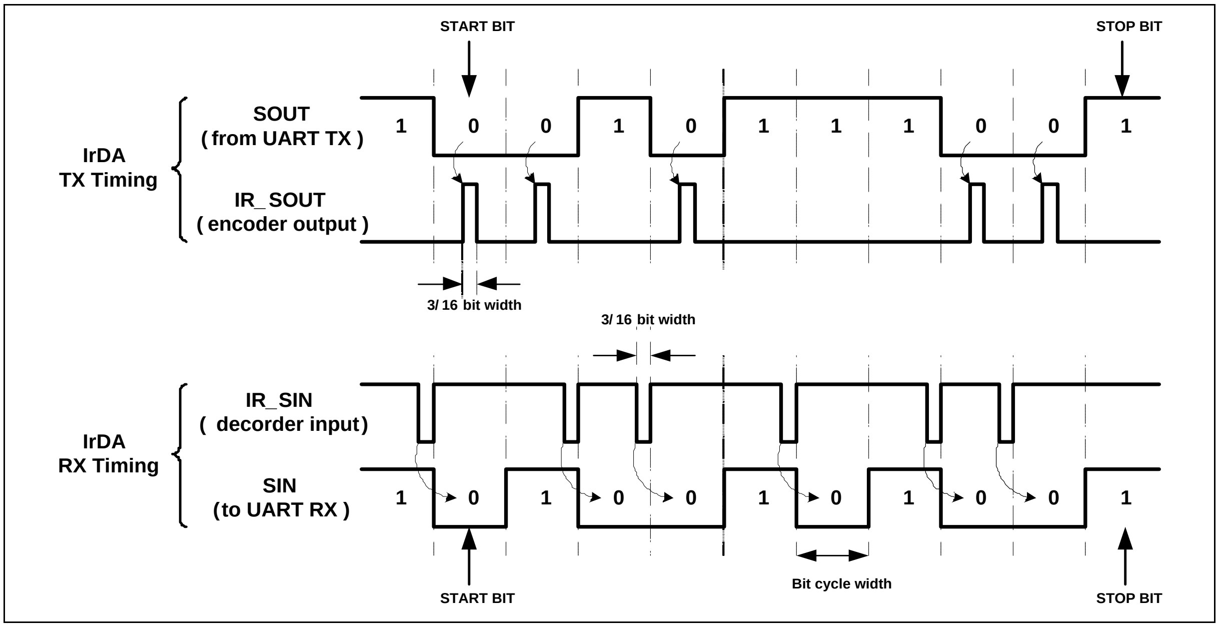


Figure 6.14-17 IrDA TX/RX Timing Diagram

6.14.5.10 LIN Function Mode (Local Interconnection Network)

The UART Controller supports LIN function. Setting FUNCSEL (UART_FUNCSEL[1:0]) to '01' to select LIN mode operation. The UART Controller supports LIN break/delimiter generation and break/delimiter detection in LIN master mode, and supports header detection and automatic resynchronization in LIN Slave mode.

Structure of LIN Frame

According to the LIN protocol, all information transmitted is packed as frames; a frame consists of a header (provided by the master task) and a response (provided by a slave task). The header (provided by the master task) consists of a break field and a sync field followed by a frame identifier (frame ID). The frame identifier uniquely defines the purpose of the frame. The slave task is appointed for providing the response associated with the frame ID. The response consists of a data field and a checksum field. Figure 6.14-18 is the structure of LIN Frame.

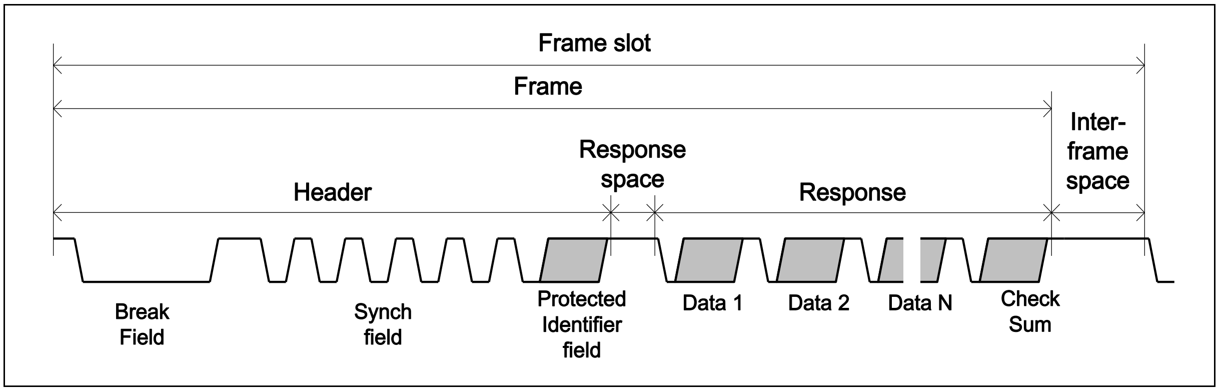


Figure 6.14-18 Structure of LIN Frame

Structure of LIN Byte

In LIN mode, each byte field is initiated by a START bit with value 0 (dominant), followed by 8 data bits and no parity bit, LSB is first and ended by 1 stop bit with value 1 (recessive) in accordance with the LIN standard. The structure of Byte is shown in Figure 6.14-19.

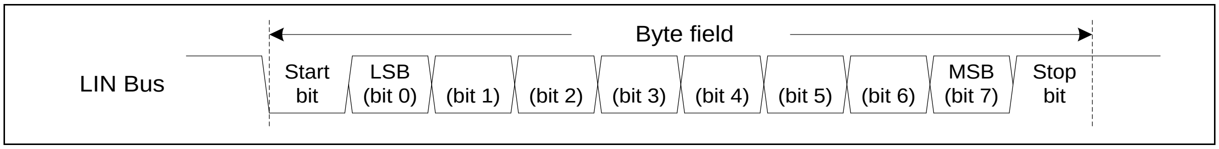


Figure 6.14-19 Structure of LIN Byte

LIN Master Mode

The UART Controller supports LIN Master mode. To enable and initialize the LIN Master mode, the following steps are necessary:

1. Set the UART_BAUD register to select the desired baud rate.
2. Set WLS (UART_LINE[1:0]) to ‘11’ to configure the word length with 8 bits, clearing PBE (UART_LINE[3]) bit to disable parity check and clearing NSB (UART_LINE[2]) bit to configure with one stop bit.
3. Set FUNCSEL (UART_FUNCSEL[1:0]) to ‘01’ to select LIN function mode operation.

A complete header consists of a break field and sync field followed by a frame identifier (frame ID). The UART controller can be selected header sending by three header selected modes. The header selected mode can be “break field” or “break field and sync field” or “break field, sync field and frame ID field” by setting HSEL (UART_LINCTL[23:22]). If the selected header is “break field”, software must handle the following sequence to send a complete header to bus by filling sync data (0x55) and frame ID data to the UART_DAT register. If the selected header is “break field and sync field”, software must handle the sequence to send a complete header to bus by filling the frame ID data to UART_DAT register, and if the selected header is “break field, sync field and frame ID field”, hardware will control the header sending sequence automatically but software must filled frame ID data to PID (UART_LINCTL [31:24]). When operating in header selected mode in which the selected header is “break field, sync field and frame ID field”, the frame ID parity bit can be calculated by software or hardware depending whether the IDPEN (UART_LINCTL[9]) bit is set or not.

HSEL	Break Field	Sync Field	ID Field
0	Generated by Hardware	Handled by Software	Handled by Software
1	Generated by Hardware	Generated by Hardware	Handled by Software
2	Generated by Hardware	Generated by Hardware	Generated by Hardware (But Software needs to fill ID to PID (UART_LINCTL[31:24]) first)

Table 6.14-12 LIN Header Selection in Master Mode

When UART is operated in LIN data transmission, LIN bus transfer state can be monitored by hardware or software. User can enable hardware monitoring by setting BITERREN (UART_LINCTL [12]) to “1”, if the input pin (UART_RX) state is not equal to the output pin (UART_TX) state in LIN transmitter state that hardware will generate an interrupt to CPU. Software can also monitor the LIN bus transfer state by checking the read back data in UART_DAT register. The following sequence is a program sequence example.

The procedure without software error monitoring in Master mode:

1. Fill Protected Identifier to PID (UART_LINCTL[31:24]).
2. Select the hardware transmission header field including “break field + sync field + protected identifier field” by setting HSEL (UART_LINCTL [23:22]) to “10”.
3. Set SENDH (UART_LINCTL[8]) bit to 1 for requesting header transmission.

4. Wait until SENDH (UART_LINCTL[8]) bit cleared by hardware.
5. Wait until TXEMPTYF (UART_FIFOSTS[28]) set to 1 by hardware.

Note1: The default setting of break field is 12 dominant bits (break field) and 1 recessive bit break/sync delimiter. Setting BRKFL (UART_LINCTL [19:16]) and BSL (UART_LINCTL[21:20]) to change the LIN break field length and break/sync delimiter length.

Note2: The default setting of break/sync delimiter length is 1-bit time and the inter-byte spaces default setting is also 1-bit time. Setting BSL (UART_LINCTL[21:20]) and DLY(UART_TOUT[15:8]) can change break/sync delimiter length and inter-byte spaces.

Note3: If the header includes the “break field, sync field and frame ID field”, software must fill frame ID to PID (UART_LINCTL[31:24]) before trigger header transmission (setting the SENDH (UART_LINCTL[8])). The frame ID parity can be generated by software or hardware depending on IDPEN (UART_LINCTL[9]) setting. If the parity generated by software with IDPEN (UART_LINCTL[9]) is set to ‘0’, software must fill 8 bit data (include 2 bit parity) in this field. If the parity generated by hardware with IDPEN (UART_LINCTL[9]) is set to ‘1’, software fills ID0~ID5 and hardware calculates P0 and P1.

Procedure with software error monitoring in Master mode:

1. Choose the hardware transmission header field to only include “break field” by setting HSEL (UART_LINCTL [23:22])] to ‘00’.
2. Enable break detection function by setting BRKDETEN (UART_LINCTL[10]).
3. Request break + break/sync delimiter transmission by setting the SENDH (UART_LINCTL[8]).
4. Wait until the BRKDETF (UART_LINSTS[8]) flag is set to “1” by hardware.
5. Request sync field transmission by writing 0x55 into UART_DAT register.
6. Wait until the RDAIF (UART_INTSTS[0]) is set to “1” by hardware and then read back the UART_DAT register.
7. Request header frame ID transmission by writing the protected identifier value to UART_DAT register.
8. Wait until the RDAIF (UART_INTSTS[0]) is set to “1” by hardware and then read back the UART_DAT register.

LIN Break and Delimiter Detection

When software enables the break detection function by setting BRKDETEN (UART_LINCTL[10]), the break detection circuit is activated. The break detection circuit is totally independent from the UART receiver.

When the break detection function is enabled, the circuit looks at the input UART_RX pin for a start signal. If UART LIN controller detects consecutive dominant is greater than 11 bits dominant followed by a recessive bit (delimiter), the BRKDETF (UART_LINSTS[8]) flag is set at the end of break field. If the LINIEN (UART_INTEN[8]) bit is set to 1, an interrupt LININT (UART_INTSTS[15]) will be generated. The behavior of the break detection and break flag are shown in Figure 6.14-20.

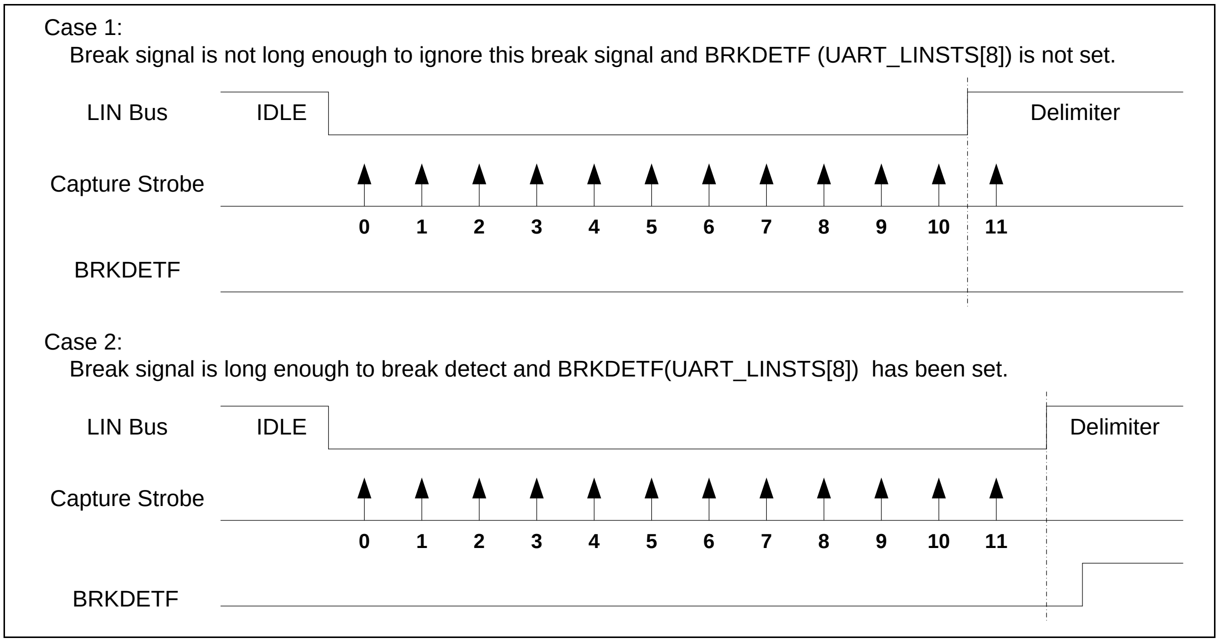

Case 1:
Break signal is not long enough to ignore this break signal and BRKDETF (UART_LINSTS[8]) is not set.
LIN Bus
IDLE
Delimiter
Capture Strobe
0 1 2 3 4 5 6 7 8 9 10 11
BRKDETF
Case 2:
Break signal is long enough to break detect and BRKDETF(UART_LINSTS[8]) has been set.
LIN Bus
IDLE
Delimiter
Capture Strobe
0 1 2 3 4 5 6 7 8 9 10 11
BRKDETF

Figure 6.14-20 Break Detection in LIN Mode

LIN Frame ID and Parity Format

The LIN frame ID value in LIN function mode is shown, the frame ID parity can be generated by software or hardware depends on IDPEN (UART_LINCTL[9]).

If the parity generated by hardware (IDPEN (UART_LINCTL[9])=1), user fill ID0~ID5 (UART_LINCTL [29:24]) hardware will calculate P0 (UART_LINCTL[30]) and P1 (UART_LINCTL[31]) otherwise user must filled frame ID and parity in this field.

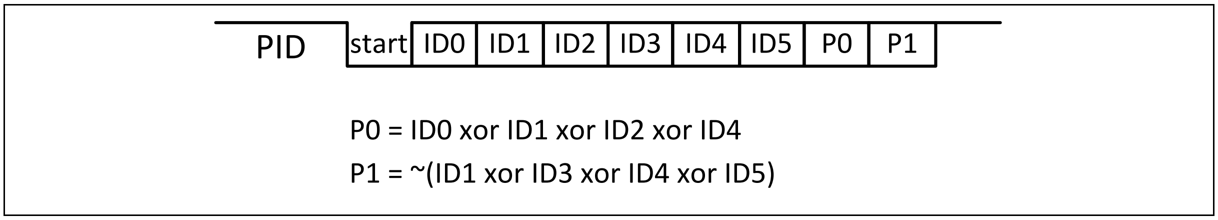

PID
start
ID0
ID1
ID2
ID3
ID4
ID5
P0
P1
P0 = ID0 xor ID1 xor ID2 xor ID4
P1 = ~(ID1 xor ID3 xor ID4 xor ID5)

Figure 6.14-21 LIN Frame ID and Parity Format

LIN Slave Mode

The UART Controller supports LIN Slave mode. To enable and initialize the LIN Slave mode, the following steps are necessary:

1. Set the UART_BAUD register to select the desired baud rate.
2. Configure the data length to 8 bits by setting WLS (UART_LINE[1:0]) to ‘11’ and disable parity check by clearing PBE (UART_LINE[3]) bit and configure with one stop bit by clearing NSB (UART_LINE[2]) bit.
3. Select LIN function mode by setting FUNCSEL (UART_FUNCSEL[1:0]) to ‘01’.
4. Enable LIN slave mode by setting the SLVEN (UART_LINCTL[0]) to 1.

LIN Header Reception

According to the LIN protocol, a slave node must wait for a valid header which comes from the master node. Next the slave task will take one of following actions (depend on the master header frame ID value).

- Receive the response.
- Transmit the response.
- Ignore the response and wait for next header.

In LIN Slave mode, user can enable the slave header detection function by setting the SLVHDEN (UART_LINCTL[1]) to detect complete frame header (receive “break field”, “sync field” and “frame ID field”). When a LIN header is received, the SLVHDETF (UART_LINSTS[0]) flag will be set. If the LINIEN (UART_INTEN[8]) bit is set to 1, an interrupt will be generated. User can enable the frame ID parity check function by setting IDPEN (UART_LINCTL[9]). If only received frame ID parity is not correct (break and sync filed are correct), the SLVIDPEF (UART_LINSTS[2]) flag is set to ‘1’. If the LINIEN(UART_INTEN[8]) is set to 1, an interrupt will be generated and SLVHDETF (UART_LINSTS[0]) is set to ‘1’. User can also put LIN in mute mode by setting MUTE (UART_LINCTL[4]) to ‘1’. This mode allows detection of headers only (break + sync + frame ID) and prevents the reception of any other characters. In order to avoid bit rate tolerance, the controller supports automatic resynchronization function to avoid clock deviation error, user can enable this feature by setting SLVAREN (UART_LINCTL[2]).

LIN Response Transmission

The LIN slave node can transmit response and receive response. When slave node is the publisher of the response, the slave node sends response by filling data to the UART_DAT register. If the slave node is the subscriber of the response, the slave node receives data from LIN bus.

LIN Header Time-out Error

The LIN slave controller contains a header time-out counter. If the entire header is not received within the maximum time limit of 57 bit times, the header error flag SLVHEF (UART_LINSTS [1]) will be set. The time-out counter is enabled at each break detect edge and stopped in the following conditions.

- A LIN frame ID field has been received.
- The header error flag asserts.
- Writing 1 to the SLVSYNCF (UART_LINSTS[3]) to re-search a new frame header.

Mute Mode and LIN Exit from Mute Mode Condition

In Mute mode, a LIN slave node will not receive any data until specified condition occurred. It allows header detection only and prevents the reception of any other characters. User can enable Mute mode by setting the MUTE (UART_LINCTL[4]) and exiting from Mute mode condition can be selected by HSEL (UART_LINCTL[23:22]).

Note: It is recommended to set LIN slave node to Mute mode after checksum transmission.

The LIN slave controller exiting from Mute mode is described as follows: If HSEL (UART_LINCTL[23:22]) is set to “break field”, when LIN slave controller detects a valid LIN break + delimiter, the controller will enable the receiver (exit from Mute mode) and subsequent data (sync data, frame ID data, response data) are received in RX FIFO.

If HSEL (UART_LINCTL[23:22]) is set to “break field and sync field”, when the LIN slave controller detects a valid LIN break + delimiter followed by a valid sync field without frame error, the controller will enable the receiver (exit from mute mode) and subsequent data(ID data, response data) are received in RX FIFO. If HSEL (UART_LINCTL[23:22]) is set to “break field, sync field and ID field”, when the LIN slave controller detects a valid LIN break + delimiter and valid sync field without frame error followed by ID data without frame error and received ID data matched PID (UART_LINCTL[31:24]) value. The controller will enable the receiver (exit from mute mode) and subsequent data (response data) are received in RX FIFO.

Slave Mode Non-automatic Resynchronization (NAR)

User can disable the automatic resynchronization function to fix the communication baud rate. When operating in Non-Automatic Resynchronization mode, software needs some initial process, and the initialization process flow of Non-Automatic Resynchronization mode is shown as follows:

1. Select the desired baud rate by setting the UART_BAUD register.
2. Select LIN function mode by setting FUNCSEL (UART_FUNCSEL[1:0]) to ‘01’.
3. Disable automatic resynchronization function by setting SLVAREN (UART_LINCTL[2]) is set to 0.
4. Enable LIN slave mode by setting the SLVEN (UART_LINCTL[0]) is set to 1.

Slave Mode with Automatic Resynchronization (AR)

In Automatic Resynchronization (AR) mode, the controller will adjust the baud rate generator after each sync field reception. The initialization process flow of Automatic Resynchronization mode is shown as follows:

1. Select the desired baud rate by setting the UART_BAUD register.
2. Select LIN function mode by setting UART_FUNCSEL (UART_FUNCSEL[1:0]) to ‘01’.
3. Enable automatic resynchronization function by setting SLVAREN (UART_LINCTL[2]) to ‘1’.
4. Enable LIN slave mode by setting the SLVEN (UART_LINCTL[0]) is set to ‘1’.

When the automatic resynchronization function is enabled, after each LIN break field, the time duration between five falling edges is sampled on peripheral clock and the result of this measurement is stored in an internal 13-bit register and the UART_BAUD register value will be automatically updated at the end of the fifth falling edge. If the measure timer (13-bit) overflows before five falling edges, then the header error flag SLVHEF (UART_LINSTS [1]) will be set.

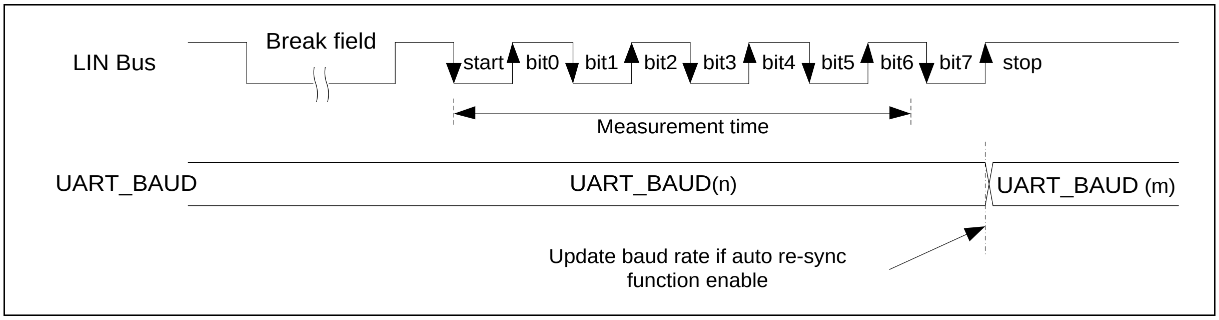


Figure 6.14-22 LIN Sync Field Measurement

When operating in Automatic Resynchronization (AR) mode, software must select the desired baud rate by setting the UART_BAUD register and hardware will store it at internal TEMP_REG register, after each LIN break field, the time duration between five falling edges is sampled on peripheral clock and the result of this measurement is stored in an internal 13-bit register BAUD_LIN and the result will be updated to UART_BAUD register automatically.

To guarantee the transmission baud rate, the baud rate generator must reload the initial value before each new break reception. The initial value is programmed by the application during initialization (TEMP_REG). User can set SLVDUEN (UART_LINCTL [3]) to enable auto reload initial baud rate value function. If the SLVDUEN (UART_LINCTL [3]) is set, when received the next character, hardware will auto reload the initial value to UART_BAUD, and when the UART_BAUD be updated, the SLVDUEN (UART_LINCTL [3]) will be cleared automatically. The behavior of LIN updated method as shown Figure 6.14-23.

Note1: It is recommended to set the SLVDUEN bit before every checksum reception.

Note2: When a header error is detected, user must write 1 to SLVSYNCF (UART_LINSTS[3]) to re-search new frame header. When writing 1 to it, hardware will reload the initial baud rate TEMP_REG and re-search new frame header.

Note3: When operating in Automatic Resynchronization mode, the baud rate setting must be operated at mode2 (BAUDM1 (UART_BAUD [29]) and BAUDM0 (UART_BAUD[28]) must be 1).

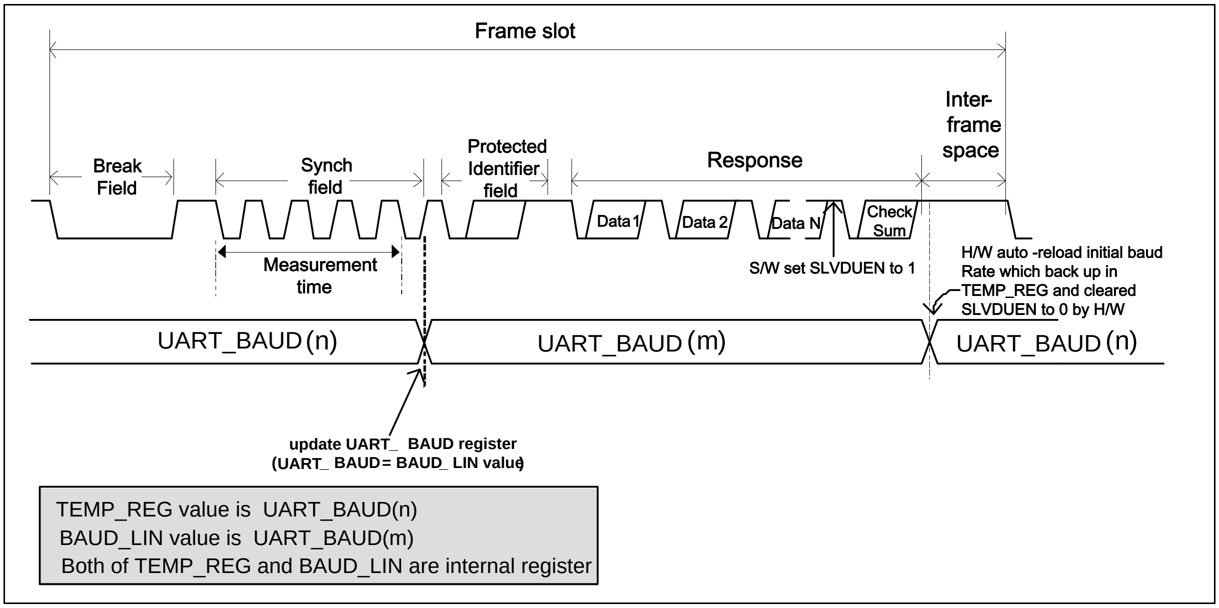


Figure 6.14-23 UART_BAUD Update Sequence in AR mode if SLVDUEN is 1

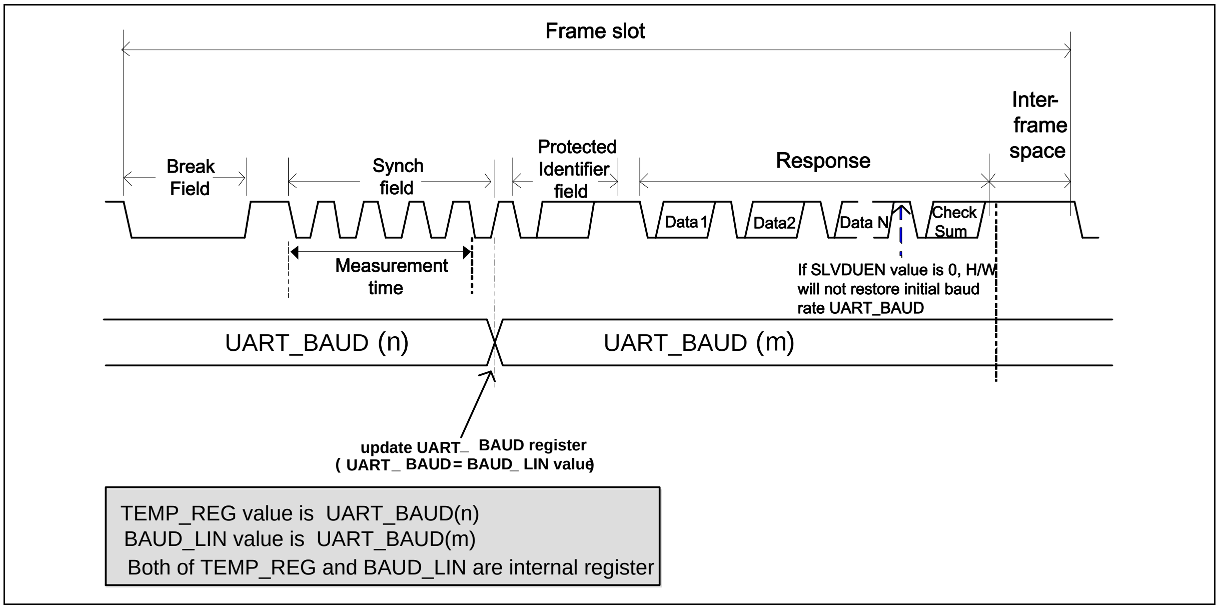

Frame slot
Inter-
frame
space
Break
Field
Synch
field
Protected
Identifier
field
Response
Data 1
Data2
Data N
Check
Sum
Measurement
time
If SLVDUEN value is 0, H/W
will not restore initial baud
rate UART_BAUD
UART_BAUD (n)
UART_BAUD (m)
update UART_ BAUD register
(UART_ BAUD= BAUD_ LIN value)
TEMP_REG value is UART_BAUD(n)
BAUD_LIN value is UART_BAUD(m)
Both of TEMP_REG and BAUD_LIN are internal register

Figure 6.14-24 UART_BAUD Update Sequence in AR mode if SLVDUEN is 0

Deviation Error on the Sync Field

When operating in Automatic Resynchronization mode, the controller will check the deviation error on the sync field. The deviation error is checked by comparing the current baud rate with the received sync field. Two checks are performed in parallel.

Check1: Based on measurement between the first falling edge and the last falling edge of the sync field.

- If the difference is more than 14.84%, the header error flag SLVHEF (UART_LINSTS[1]) will be set.
- If the difference is less than 14.06%, the header error flag SLVHEF (UART_LINSTS[1]) will not be set.
- If the difference is between 14.84% and 14.06%, the header error flag SLVHEF (UART_LINSTS[1]) may either set or not.

Check2: Based on measurement of time between each falling edge of the sync field.

- If the difference is more than 18.75%, the header error flag SLVHEF (UART_LINSTS[1]) will be set.
- If the difference is less than 15.62%, the header error flag SLVHEF (UART_LINSTS[1]) will not be set.
- If the difference is between 18.75% and 15.62%, the header error flag SLVHEF (UART_LINSTS[1]) may either set or not.

Note: The deviation check is based on the current baud rate clock. Therefore, in order to guarantee correct deviation checking, the baud rate must reload the nominal value before each new break reception by setting SLVDUEN (UART_LINCTL[3]) register (It is recommend setting the SLVDUEN (UART_LINCTL[3]) bit before every checksum reception).

LIN Header Error Detection

In LIN Slave function mode, when user enables the header detection function by setting the SLVHDEN (UART_LINCTL[1]), hardware will handle the header detect flow. If the header has an error, the LIN header error flag SLVHEF (UART_LINSTS[1]) will be set and an interrupt is generated if the LINIEN (UART_INTEN[8]) bit is set. When header error is detected, user must reset the detect circuit to re-search a new frame header by writing 1 to SLVSYNCF (UART_LINSTS[3]) to re-search a new frame header.

The LIN header error flag SLVHEF (UART_LINSTS[1]) is set if one of the following conditions occurs:

- Break Delimiter is too short (less than 0.5-bit time).
- Frame error in sync field or Identifier field.
- The sync field data is not 0x55 (Non-Automatic Resynchronization mode).
- The sync field deviation error (With Automatic Resynchronization mode).
- The sync field measure time-out (With Automatic Resynchronization mode).
- LIN header reception time-out.

6.14.5.11 RS-485 Function Mode

Another alternate function of UART controller is RS-485 function (user must set UART_FUNCSEL [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.

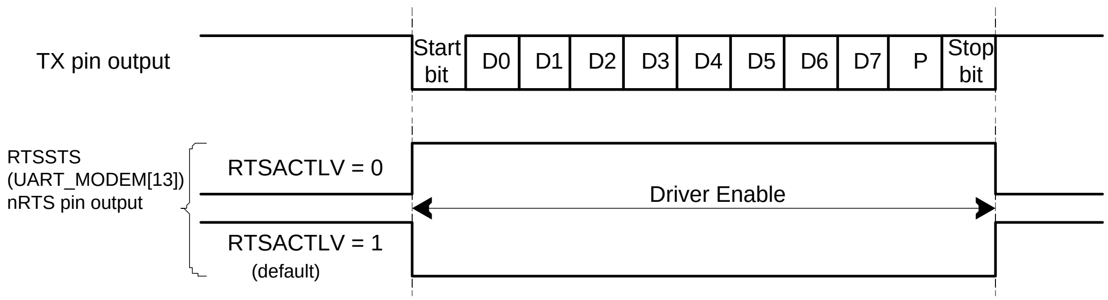
RS-485 Auto Direction Function (AUD)

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 6.14-25 demonstrates the RS-485 nRTS driving level in AUD mode. The nRTS pin will be automatically driven during TX data transmission.

Setting RTSACTLV(UART_MODEM[9]) can control nRTS pin output driving level. User can read the RTSSTS(UART_MODEM[13]) bit to get real nRTS pin output voltage logic status.

nRTS pin output status of RS-485 function mode (RS-485 AUD mode enabled)

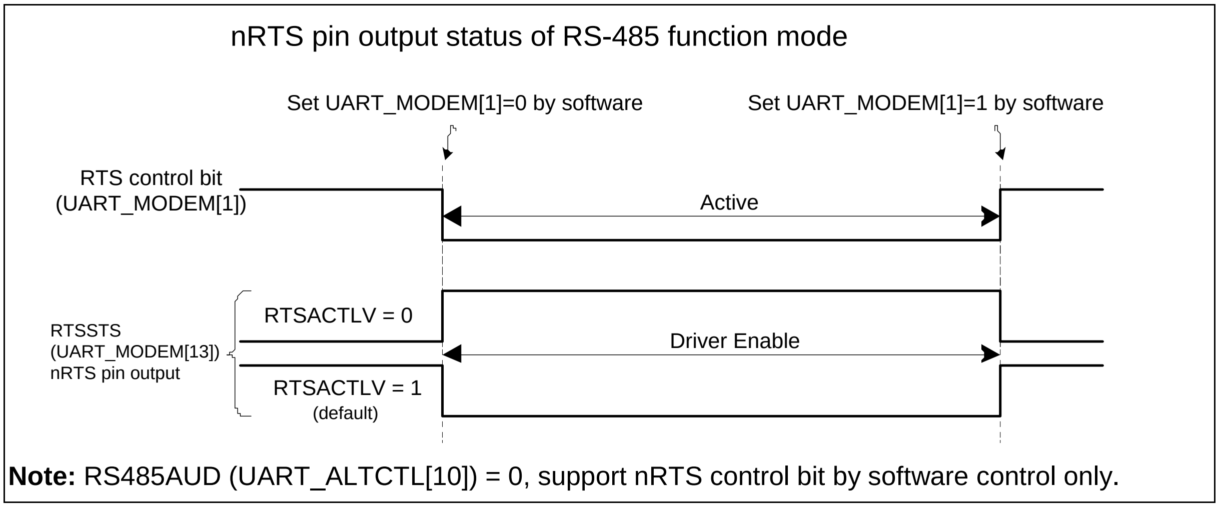


Note: RS485AUD(UART_ALTCTL[10]) = 1, the nRTS pin output by hardware control only.

Figure 6.14-25 RS-485 nRTS Driving Level in Auto Direction Mode

Figure 6.14-26 demonstrates the RS-485 nRTS driving level in software control (RS485AUD (UART_ALTCTL[10])=0). The nRTS driving level is controlled by programing the RTS(UART_MODEM[1]) control bit.

Setting RTSACTLV (UART_MODEM[9]) can control the nRTS pin output is inverse or non-inverse from RTS(UART_MODEM[1]) control bit. User can read the RTSSTS (UART_MODEM[13]) bit to get real nRTS pin output voltage logic status. The structure of RS-485 frame is shown in Figure 6.14-27.



Note: RS485AUD (UART_ALTCTL[10]) = 0, support nRTS control bit by software control only.

Figure 6.14-26 RS-485 nRTS Driving Level with Software Control

Programming Sequence Example:

1. Program FUNCSEL in UART_FUNCSEL to select RS-485 function.
2. Program the RXOFF (UART_FIFO[8]) to determine enable or disable the receiver RS-485 receiver.
3. Program the RS485NMM (UART_ALTCTL[8]) or RS485AAD (UART_ALTCTL[9]) mode.
4. If the RS485AAD (UART_ALTCTL[9]) mode is selected, the ADDRMV (UART_ALTCTL[31:24]) is programmed for auto address match value.
5. Determine auto direction control by programming RS485AUD (UART_ALTCTL[10]).

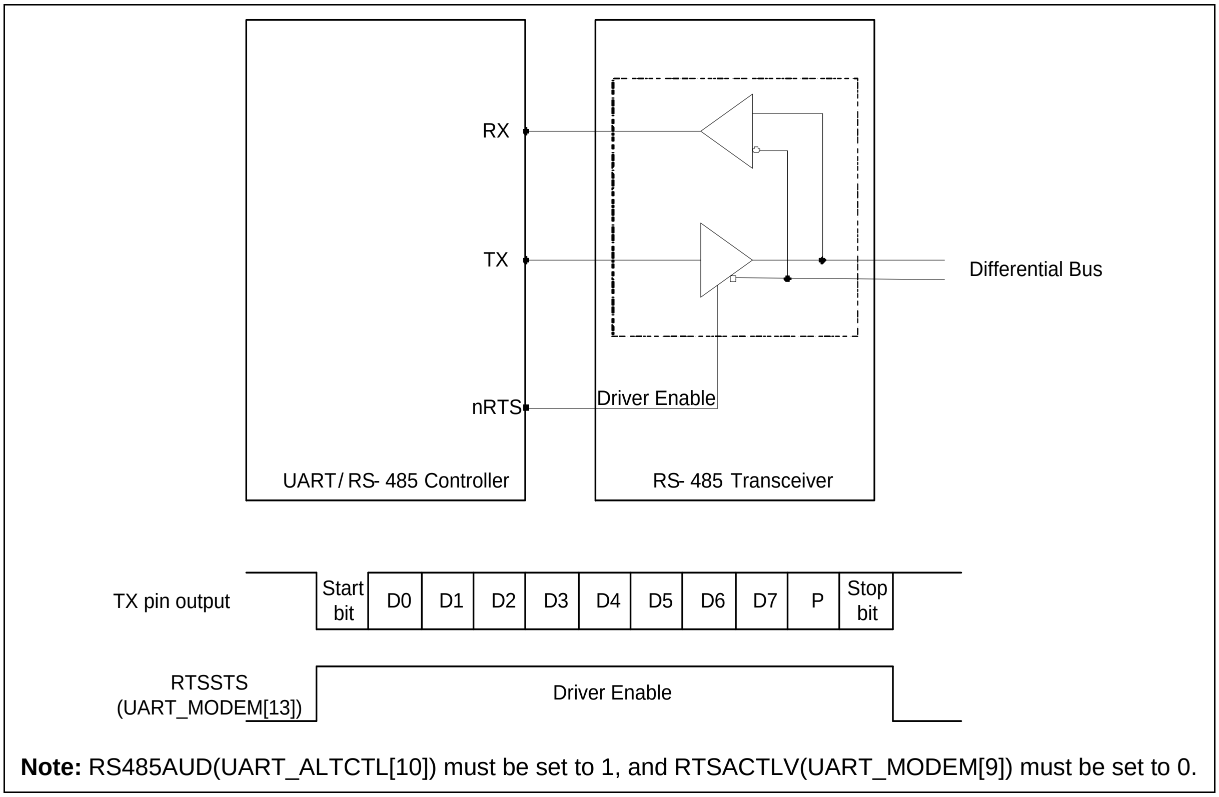


Figure 6.14-27 Structure of RS-485 Frame

6.14.5.12 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.

6.14.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
UART Base Address: **UARTx_BA = 0xB007_0000 + (0x1000 * x)** **x=0,1,2,3,4,5,6,7,8,9**				
UART_DAT **x=0,1,2,3,4,5,6,7,8,9**	UARTx_BA+0x00	R/W	UART Receive/Transmit Buffer Register	Undefined
UART_INTEN **x=0,1,2,3,4,5,6,7,8,9**	UARTx_BA+0x04	R/W	UART Interrupt Enable Register	0x0000_0000
UART_FIFO **x=0,1,2,3,4,5,6,7,8,9**	UARTx_BA+0x08	R/W	UART FIFO Control Register	0x0000_0101
UART_LINE **x=0,1,2,3,4,5,6,7,8,9**	UARTx_BA+0x0C	R/W	UART Line Control Register	0x0000_0000
UART_MODEM **x=0,1,2,3,4,5,6,7,8,9**	UARTx_BA+0x10	R/W	UART Modem Control Register	0x0000_0200
UART_MODEMSTS **x=0,1,2,3,4,5,6,7,8,9**	UARTx_BA+0x14	R/W	UART Modem Status Register	0x0000_0110
UART_FIFOSTS **x=0,1,2,3,4,5,6,7,8,9**	UARTx_BA+0x18	R/W	UART FIFO Status Register	0xB040_4000
UART_INTSTS **x=0,1,2,3,4,5,6,7,8,9**	UARTx_BA+0x1C	R/W	UART Interrupt Status Register	0x0040_0002
UART_TOUT **x=0,1,2,3,4,5,6,7,8,9**	UARTx_BA+0x20	R/W	UART Time-out Register	0x0000_0000
UART_BAUD **x=0,1,2,3,4,5,6,7,8,9**	UARTx_BA+0x24	R/W	UART Baud Rate Divider Register	0x0F00_0000
UART_IRDA **x=0,1,2,3,4,5,6,7,8,9**	UARTx_BA+0x28	R/W	UART IrDA Control Register	0x0000_0040
UART_ALTCTL **x=0,1,2,3,4,5,6,7,8,9**	UARTx_BA+0x2C	R/W	UART Alternate Control/Status Register	0x0000_000C

UART_FUNCSEL **x=0,1,2,3,4,5,6,7,8,9**	UARTx_BA+0x30	R/W	UART Function Select Register	0x0000_0000
UART_LINCTL **x=1,2**	UARTx_BA+0x34	R/W	UART LIN Control Register	0x000C_0000
UART_LINSTS **x=1,2**	UARTx_BA+0x38	R/W	UART LIN Status Register	0x0000_0000
UART_BRCOMP **x=0,1,2,3,4,5,6,7,8,9**	UARTx_BA+0x3C	R/W	UART Baud Rate Compensation Register	0x0000_0000
UART_WKCTL **x=0,1,2,3,4,5,6,7,8,9**	UARTx_BA+0x40	R/W	UART Wake-up Control Register	0x0000_0000
UART_WKSTS **x=0,1,2,3,4,5,6,7,8,9**	UARTx_BA+0x44	R/W	UART Wake-up Status Register	0x0000_0000
UART_DWKCOMP **x=0,1,2,3,4,5,6,7,8,9**	UARTx_BA+0x48	R/W	UART Incoming Data Wake-up Compensation Register	0x0000_0000

6.14.7 Register Description

UART Receive/Transmit Buffer Register (UART_DAT)

Register	Offset	R/W	Description	Reset Value
UART_DAT **x=0,1,2,3,4,5,6,7,8,9**	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=0,1,2,3,4,5,6,7,8,9**	UARTx_BA+0x04	R/W	UART Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	**TXENDIEN**	**Reserved**			**ABRIEN**	**Reserved**	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RXPDMAEN	**TXPDMAEN**	**ATOCTSEN**	**ATORTSEN**	**TOCNTEN**	**Reserved**		**LINIEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**WKIEN**	**BUFERRIEN**	**RXTOIEN**	**MODEMIEN**	**RLSIEN**	**THREIEN**	**RDAIEN**

Bits	Description	
[31:23]	**Reserved**	Reserved.
[22]	**TXENDIEN**	**Transmitter Empty Interrupt Enable Bit** If TXENDIEN (UART_INTEN[22]) is enabled, the Transmitter Empty interrupt TXENDINT (UART_INTSTS[30]) will be generated when TXENDIF (UART_INTSTS[22]) is set (TX FIFO (UART_DAT) is empty and the STOP bit of the last byte has been transmitted). 0 = Transmitter empty interrupt Disabled. 1 = Transmitter empty interrupt Enabled.
[21:19]	**Reserved**	Reserved.
[18]	**ABRIEN**	**Auto-baud Rate Interrupt Enable Bit** 0 = Auto-baud rate interrupt Disabled. 1 = Auto-baud rate interrupt Enabled.
[17:16]	**Reserved**	Reserved.
[15]	**RXPDMAEN**	**RX PDMA Enable Bit** This bit can enable or disable RX PDMA service. 0 = RX PDMA Disabled. 1 = RX PDMA Enabled. **Note:** If RLSIEN (UART_INTEN[2]) is enabled and HWRLSINT (UART_INTSTS[26]) is set to 1, the RLS (Receive Line Status) Interrupt is caused. If RLS interrupt is caused by Break Error Flag BIF(UART_FIFOSTS[6]), Frame Error Flag FEF(UART_FIFO[5]) or Parity Error Flag PEF(UART_FIFOSTS[4]), UART PDMA receive request operation is stopped. Clear Break Error Flag BIF or Frame Error Flag FEF or Parity Error Flag PEF by writing "1" to corresponding BIF, FEF and PEF to make UART PDMA receive request operation continue.
[14]	**TXPDMAEN**	**TX PDMA Enable Bit** 0 = TX PDMA Disabled. 1 = TX PDMA Enabled.

		Note: If RLSIEN (UART_INTEN[2]) is enabled and HWRLSINT (UART_INTSTS[26]) is set to 1, the RLS (Receive Line Status) Interrupt is caused. If RLS interrupt is caused by Break Error Flag BIF(UART_FIFOSTS[6]), Frame Error Flag FEF(UART_FIFO[5]) or Parity Error Flag PEF(UART_FIFOSTS[4]), UART PDMA transmit request operation is stopped. Clear Break Error Flag BIF or Frame Error Flag FEF or Parity Error Flag PEF by writing "1" to corresponding BIF, FEF and PEF to make UART PDMA transmit request operation continue.
[13]	**ATOCTSEN**	**nCTS Auto-flow Control Enable Bit** 0 = nCTS auto-flow control Disabled. 1 = nCTS auto-flow control Enabled. **Note:** When nCTS auto-flow is enabled, the UART will send data to external device if nCTS input assert (UART will not send data to device until nCTS is asserted).
[12]	**ATORTSEN**	**nRTS Auto-flow Control Enable Bit** 0 = nRTS auto-flow control Disabled. 1 = nRTS auto-flow control Enabled. **Note:** When nRTS auto-flow is enabled, if the number of bytes in the RX FIFO equals the RTSTRGLV (UART_FIFO[19:16]), the UART will de-assert nRTS signal.
[11]	**TOCNTEN**	**Receive Buffer Time-out Counter Enable Bit** 0 = Receive Buffer Time-out counter Disabled. 1 = Receive Buffer Time-out counter Enabled.
[10:9]	**Reserved**	Reserved.
[8]	**LINIEN**	**LIN Bus Interrupt Enable Bit** 0 = LIN bus interrupt Disabled. 1 = LIN bus interrupt Enabled. **Note:** This bit is used for LIN function mode.
[7]	**Reserved**	Reserved.
[6]	**WKIEN**	**Wake-up Interrupt Enable Bit** 0 = Wake-up Interrupt Disabled. 1 = Wake-up Interrupt Enabled.
[5]	**BUFERRIEN**	**Buffer Error Interrupt Enable Bit** 0 = Buffer error interrupt Disabled. 1 = Buffer error interrupt Enabled.
[4]	**RXTOIEN**	**RX Time-out Interrupt Enable Bit** 0 = RX time-out interrupt Disabled. 1 = RX time-out interrupt Enabled.
[3]	**MODEMIEN**	**Modem Status Interrupt Enable Bit** 0 = Modem status interrupt Disabled. 1 = Modem status interrupt Enabled.
[2]	**RLSIEN**	**Receive Line Status Interrupt Enable Bit** 0 = Receive Line Status interrupt Disabled. 1 = Receive Line Status interrupt Enabled.
[1]	**THREIEN**	**Transmit Holding Register Empty Interrupt Enable Bit** 0 = Transmit holding register empty interrupt Disabled. 1 = Transmit holding register empty interrupt Enabled.
[0]	**RDAIEN**	**Receive Data Available Interrupt Enable Bit**

		0 = Receive data available interrupt Disabled. 1 = Receive data available interrupt Enabled.

UART FIFO Control Register (UART_FIFO)

Register	Offset	R/W	Description	Reset Value
UART_FIFO **x=0,1,2,3,4,5,6,7,8,9**	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. **Note1:** This bit will automatically clear at least 3 UART peripheral clock cycles. **Note2:** Before setting this bit, it should wait for the TXEMPTYF (UART_FIFOSTS[28]) be set.
[1]	**RXRST**	**RX Field Software Reset** When RXRST (UART_FIFO[1]) is set, all the byte in the receiver FIFO and RX internal state machine are cleared. 0 = No effect. 1 = Reset the RX internal state machine and pointers. **Note1:** This bit will automatically clear at least 3 UART peripheral clock cycles. **Note2:** Before setting this bit, it should wait for the RXIDLE (UART_FIFOSTS[29]) be set.
[0]	**Reserved**	Reserved.

UART Line Control Register (UART_LINE)

Register	Offset	R/W	Description	Reset Value
UART_LINE **x=0,1,2,3,4,5,6,7,8,9**	UARTx_BA+0x0C	R/W	UART Line Control Register	0x0000_0000

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**RXDINV**	**TXDINV**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PSS	**BCB**	**SPE**	**EPE**	**PBE**	**NSB**	**WLS**	

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**RXDINV**	**RX Data Inverted** 0 = Received data signal inverted Disabled. 1 = Received data signal inverted Enabled. **Note1:** Before setting this bit, TXRXDIS (UART_FUNCSEL[3]) should be set then waited for TXRXACT (UART_FIFOSTS[31]) is cleared. When the configuration is done, cleared TXRXDIS (UART_FUNCSEL[3]) to activate UART controller. **Note2:** This bit is valid when FUNCSEL (UART_FUNCSEL[1:0]) is select UART, LIN or RS485 function.
[8]	**TXDINV**	**TX Data Inverted** 0 = Transmitted data signal inverted Disabled. 1 = Transmitted data signal inverted Enabled. **Note1:** Before setting this bit, TXRXDIS (UART_FUNCSEL[3]) should be set then waited for TXRXACT (UART_FIFOSTS[31]) is cleared. When the configuration is done, cleared TXRXDIS (UART_FUNCSEL[3]) to activate UART controller. **Note2:** This bit is valid when FUNCSEL (UART_FUNCSEL[1:0]) is select UART, LIN or RS485 function.
[7]	**PSS**	**Parity Bit Source Selection** The parity bit can be selected to be generated and checked automatically or by software. 0 = Parity bit is generated by EPE (UART_LINE[4]) and SPE (UART_LINE[5]) setting and checked automatically. 1 = Parity bit generated and checked by software. **Note1:** This bit has effect only when PBE (UART_LINE[3]) is set. **Note2:** If PSS is 0, the parity bit is transmitted and checked automatically. If PSS is 1, the transmitted parity bit value can be determined by writing PARITY (UART_DAT[8]) and the parity bit can be read by reading PARITY (UART_DAT[8]).
[6]	**BCB**	**Break Control Bit**

		0 = Break Control Disabled. 1 = Break Control Enabled. **Note:** When this bit is set to logic 1, the transmitted serial data output (TX) is forced to the Spacing State (logic 0). This bit acts only on TX line and has no effect on the transmitter logic.
[5]	**SPE**	**Stick Parity Enable Bit** 0 = Stick parity Disabled. 1 = Stick parity Enabled. **Note:** If PBE (UART_LINE[3]) and EPE (UART_LINE[4]) are logic 1, the parity bit is transmitted and checked as logic 0. If PBE (UART_LINE[3]) is 1 and EPE (UART_LINE[4]) is 0 then the parity bit is transmitted and checked as 1.
[4]	**EPE**	**Even Parity Enable Bit** 0 = Odd number of logic 1's is transmitted and checked in each word. 1 = Even number of logic 1's is transmitted and checked in each word. **Note:** This bit has effect only when PBE (UART_LINE[3]) is set.
[3]	**PBE**	**Parity Bit Enable Bit** 0 = Parity bit generated Disabled. 1 = Parity bit generated Enabled. **Note:** Parity bit is generated on each outgoing character and is checked on each incoming data.
[2]	**NSB**	**Number of "STOP Bit"** 0 = One "STOP bit" is generated in the transmitted data. 1 = When select 5-bit word length, 1.5 "STOP bit" is generated in the transmitted data. When select 6-, 7- and 8-bit word length, 2 "STOP bit" is generated in the transmitted data.
[1:0]	**WLS**	**Word Length Selection** This field sets UART word length. 00 = 5 bits. 01 = 6 bits. 10 = 7 bits. 11 = 8 bits.

UART Modem Control Register (UART_MODEM)

Register	Offset	R/W	Description	Reset Value
UART_MODEM **x=0,1,2,3,4,5,6,7,8,9**	UARTx_BA+0x10	R/W	UART Modem Control Register	0x0000_0200

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

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13]	**RTSSTS**	**nRTS Pin Status (Read Only)** This bit mirror from nRTS pin output of voltage logic status. 0 = nRTS pin output is low level voltage logic state. 1 = nRTS pin output is high level voltage logic state.
[12:10]	**Reserved**	Reserved.
[9]	**RTSACTLV**	**nRTS Pin Active Level** This bit defines the active level state of nRTS pin output. 0 = nRTS pin output is high level active. 1 = nRTS pin output is low level active. (Default) **Note1:** Refer to Figure 6.14-14 and Figure 6.14-15 for UART function mode. **Note2:** Refer to Figure 6.14-25 and Figure 6.14-26 for RS-485 function mode. **Note3:** Before setting this bit, TXRXDIS (UART_FUNCSEL[3]) should be set then waited for TXRXACT (UART_FIFOSTS[31]) is cleared. When the configuration is done, cleared TXRXDIS (UART_FUNCSEL[3]) to activate UART controller.
[8:2]	**Reserved**	Reserved.
[1]	**RTS**	**nRTS (Request-to-send) Signal Control** This bit is direct control internal nRTS signal active or not, and then drive the nRTS pin output with RTSACTLV bit configuration. 0 = nRTS signal is active. 1 = nRTS signal is inactive. **Note1:** The nRTS signal control bit is not effective when nRTS auto-flow control is enabled in UART function mode. **Note2:** The nRTS signal control bit is not effective when RS-485 auto direction mode (AUD) is enabled in RS-485 function mode.

[0]	**Reserved**	Reserved.

UART Modem Status Register (UART_MODEMSTS)

Register	Offset	R/W	Description	Reset Value
UART_MODEMSTS **x=0,1,2,3,4,5,6,7,8,9**	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=0,1,2,3,4,5,6,7,8,9**	UARTx_BA+0x18	R/W	UART FIFO Status Register	0xB040_4000

31	30	29	28	27	26	25	24
TXRXACT	Reserved	RXIDLE	TXEMPTYF	Reserved			TXOVIF
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TXFULL	TXEMPTY	TXPTR					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RXFULL	RXEMPTY	RXPTR					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	BIF	FEF	PEF	ADDRDETF	ABRDTOIF	ABRDIF	RXOVIF

Bits	Description	
[31]	**TXRXACT**	**TX and RX Active Status (Read Only)** This bit indicates TX and RX are active or inactive. 0 = TX and RX are inactive. 1 = TX and RX are active. (Default) **Note:** When TXRXDIS (UART_FUNCSEL[3]) is set and both TX and RX are in idle state, this bit is cleared. The UART controller can not transmit or receive data at this moment. Otherwise this bit is set.
[30]	**Reserved**	Reserved.
[29]	**RXIDLE**	**RX Idle Status (Read Only)** This bit is set by hardware when RX is idle. 0 = RX is busy. 1 = RX is idle. (Default)
[28]	**TXEMPTYF**	**Transmitter Empty Flag (Read Only)** This bit is set by hardware when TX FIFO (UART_DAT) is empty and the STOP bit of the last byte has been transmitted. 0 = TX FIFO is not empty or the STOP bit of the last byte has been not transmitted. 1 = TX FIFO is empty and the STOP bit of the last byte has been transmitted. **Note:** This bit is cleared automatically when TX FIFO is not empty or the last byte transmission has not completed.
[27:25]	**Reserved**	Reserved.
[24]	**TXOVIF**	**TX Overflow Error Interrupt Flag** If TX FIFO (UART_DAT) is full, an additional write to UART_DAT will cause this bit to logic 1. 0 = TX FIFO is not overflow. 1 = TX FIFO is overflow.

		Note: This bit can be cleared by writing "1" to it.
[23]	**TXFULL**	**Transmitter FIFO Full (Read Only)** This bit indicates TX FIFO full or not. 0 = TX FIFO is not full. 1 = TX FIFO is full. **Note:** This bit is set when the number of usage in TX FIFO Buffer is equal to 16, otherwise it is cleared by hardware.
[22]	**TXEMPTY**	**Transmitter FIFO Empty (Read Only)** This bit indicates TX FIFO empty or not. 0 = TX FIFO is not empty. 1 = TX FIFO is empty. **Note:** When the last byte of TX FIFO has been transferred to Transmitter Shift Register, hardware sets this bit high. It will be cleared when writing data into UART_DAT (TX FIFO not empty).
[21:16]	**TXPTR**	**TX FIFO Pointer (Read Only)** This field indicates the TX FIFO Buffer Pointer. When CPU writes one byte into UART_DAT, TXPTR increases one. When one byte of TX FIFO is transferred to Transmitter Shift Register, TXPTR decreases one. The Maximum value shown in TXPTR is 15. When the using level of TX FIFO Buffer equal to 16, the TXFULL bit is set to 1 and TXPTR will show 0. As one byte of TX FIFO is transferred to Transmitter Shift Register, the TXFULL bit is cleared to 0 and TXPTR will show 15.
[15]	**RXFULL**	**Receiver FIFO Full (Read Only)** This bit initiates RX FIFO full or not. 0 = RX FIFO is not full. 1 = RX FIFO is full. **Note:** This bit is set when the number of usage in RX FIFO Buffer is equal to 16, otherwise it is cleared by hardware.
[14]	**RXEMPTY**	**Receiver FIFO Empty (Read Only)** This bit initiate RX FIFO empty or not. 0 = RX FIFO is not empty. 1 = RX FIFO is empty. **Note:** When the last byte of RX FIFO has been read by CPU, hardware sets this bit high. It will be cleared when UART receives any new data.
[13:8]	**RXPTR**	**RX FIFO Pointer (Read Only)** This field indicates the RX FIFO Buffer Pointer. When UART receives one byte from external device, RXPTR increases one. When one byte of RX FIFO is read by CPU, RXPTR decreases one. The Maximum value shown in RXPTR is 15. When the using level of RX FIFO Buffer equal to 16, the RXFULL bit is set to 1 and RXPTR will show 0. As one byte of RX FIFO is read by CPU, the RXFULL bit is cleared to 0 and RXPTR will show 15.
[7]	**Reserved**	Reserved.
[6]	**BIF**	**Break Interrupt Flag** This bit is set to logic 1 whenever the received data input (RX) is held in the "spacing state" (logic 0) for longer than a full word transmission time (that is, the total time of "start bit" + data bits + parity + stop bits). 0 = No Break interrupt is generated. 1 = Break interrupt is generated. **Note:** This bit can be cleared by writing "1" to it.

[5]	**FEF**	**Framing Error Flag** This bit is set to logic 1 whenever the received character does not have a valid “stop bit” (that is, the stop bit following the last data bit or parity bit is detected as logic 0). 0 = No framing error is generated. 1 = Framing error is generated. **Note:** This bit can be cleared by writing “1” to it.
[4]	**PEF**	**Parity Error Flag** This bit is set to logic 1 whenever the received character does not have a valid “parity bit”. 0 = No parity error is generated. 1 = Parity error is generated. **Note:** This bit can be cleared by writing “1” to it.
[3]	**ADDRDETF**	**RS-485 Address Byte Detect Flag** 0 = Receiver detects a data that is not an address bit (bit 9 =’0’). 1 = Receiver detects a data that is an address bit (bit 9 =’1’). **Note1:** This field is used for RS-485 function mode and ADDRDEN (UART_ALTCTL[15]) is set to 1 to enable Address detection mode. **Note2:** This bit can be cleared by writing “1” to it.
[2]	**ABRDTOIF**	**Auto-baud Rate Detect Time-out Interrupt Flag** This bit is set to logic “1” in Auto-baud Rate Detect mode when the baud rate counter is overflow. 0 = Auto-baud rate counter is underflow. 1 = Auto-baud rate counter is overflow. **Note:** This bit can be cleared by writing “1” to it.
[1]	**ABRDIF**	**Auto-baud Rate Detect Interrupt Flag** This bit is set to logic “1” when auto-baud rate detect function is finished. 0 = Auto-baud rate detect function is not finished. 1 = Auto-baud rate detect function is finished. **Note:** This bit can be cleared by writing “1” to it.
[0]	**RXOVIF**	**RX Overflow Error Interrupt Flag** This bit is set when RX FIFO overflow. If the number of bytes of received data is greater than RX_FIFO (UART_DAT) size 16 bytes, this bit will be set. 0 = RX FIFO is not overflow. 1 = RX FIFO is overflow. **Note:** This bit can be cleared by writing “1” to it.

UART Interrupt Status Register (UART_INTSTS)

Register	Offset	R/W	Description	Reset Value
UART_INTSTS **x=0,1,2,3,4,5,6,7, 8,9**	UARTx_BA+0x1C	R/W	UART Interrupt Status Register	0x0040_0002

31	30	29	28	27	26	25	24
ABRINT	TXENDINT	HWBUFEINT	HWTOINT	HWMODINT	HWRLSINT	Reserved	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	TXENDIF	HWBUFEIF	HWTOIF	HWMODIF	HWRLSIF	Reserved	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
LININT	WKINT	BUFERRINT	RXTOINT	MODEMINT	RLSINT	THREINT	RDAINT
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LINIF	WKIF	BUFERRIF	RXTOIF	MODEMIF	RLSIF	THREIF	RDAIF

Bits	Description	
[31]	**ABRINT**	**Auto-baud Rate Interrupt Indicator (Read Only)** This bit is set if ABRIEN (UART_INTEN[18]) and ABRIF (UART_ALTCTL[17]) are both set to 1. 0 = No Auto-baud Rate interrupt is generated. 1 = The Auto-baud Rate interrupt is generated.
[30]	**TXENDINT**	**Transmitter Empty Interrupt Indicator (Read Only)** This bit is set if TXENDIEN (UART_INTEN[22]) and TXENDIF(UART_INTSTS[22]) are both set to 1. 0 = No Transmitter Empty interrupt is generated. 1 = Transmitter Empty interrupt is generated.
[29]	**HWBUFEINT**	**PDMA Mode Buffer Error Interrupt Indicator (Read Only)** This bit is set if BUFERRIEN (UART_INTEN[5]) and HWBUFEIF (UART_INTSTS[21]) are both set to 1. 0 = No buffer error interrupt is generated in PDMA mode. 1 = Buffer error interrupt is generated in PDMA mode.
[28]	**HWTOINT**	**PDMA Mode RX Time-out Interrupt Indicator (Read Only)** This bit is set if RXTOIEN (UART_INTEN[4]) and HWTOIF(UART_INTSTS[20]) are both set to 1. 0 = No RX time-out interrupt is generated in PDMA mode. 1 = RX time-out interrupt is generated in PDMA mode.
[27]	**HWMODINT**	**PDMA Mode MODEM Status Interrupt Indicator (Read Only)** This bit is set if MODEMIEN (UART_INTEN[3]) and HWMODIF(UART_INTSTS[19]) are both set to 1. 0 = No Modem interrupt is generated in PDMA mode. 1 = Modem interrupt is generated in PDMA mode.

[26]	**HWRLSINT**	**PDMA Mode Receive Line Status Interrupt Indicator (Read Only)** This bit is set if RLSIEN (UART_INTEN[2]) and HWRLSIF(UART_INTSTS[18]) are both set to 1. 0 = No RLS interrupt is generated in PDMA mode. 1 = RLS interrupt is generated in PDMA mode.
[25:23]	**Reserved**	Reserved.
[22]	**TXENDIF**	**Transmitter Empty Interrupt Flag** This bit is set when TX FIFO (UART_DAT) is empty and the STOP bit of the last byte has been transmitted (TXEMPTYF (UART_FIFOSTS[28]) is set). If TXENDIEN (UART_INTEN[22]) is enabled, the Transmitter Empty interrupt will be generated. 0 = No transmitter empty interrupt flag is generated. 1 = Transmitter empty interrupt flag is generated. **Note:** This bit is cleared automatically when TX FIFO is not empty or the last byte transmission has not completed.
[21]	**HWBUFEIF**	**PDMA Mode Buffer Error Interrupt Flag (Read Only)** This bit is set when the TX or RX FIFO overflows (TXOVIF (UART_FIFOSTS [24]) or RXOVIF (UART_FIFOSTS[0]) is set). When BUFERRIF (UART_INTSTS[5]) is set, the transfer maybe is not correct. If BUFERRIEN (UART_INTEN [5]) is enabled, the buffer error interrupt will be generated. 0 = No buffer error interrupt flag is generated in PDMA mode. 1 = Buffer error interrupt flag is generated in PDMA mode. **Note:** This bit is cleared when both TXOVIF (UART_FIFOSTS[24]]) and RXOVIF (UART_FIFOSTS[0]) are cleared.
[20]	**HWTOIF**	**PDMA Mode RX Time-out Interrupt Flag (Read Only)** This bit is set when the RX FIFO is not empty and no activities occurred in the RX FIFO and the time-out counter equal to TOIC (UART_TOUT[7:0]). If RXTOIEN (UART_INTEN [4]) is enabled, the RX time-out interrupt will be generated. 0 = No RX time-out interrupt flag is generated in PDMA mode. 1 = RX time-out interrupt flag is generated in PDMA mode. **Note:** This bit is read only and user can read UART_DAT (RX is in active) to clear it.
[19]	**HWMODIF**	**PDMA Mode MODEM Interrupt Flag (Read Only)** This bit is set when the nCTS pin has state change (CTSDETF (UART_MODEMSTS [0] =1)). If MODEMIEN (UART_INTEN [3]) is enabled, the Modem interrupt will be generated. 0 = No Modem interrupt flag is generated in PDMA mode. 1 = Modem interrupt flag is generated in PDMA mode. **Note:** This bit is read only and reset to 0 when the bit CTSDETF (UART_MODEMSTS[0]) is cleared by writing 1 on CTSDETF (UART_MODEMSTS [0]).
[18]	**HWRLSIF**	**PDMA Mode Receive Line Status Flag (Read Only)** This bit is set when the RX receive data have parity error, frame error or break error (at least one of 3 bits, BIF (UART_FIFOSTS[6]), FEF (UART_FIFOSTS[5]) and PEF (UART_FIFOSTS[4]) is set). If RLSIEN (UART_INTEN [2]) is enabled, the RLS interrupt will be generated. 0 = No RLS interrupt flag is generated in PDMA mode. 1 = RLS interrupt flag is generated in PDMA mode. **Note1:** In RS-485 function mode, this field include “receiver detect any address byte received address byte character (bit9 = ‘1’) bit". **Note2:** In UART function mode, this bit is read only and reset to 0 when all bits of BIF(UART_FIFOSTS[6]), FEF(UART_FIFOSTS[5]) and PEF(UART_FIFOSTS[4]) are cleared. **Note3:** In RS-485 function mode, this bit is read only and reset to 0 when all bits of

		BIF(UART_FIFOSTS[6]), FEF(UART_FIFOSTS[5]), PEF(UART_FIFOSTS[4]) and ADDRDETF (UART_FIFOSTS[3]) are cleared.
[17:16]	**Reserved**	Reserved.
[15]	**LININT**	**LIN Bus Interrupt Indicator (Read Only)** This bit is set if LINIEN (UART_INTEN[8]) and LINIF(UART_INTSTS[7]) are both set to 1. 0 = No LIN Bus interrupt is generated. 1 = The LIN Bus interrupt is generated.
[14]	**WKINT**	**UART Wake-up Interrupt Indicator (Read Only)** This bit is set if WKIEN (UART_INTEN[6]) and WKIF (UART_INTSTS[6]) are both set to 1. 0 = No UART wake-up interrupt is generated. 1 = UART wake-up interrupt is generated.
[13]	**BUFERRINT**	**Buffer Error Interrupt Indicator (Read Only)** This bit is set if BUFERRIEN(UART_INTEN[5]) and BUFERRIF(UART_ INTSTS[5]) are both set to 1. 0 = No buffer error interrupt is generated. 1 = Buffer error interrupt is generated.
[12]	**RXTOINT**	**RX Time-out Interrupt Indicator (Read Only)** This bit is set if RXTOIEN (UART_INTEN[4]) and RXTOIF(UART_INTSTS[4]) are both set to 1. 0 = No RX time-out interrupt is generated. 1 = RX time-out interrupt is generated.
[11]	**MODEMINT**	**MODEM Status Interrupt Indicator (Read Only)** This bit is set if MODEMIEN(UART_INTEN[3]) and MODEMIF(UART_INTSTS[3]) are both set to 1 0 = No Modem interrupt is generated. 1 = Modem interrupt is generated..
[10]	**RLSINT**	**Receive Line Status Interrupt Indicator (Read Only)** This bit is set if RLSIEN (UART_INTEN[2]) and RLSIF(UART_INTSTS[2]) are both set to 1. 0 = No RLS interrupt is generated. 1 = RLS interrupt is generated.
[9]	**THREINT**	**Transmit Holding Register Empty Interrupt Indicator (Read Only)** This bit is set if THREIEN (UART_INTEN[1]) and THREIF(UART_INTSTS[1]) are both set to 1. 0 = No THRE interrupt is generated. 1 = THRE interrupt is generated.
[8]	**RDAINT**	**Receive Data Available Interrupt Indicator (Read Only)** This bit is set if RDAIEN (UART_INTEN[0]) and RDAIF (UART_INTSTS[0]) are both set to 1. 0 = No RDA interrupt is generated. 1 = RDA interrupt is generated.
[7]	**LINIF**	**LIN Bus Interrupt Flag** This bit is set when LIN slave header detect (SLVHDETF (UART_LINSTS[0] =1)), LIN break detect (BRKDETF(UART_LINSTS[8]=1)), bit error detect (BITEF(UART_LINSTS[9]=1)), LIN slave ID parity error (SLVIDPEF(UART_LINSTS[2] = 1)) or LIN slave header error detect (SLVHEF (UART_LINSTS[1])). If LINIEN

		(UART_INTEN [8]) is enabled the LIN interrupt will be generated. 0 = None of SLVHDETF, BRKDETF, BITEF, SLVIDPEF and SLVHEF is generated. 1 = At least one of SLVHDETF, BRKDETF, BITEF, SLVIDPEF and SLVHEF is generated. **Note:** This bit is cleared when SLVHDETF(UART_LINSTS[0]), BRKDETF(UART_LINSTS[8]), BITEF(UART_LINSTS[9]), SLVIDPEF (UART_LINSTS[2]) and SLVHEF(UART_LINSTS[1]) all are cleared and software writing '1' to LINIF(UART_INTSTS[7]).
[6]	**WKIF**	**UART Wake-up Interrupt Flag (Read Only)** This bit is set when TOUTWKF (UART_WKSTS[4]), RS485WKF (UART_WKSTS[3]), RFRTWKF (UART_WKSTS[2]), DATWKF (UART_WKSTS[1]) or CTSWKF(UART_WKSTS[0]) is set to 1. 0 = No UART wake-up interrupt flag is generated. 1 = UART wake-up interrupt flag is generated. **Note:** This bit is cleared if all of TOUTWKF, RS485WKF, RFRTWKF, DATWKF and CTSWKF are cleared to 0 by writing 1 to the corresponding interrupt flag.
[5]	**BUFERRIF**	**Buffer Error Interrupt Flag (Read Only)** This bit is set when the TX FIFO or RX FIFO overflows (TXOVIF (UART_FIFOSTS[24]) or RXOVIF (UART_FIFOSTS[0]) is set). When BUFERRIF (UART_INTSTS[5]) is set, the transfer is not correct. If BUFERRIEN (UART_INTEN [5]) is enabled, the buffer error interrupt will be generated. 0 = No buffer error interrupt flag is generated. 1 = Buffer error interrupt flag is generated. **Note:** This bit is cleared if both of RXOVIF(UART_FIFOSTS[0]) and TXOVIF(UART_FIFOSTS[24]) are cleared to 0 by writing 1 to RXOVIF(UART_FIFOSTS[0]) and TXOVIF(UART_FIFOSTS[24]).
[4]	**RXTOIF**	**RX Time-out Interrupt Flag (Read Only)** This bit is set when the RX FIFO is not empty and no activities occurred in the RX FIFO and the time-out counter equal to TOIC (UART_TOUT[7:0]). If RXTOIEN (UART_INTEN [4]) is enabled, the RX time-out interrupt will be generated. 0 = No RX time-out interrupt flag is generated. 1 = RX time-out interrupt flag is generated. **Note:** This bit is read only and user can read UART_DAT (RX is in active) to clear it.
[3]	**MODEMIF**	**MODEM Interrupt Flag (Read Only)** This bit is set when the nCTS pin has state change (CTSDETF (UART_MODEMSTS[0]) = 1). If MODEMIEN (UART_INTEN [3]) is enabled, the Modem interrupt will be generated. 0 = No Modem interrupt flag is generated. 1 = Modem interrupt flag is generated. **Note:** This bit is read only and reset to 0 when bit CTSDETF is cleared by a write 1 on CTSDETF(UART_MODEMSTS[0]).
[2]	**RLSIF**	**Receive Line Interrupt Flag (Read Only)** This bit is set when the RX receive data have parity error, frame error or break error (at least one of 3 bits, BIF(UART_FIFOSTS[6]), FEF(UART_FIFOSTS[5]) and PEF(UART_FIFOSTS[4]), is set). If RLSIEN (UART_INTEN [2]) is enabled, the RLS interrupt will be generated. 0 = No RLS interrupt flag is generated. 1 = RLS interrupt flag is generated. **Note1:** In RS-485 function mode, this field is set include "receiver detect and received address byte character (bit9 = '1') bit". At the same time, the bit of ADDRDETF (UART_FIFOSTS[3]) is also set. **Note2:** This bit is read only and reset to 0 when all bits of BIF (UART_FIFOSTS[6]),

		FEF(UART_FIFOSTS[5]) and PEF(UART_FIFOSTS[4]) are cleared. **Note3:** In RS-485 function mode, this bit is read only and reset to 0 when all bits of BIF (UART_FIFOSTS[6]), FEF(UART_FIFOSTS[5]), PEF(UART_FIFOSTS[4]) and ADDRDETF (UART_FIFOSTS[3]) are cleared.
[1]	**THREIF**	**Transmit Holding Register Empty Interrupt Flag** This bit is set when the last data of TX FIFO is transferred to Transmitter Shift Register. If THREIEN (UART_INTEN[1]) is enabled, the THRE interrupt will be generated. 0 = No THRE interrupt flag is generated. 1 = THRE interrupt flag is generated. **Note:** This bit is read only and it will be cleared when writing data into UART_DAT (TX FIFO not empty).
[0]	**RDAIF**	**Receive Data Available Interrupt Flag** When the number of bytes in the RX FIFO equals the RFITL then the RDAIF(UART_INTSTS[0]) will be set. If RDAIEN (UART_INTEN [0]) is enabled, the RDA interrupt will be generated. 0 = No RDA interrupt flag is generated. 1 = RDA interrupt flag is generated. **Note:** This bit is read only and it will be cleared when the number of unread bytes of RX FIFO drops below the threshold level (RFITL(UART_FIFO[7:4]).

UART Time-out Register (UART_TOUT)

Register	Offset	R/W	Description	Reset Value
UART_TOUT **x=0,1,2,3,4,5,6, 7,8,9**	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=0,1,2,3,4,5,6,7,8,9**	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 detailed description is shown in Table 6.14-4. **Note:** In IrDA mode must be operated in mode 0.
[28]	**BAUDM0**	**BAUD Rate Mode Selection Bit 0** This bit is baud rate mode selection bit 0. UART provides three baud rate calculation modes. This bit combines with BAUDM1 (UART_BAUD[29]) to select baud rate calculation mode. The detailed description is shown in Table 6.14-4.
[27:24]	**EDIVM1**	**Extra Divider for BAUD Rate Mode 1** This field is used for baud rate calculation in mode 1 and has no effect for baud rate calculation in mode 0 and mode 2. The detailed description is shown in Table 6.14-4.
[23:16]	**Reserved**	Reserved.
[15:0]	**BRD**	**Baud Rate Divider** The field indicates the baud rate divider. This filed is used in baud rate calculation. The detailed description is shown in Table 6.14-4.

UART IrDA Control Register (UART_IRDA)

Register	Offset	R/W	Description	Reset Value
UART_IRDA **x=0,1,2,3,4,5,6,7,8,9**	UARTx_BA+0x28	R/W	UART IrDA Control Register	0x0000_0040

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

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**RXINV**	**IrDA Inverse Receive Input Signal** 0 = None inverse receiving input signal. 1 = Inverse receiving input signal. (Default) **Note1:** Before setting this bit, TXRXDIS (UART_FUNCSEL[3]) should be set then waited for TXRXACT (UART_FIFOSTS[31]) is cleared. When the configuration is done, cleared TXRXDIS (UART_FUNCSEL[3]) to activate UART controller. **Note2:** This bit is valid when FUNCSEL (UART_FUNCSEL[1:0]) is select IrDA function.
[5]	**TXINV**	**IrDA Inverse Transmitting Output Signal** 0 = None inverse transmitting signal. (Default). 1 = Inverse transmitting output signal. **Note1:** Before setting this bit, TXRXDIS (UART_FUNCSEL[3]) should be set then waited for TXRXACT (UART_FIFOSTS[31]) is cleared. When the configuration is done, cleared TXRXDIS (UART_FUNCSEL[3]) to activate UART controller. **Note2:** This bit is valid when FUNCSEL (UART_FUNCSEL[1:0]) is select IrDA function.
[4:2]	**Reserved**	Reserved.
[1]	**TXEN**	**IrDA Receiver/Transmitter Selection Enable Bit** 0 = IrDA Transmitter Disabled and Receiver Enabled. (Default) 1 = IrDA Transmitter Enabled and Receiver Disabled.
[0]	**Reserved**	Reserved.

UART Alternate Control/Status Register (UART_ALTCTL)

Register	Offset	R/W	Description	Reset Value
UART_ALTCTL **x=0,1,2,3,4,5,6,7,8,9**	UARTx_BA+0x2C	R/W	UART Alternate Control/Status Register	0x0000_000C

31	30	29	28	27	26	25	24
ADDRMV							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved			ABRDBITS		ABRDEN	ABRIF	Reserved
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ADDRDEN	Reserved				RS485AUD	RS485AAD	RS485NMM
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LINTXEN	LINRXEN	Reserved		BRKFL			

Bits	Description	
[31:24]	**ADDRMV**	**Address Match Value** This field contains the RS-485 address match values. **Note:** This field is used for RS-485 auto address detection mode.
[23:21]	**Reserved**	Reserved.
[20:19]	**ABRDBITS**	**Auto-baud Rate Detect Bit Length** 00 = 1-bit time from Start bit to the 1st rising edge. The input pattern shall be 0x01. 01 = 2-bit time from Start bit to the 1st rising edge. The input pattern shall be 0x02. 10 = 4-bit time from Start bit to the 1st rising edge. The input pattern shall be 0x08. 11 = 8-bit time from Start bit to the 1st rising edge. The input pattern shall be 0x80. **Note :** The calculation of bit number includes the START bit.
[18]	**ABRDEN**	**Auto-baud Rate Detect Enable Bit** 0 **=** Auto-baud rate detect function Disabled. 1 = Auto-baud rate detect function Enabled. **Note :** This bit is cleared automatically after auto-baud detection is finished.
[17]	**ABRIF**	**Auto-baud Rate Interrupt Flag (Read Only)** This bit is set when auto-baud rate detection function finished or the auto-baud rate counter was overflow and if ABRIEN(UART_INTEN [18]) is set then the auto-baud rate interrupt will be generated. 0 = No auto-baud rate interrupt flag is generated. 1 = Auto-baud rate interrupt flag is generated. **Note:** This bit is read only, but it can be cleared by writing “1” to ABRDTOIF (UART_FIFOSTS[2]) and ABRDIF(UART_FIFOSTS[1]).
[16]	**Reserved**	Reserved.
[15]	**ADDRDEN**	**RS-485 Address Detection Enable Bit**

		This bit is used to enable RS-485 Address Detection mode. 0 = Address detection mode Disabled. 1 = Address detection mode Enabled. **Note:** This bit is used for RS-485 any operation mode.
[14:11]	**Reserved**	Reserved.
[10]	**RS485AUD**	**RS-485 Auto Direction Function (AUD)** 0 = RS-485 Auto Direction Operation function (AUD) Disabled. 1 = RS-485 Auto Direction Operation function (AUD) Enabled. **Note:** It can be active with RS-485_AAD or RS-485_NMM operation mode.
[9]	**RS485AAD**	**RS-485 Auto Address Detection Operation Mode (AAD)** 0 = RS-485 Auto Address Detection Operation mode (AAD) Disabled. 1 = RS-485 Auto Address Detection Operation mode (AAD) Enabled. **Note:** It cannot be active with RS-485_NMM operation mode.
[8]	**RS485NMM**	**RS-485 Normal Multi-drop Operation Mode (NMM)** 0 = RS-485 Normal Multi-drop Operation mode (NMM) Disabled. 1 = RS-485 Normal Multi-drop Operation mode (NMM) Enabled. **Note:** It cannot be active with RS-485_AAD operation mode.
[7]	**LINTXEN**	**LIN TX Break Mode Enable Bit** 0 = LIN TX Break mode Disabled. 1 = LIN TX Break mode Enabled. **Note:** When TX break field transfer operation finished, this bit will be cleared automatically.
[6]	**LINRXEN**	**LIN RX Enable Bit** 0 = LIN RX mode Disabled. 1 = LIN RX mode Enabled.
[5:4]	**Reserved**	Reserved.
[3:0]	**BRKFL**	**UART LIN Break Field Length** This field indicates a 4-bit LIN TX break field count. **Note1:** This break field length is BRKFL + 1. **Note2:** According to LIN spec, the reset value is 0xC (break field length = 13).

UART Function Select Register (UART_FUNCSEL)

Register	Offset	R/W	Description	Reset Value
UART_FUNCSEL **x=0,1,2,3,4,5,6,7,8,9**	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. 001 = LIN function. 010 = IrDA function. 011 = RS-485 function. Others = Reserved.

UART LIN Control Register (UART_LINCTL)

Register	Offset	R/W	Description	Reset Value
UART_LINCTL **x=1,2**	UARTx_BA+0x34	R/W	UART LIN Control Register	0x000C_0000

31	30	29	28	27	26	25	24
PID							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
HSEL		BSL		BRKFL			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			BITERREN	LINRXOFF	BRKDETEN	IDPEN	SENDH
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			MUTE	SLVDUEN	SLVAREN	SLVHDEN	SLVEN

Bits	Description	
[31:24]	**PID**	**LIN PID Bits** This field contains the LIN frame ID value in LIN function mode, and the frame ID parity can be generated by software or hardware depends on IDPEN (UART_LINCTL[9]) = 1. If the parity generated by hardware, user fill ID0~ID5 (PID [29:24]), hardware will calculate P0 (PID[30]) and P1 (PID[31]), otherwise user must filled frame ID and parity in this field. **Note1:** User can fill any 8-bit value to this field and the bit 24 indicates ID0 (LSB first). **Note2:** This field can be used for LIN master mode or slave mode.
[23:22]	**HSEL**	**LIN Header Select** 00 = The LIN header includes “break field”. 01 = The LIN header includes “break field” and “sync field”. 10 = The LIN header includes “break field”, “sync field” and “frame ID field”. 11 = Reserved. **Note:** This bit is used to master mode for LIN to send header field (SENDH (UART_LINCTL [8]) = 1) or used to slave to indicates exit from mute mode condition (MUTE (UART_LINCTL[4] = 1).
[21:20]	**BSL**	**LIN Break/Sync Delimiter Length** 00 = The LIN break/sync delimiter length is 1-bit time. 01 = The LIN break/sync delimiter length is 2-bit time. 10 = The LIN break/sync delimiter length is 3-bit time. 11 = The LIN break/sync delimiter length is 4-bit time. **Note:** This bit used for LIN master to sending header field.
[19:16]	**BRKFL**	**LIN Break Field Length** This field indicates a 4-bit LIN TX break field count. **Note1:** These registers are shadow registers of BRKFL (UART_ALTCTL[3:0]), User can read/write it by setting BRKFL (UART_ALTCTL[3:0]) or BRKFL (UART_LINCTL[19:16]). **Note2:** This break field length is BRKFL + 1. **Note3:** According to LIN spec, the reset value is 12 (break field length = 13).

[16:15]	**Reserved**	Reserved.
[12]	**BITERREN**	**Bit Error Detect Enable Bit** 0 = Bit error detection function Disabled. 1 = Bit error detection function Enabled. **Note:** In LIN function mode, when occur bit error, the BITEF (UART_LINSTS[9]) flag will be asserted. If the LINIEN (UART_INTEN[8]) = 1, an interrupt will be generated.
[11]	**LINRXOFF**	**LIN Receiver Disable Bit** If the receiver is enabled (LINRXOFF (UART_LINCTL[11]) = 0), all received byte data will be accepted and stored in the RX FIFO, and if the receiver is disabled (LINRXOFF (UART_LINCTL[11] = 1), all received byte data will be ignore. 0 = LIN receiver Enabled. 1 = LIN receiver Disabled. **Note:** This bit is only valid when operating in LIN function mode (FUNCSEL (UART_FUNCSEL[1:0]) = 01).
[10]	**BRKDETEN**	**LIN Break Detection Enable Bit** When detect consecutive dominant greater than 11 bits, and are followed by a delimiter character, the BRKDETF (UART_LINSTS[8]) flag is set at the end of break field. If the LINIEN (UART_INTEN [8])=1, an interrupt will be generated. 0 = LIN break detection Disabled. 1 = LIN break detection Enabled.
[9]	**IDPEN**	**LIN ID Parity Enable Bit** 0 = LIN frame ID parity Disabled. 1 = LIN frame ID parity Enabled. **Note1:** This bit can be used for LIN master to sending header field (SENDH (UART_LINCTL[8])) = 1 and HSEL (UART_LINCTL[23:22]) = 10 or be used for enable LIN slave received frame ID parity checked. **Note2:** This bit is only used when the operation header transmitter is in HSEL (UART_LINCTL[23:22]) = 10.
[8]	**SENDH**	**LIN TX Send Header Enable Bit** The LIN TX header can be “break field” or “break and sync field” or “break, sync and frame ID field”, it is depend on setting HSEL (UART_LINCTL[23:22]). 0 = Send LIN TX header Disabled. 1 = Send LIN TX header Enabled. **Note1:** This bit is shadow bit of LINTXEN (UART_ALTCTL [7]); user can read/write it by setting LINTXEN (UART_ALTCTL [7]) or SENDH (UART_LINCTL [8]). **Note2:** When transmitter header field (it may be “break” or “break + sync” or “break + sync + frame ID” selected by HSEL (UART_LINCTL[23:22]) field) transfer operation finished, this bit will be cleared automatically.
[7:5]	**Reserved**	Reserved.
[4]	**MUTE**	**LIN Mute Mode Enable Bit** 0 = LIN mute mode Disabled. 1 = LIN mute mode Enabled. **Note:** The exit from mute mode condition and each control and interactions of this field are explained in 6.14.5.10 (LIN slave mode).
[3]	**SLVDUEN**	**LIN Slave Divider Update Method Enable Bit** 0 = UART_BAUD updated is written by software (if no automatic resynchronization update occurs at the same time). 1 = UART_BAUD is updated at the next received character. User must set the bit before checksum reception.

		Note1: This bit only is valid in LIN slave mode (SLVEN (UART_LINCTL[0]) = 1). **Note2:** This bit used for LIN Slave Automatic Resynchronization mode. (for Non-Automatic Resynchronization mode, this bit should be kept cleared) **Note3:** The control and interactions of this field are explained in 6.14.5.10 (Slave mode with automatic resynchronization).
[2]	**SLVAREN**	**LIN Slave Automatic Resynchronization Mode Enable Bit** 0 = LIN automatic resynchronization Disabled. 1 = LIN automatic resynchronization Enabled. **Note1:** This bit only is valid in LIN slave mode (SLVEN (UART_LINCTL[0]) = 1). **Note2:** When operation in Automatic Resynchronization mode, the baud rate setting must be mode2 (BAUDM1 (UART_BAUD [29]) and BAUDM0 (UART_BAUD [28]) must be 1). **Note3:** The control and interactions of this field are explained in 6.14.5.10 (Slave mode with automatic resynchronization).
[1]	**SLVHDEN**	**LIN Slave Header Detection Enable Bit** 0 = LIN slave header detection Disabled. 1 = LIN slave header detection Enabled. **Note1:** This bit is only valid in LIN slave mode (SLVEN (UART_LINCTL[0]) = 1). **Note2:** In LIN function mode, when detect header field (break + sync + frame ID), SLVHDETF (UART_LINSTS [0]) flag will be asserted. If the LINIEN (UART_INTEN[8]) = 1, an interrupt will be generated.
[0]	**SLVEN**	**LIN Slave Mode Enable Bit** 0 = LIN slave mode Disabled. 1 = LIN slave mode Enabled.

UART LIN Status Register (UART_LINSTS)

Register	Offset	R/W	Description	Reset Value
UART_LINSTS **x=1,2**	UARTx_BA+0x38	R/W	UART LIN Status Register	0x0000_0000

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

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**BITEF**	**Bit Error Detect Status Flag** At TX transfer state, hardware will monitor the bus state, if the input pin (UART_RXD) state not equals to the output pin (UART_TXD) state, BITEF (UART_LINSTS[9]) will be set. When occur bit error, if the LINIEN (UART_INTEN[8]) = 1, an interrupt will be generated. 0 = Bit error not detected. 1 = Bit error detected. **Note1:** This bit can be cleared by writing 1 to it. **Note2:** This bit is only valid when enable bit error detection function (BITERREN (UART_LINCTL [12]) = 1).
[8]	**BRKDETF**	**LIN Break Detection Flag** This bit is set by hardware when a break is detected and be cleared by writing 1 to it through software. 0 = LIN break not detected. 1 = LIN break detected. **Note1:** This bit can be cleared by writing 1 to it. **Note2:** This bit is only valid when LIN break detection function is enabled (BRKDETEN (UART_LINCTL[10]) =1).
[7:4]	**Reserved**	Reserved.
[3]	**SLVSYNCF**	**LIN Slave Sync Field** This bit indicates that the LIN sync field is being analyzed in Automatic Resynchronization mode. When the receiver header have some error been detect, user must reset the internal circuit to re-search new frame header by writing 1 to this bit. 0 = The current character is not at LIN sync state. 1 = The current character is at LIN sync state. **Note1:** This bit is only valid in LIN Slave mode (SLVEN(UART_LINCTL[0]) = 1).

		Note2: This bit can be cleared by writing 1 to it. **Note3:** When writing 1 to it, hardware will reload the initial baud rate and re-search a new frame header.
[2]	**SLVIDPEF**	**LIN Slave ID Parity Error Flag** This bit is set by hardware when receipted frame ID parity is not correct. 0 = No active. 1 = Receipted frame ID parity is not correct. **Note1:** This bit can be cleared by writing 1 to it. **Note2:** This bit is only valid in LIN slave mode (SLVEN (UART_LINCTL [0])= 1) and enable LIN frame ID parity check function IDPEN (UART_LINCTL [9]).
[1]	**SLVHEF**	**LIN Slave Header Error Flag** This bit is set by hardware when a LIN header error is detected in LIN slave mode and be cleared by writing 1 to it. The header errors include “break delimiter is too short (less than 0.5 bit time)”, “frame error in sync field or Identifier field”, “sync field data is not 0x55 in Non-Automatic Resynchronization mode”, “sync field deviation error with Automatic Resynchronization mode”, “sync field measure time-out with Automatic Resynchronization mode” and “LIN header reception time-out”. 0 = LIN header error not detected. 1 = LIN header error detected. **Note1:** This bit can be cleared by writing 1 to it. **Note2:** This bit is only valid when UART is operated in LIN slave mode (SLVEN (UART_LINCTL [0]) = 1) and enables LIN slave header detection function (SLVHDEN (UART_LINCTL [1])).
[0]	**SLVHDETF**	**LIN Slave Header Detection Flag** This bit is set by hardware when a LIN header is detected in LIN slave mode and be cleared by writing 1 to it. 0 = LIN header not detected. 1 = LIN header detected (break + sync + frame ID). **Note1:** This bit can be cleared by writing 1 to it. **Note2:** This bit is only valid in LIN slave mode (SLVEN (UART_LINCTL [0]) = 1) and enable LIN slave header detection function (SLVHDEN (UART_LINCTL [1])). **Note3:** When enable ID parity check IDPEN (UART_LINCTL [9]), if hardware detect complete header (“break + sync + frame ID”), the SLVHDETF will be set whether the frame ID correct or not.

UART Baud Rate Compensation Register (UART_BRCOMP)

Register	Offset	R/W	Description	Reset Value
UART_BRCOMP **x=0,1,2,3,4,5,6,7,8,9**	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, 7,8,9**	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. **Note1:** When the system is in Power-down mode, Received Data FIFO reached threshold time-out will wake up system from Power-down mode. **Note2:** It is suggested the function is enabled when the WKRFRTEN (UART_WKCTL[2]) is set to 1.
[3]	**WKRS485EN**	**RS-485 Address Match (AAD Mode) Wake-up Enable Bit** 0 = RS-485 Address Match (AAD mode) wake-up system function Disabled. 1 = RS-485 Address Match (AAD mode) wake-up system function Enabled. **Note1:** When the system is in.Power-down mode, RS-485 Address Match will wake-up system from Power-down mode. **Note2:** This bit is used for RS-485 Auto Address Detection (AAD) mode in RS-485 function mode and ADDRDEN (UART_ALTCTL[15]) is set to 1.
[2]	**WKRFRTEN**	**Received Data FIFO Reached Threshold Wake-up Enable Bit** 0 = Received Data FIFO reached threshold wake-up system function Disabled. 1 = Received Data FIFO reached threshold wake-up system function Enabled. **Note:** When the system is in Power-down mode, Received Data FIFO reached threshold will wake-up system from Power-down mode.
[1]	**WKDATEN**	**Incoming Data Wake-up Enable Bit** 0 = Incoming data wake-up system function Disabled. 1 = Incoming data wake-up system function Enabled. **Note:**When the system is in Power-down mode, incoming data will wake-up system from Power-down mode.

[0]	**WKCTSEN**	**nCTS Wake-up Enable Bit** 0 = nCTS Wake-up system function Disabled. 1 = nCTS Wake-up system function Enabled. **Note:**When the system is in Power-down mode, an external.nCTS change will wake up system from Power-down mode.

UART Wake-up Status Register (UART_WKSTS)

Register	Offset	R/W	Description	Reset Value
UART_WKSTS **x=0,1,2,3,4,5,6, 7,8,9**	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. **Note1:** If WKTOUTEN (UART_WKCTL[4]) is enabled, the Received Data FIFO reached threshold time-out wake-up cause this bit is set to '1'. **Note2:** This bit can be cleared by writing '1' to it.
[3]	**RS485WKF**	**RS-485 Address Match (AAD Mode) Wake-up Flag** This bit is set if chip wake-up from power-down state by RS-485 Address Match (AAD mode). 0 = Chip stays in power-down state. 1 = Chip wake-up from power-down state by RS-485 Address Match (AAD mode) wake-up. **Note1:** If WKRS485EN (UART_WKCTL[3]) is enabled, the RS-485 Address Match (AAD mode) wake-up cause this bit is set to '1'. **Note2:** This bit can be cleared by writing '1' to it.
[2]	**RFRTWKF**	**Received Data FIFO Reached Threshold Wake-up Flag** This bit is set if chip is woken up from power-down state by Received Data FIFO reached threshold wake-up. 0 = Chip stays in power-down state. 1 = Chip wake-up from power-down state by Received Data FIFO Reached Threshold wake-up. **Note1:** If WKRFRTEN (UART_WKCTL[2]) is enabled, the Received Data FIFO Reached

		Threshold wake-up cause this bit is set to ‘1’. **Note2:** This bit can be cleared by writing ‘1’ to it.
[1]	**DATWKF**	**Incoming Data Wake-up Flag** This bit is set if chip wake-up from power-down state by data wake-up. 0 = Chip stays in power-down state. 1 = Chip wake-up from power-down state by Incoming Data wake-up. **Note1:** If WKDATEN (UART_WKCTL[1]) is enabled, the Incoming Data wake-up cause this bit is set to ‘1’. **Note2:** This bit can be cleared by writing ‘1’ to it.
[0]	**CTSWKF**	**nCTS Wake-up Flag** This bit is set if chip wake-up from power-down state by nCTS wake-up. 0 = Chip stays in power-down state. 1 = Chip wake-up from power-down state by nCTS wake-up. **Note1:** If WKCTSEN (UART_WKCTL[0]) is enabled, the nCTS wake-up cause this bit is set to ‘1’. **Note2:** This bit can be cleared by writing ‘1’ to it.

UART Incoming Data Wake-up Compensation Register (UART_DWKCOMP)

Register	Offset	R/W	Description	Reset Value
UART_DWKCOMP **x=0,1,2,3,4,5,6,7,8,9**	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.

6.15 Smart Card Host Interface (SC)

6.15.1 Overview

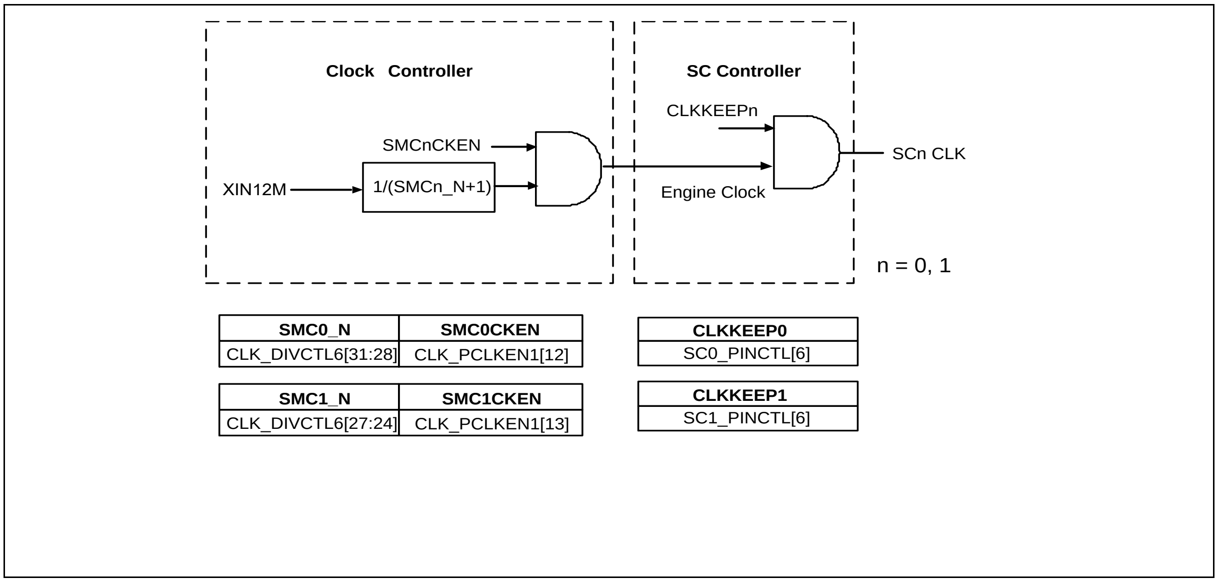
The Smart Card Interface controller (SC controller) is based on ISO/IEC 7816-3 standard and fully compliant with PC/SC Specifications. It also provides status of card insertion/removal.

6.15.2 Features

- ISO 7816-3 T = 0, T = 1 compliant
- EMV2000 compliant
- Two ISO 7816-3 ports
- Separates receive/transmit 4 byte entry FIFO for data payloads
- Programmable transmission clock frequency
- Programmable receiver buffer trigger level
- Programmable guard time selection (11 ETU ~ 267 ETU)
- One 24-bit timer and two 8-bit timers for Answer to Reset (ATR) and waiting times processing
- Supports auto direct / inverse convention function
- Supports transmitter and receiver error retry and error number limiting function
- Supports hardware activation sequence process, and the time between PWR on and CLK start is configurable
- Supports hardware warm reset sequence process
- Supports hardware deactivation sequence process
- Supports hardware auto deactivation sequence when detected the card removal
- Supports UART mode
  - Full duplex, asynchronous communications
  - Separates receiving / transmitting 4 bytes entry FIFO for data payloads
  - Supports programmable baud rate generator
  - Supports programmable receiver buffer trigger level
  - Programmable transmitting data delay time between the last stop bit leaving the TX-FIFO and the de-assertion by setting EGT (SCn_EGT[7:0])
  - Programmable even, odd or no parity bit generation and detection
  - Programmable stop bit, 1- or 2- stop bit generation

6.15.3 Block Diagram

The SC clock control and block diagram are shown in SC Clock Control Diagram (4-bit Pre-scale Counter in Clock Controller) and SC Controller Block Diagram. The SC controller is completely asynchronous design with two clock domains, PCLK and engine clock. Note that the PCLK should be higher than or equal to the frequency of engine clock.



SMC0_N	SMC0CKEN
CLK_DIVCTL6[31:28]	CLK_PCLKEN1[12]

SMC1_N	SMC1CKEN
CLK_DIVCTL6[27:24]	CLK_PCLKEN1[13]

CLKKEEP0
SC0_PINCTL[6]

CLKKEEP1
SC1_PINCTL[6]

Figure 6.15-1 SC Clock Control Diagram (4-bit Pre-scale Counter in Clock Controller)

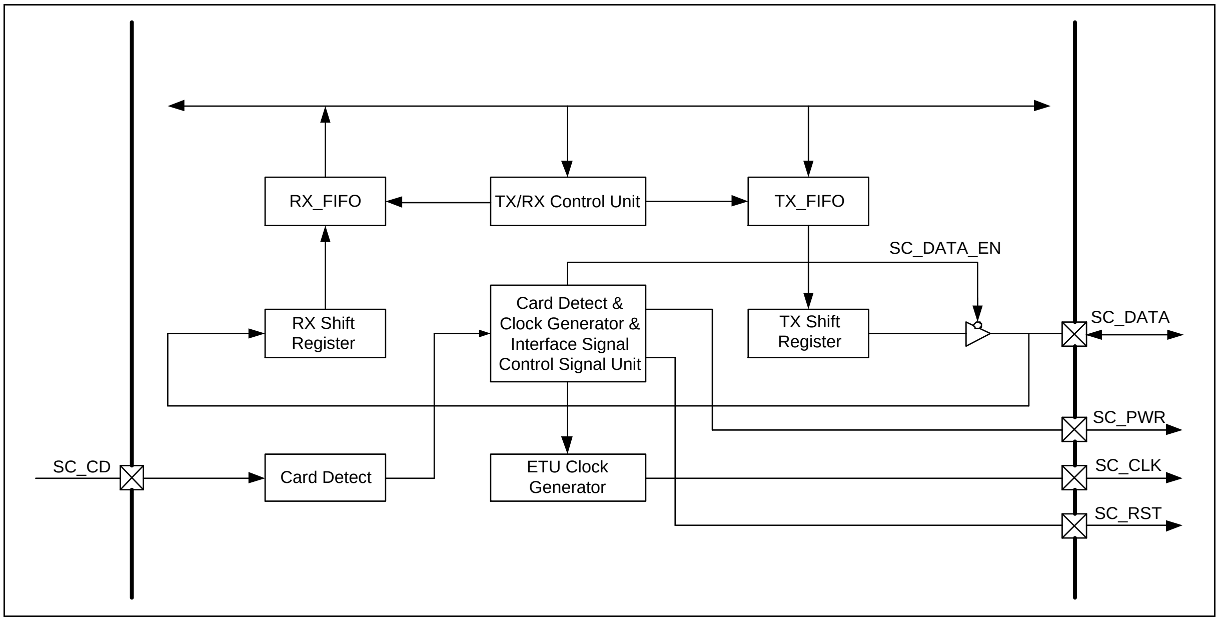


Figure 6.15-2 SC Controller Block Diagram

6.15.4 Basic Configuration

SC Host Controller Pin description is shown in:

Pin	Type	Description
SCn_DATA	Bi-direction	SC Host Controller DATA
SCn_CD	Input	SC Host Controller Card Detect
SCn_PWR	Output	SC Host Controller Power ON/OFF
SCn_CLK	Output	SC Host Controller Clock
SCn_RST	Output	SC Host Controller Reset

Table 6.15-1 SC Host Controller Pin Description

UART Mode Pin description is shown in UART Pin Description:

Pin	Type	Description
SCn_DATA	Input	UART Receive Data
SCn_CLK	Output	UART Transmit Data

Table 6.15-2 UART Pin Description

6.15.4.1 SC0 Basic Configuration

- Clock Source Configuration
  - Select the clock divider number of SC0 peripheral clock on SMC0_N(CLK_DIVCTL6[27:24]).
  - Enable SC0 peripheral clock in SC0CKEN (CLK_PCLKEN1[12]).
- Reset Configuration
  - Reset SC0 controller in SC0RST (SYS_APBIPRST2[12]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
SC0	SC0_CD	PA.2	MFP3
		PC.15	MFP4
	SC0_CLK	PA.5	MFP3
		PC.12	MFP4
	SC0_DAT	PA.4	MFP3
		PC.13	MFP4
	SC0_PWR	PA.3	MFP3
		PC.14	MFP4
	SC0_RST	PA.6	MFP3
		PC.11	MFP4

Table 6.15-3 SC0 Pin Configuration

6.15.4.2 SC1 Basic Configuration

- Clock Source Configuration
  - Select the clock divider number of SC1 peripheral clock on SMC1_N(CLK_DIVCTL6[31:28]).
  - Enable SC1 peripheral clock in SC1CKEN (CLK_PCLKEN1[13]).
- Reset Configuration
  - Reset SC1 controller in SC1RST (SYS_APBIPRST2[13]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
SC1	SC1_CD	PC.10	MFP4
		PF.4	MFP4
	SC1_CLK	PC.7	MFP4
		PF.1	MFP4
	SC1_DAT	PC.8	MFP4
		PF.2	MFP4
	SC1_PWR	PC.9	MFP4
		PF.3	MFP4
	SC1_RST	PC.6	MFP4
		PF.0	MFP4

Table 6.15-4 SC1 Pin Configuration

6.15.5 Functional Description

Basically, the smart card interface acts as a half-duplex asynchronous communication port and its data format is composed of ten consecutive bits which is shown in SC Data Character.

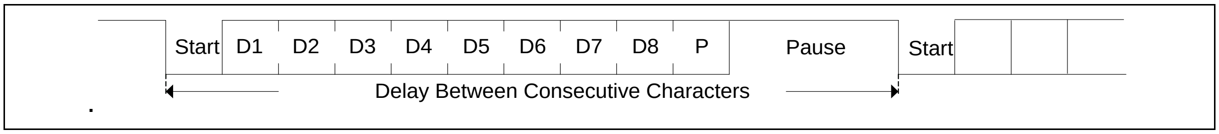


Figure 6.15-3 SC Data Character

The Smart Card Interface controller supports hardware activation, warm reset and deactivation sequence. The activation, warm reset and deactivation sequence are shown as follows.

6.15.5.1 Smart card pin configuration

The Smart Card Interface pin status can be observed by polling following registers.

1. SCn_RST is a output pin. Its status can be observed by RSTSTS (SCn_PINCTL[18]). Programming RSTEN (SCn_PINCTL[1]) ‘0’ or ‘1’ can drive this output pin low or high.
2. SCn_PWR is a output pin. Its status can be observed by PWRSTS (SCn_PINCTL[17]). Programming PWREN (SCn_PINCTL[0]) to ‘0’ or ‘1’ can drive this output pin low or high.

PWRINV (SCn_PINCTL[11]) can inverse the SCn_PWR output. User must select PWRINV (SCn_PINCTL[11]) before smart card is enabled by SCEN (SCn_CTL[0]).

3. SCn_DATA is a bidirectional pin, DATASTS(SCn_PINCTL[16]) shows the pin status when SC is receiving data. programming SCDATA (SCn_PINCTL[9]) to ‘0’ or ‘1’ can drive this pin output low or high.
4. SCn_CLK is a output pin. It outputs Smart card clock SCn CLK. Programming CLKKEEP (SCn_PINCTL[6]) ‘0’ or ‘1’ to disable or enable this pin. Programming CSTOPLV (SCn_PINCTL[5]) can determine the SCn_CLK is stopped at high or low when this pin is disable.
5. SCn_CD(Card Detect Pin) state represent the status of the card is inserted or not. SCn_CD pin status can be observed by CDPINSTS(SC_CDPINSTS[13]). SCn_CD related function can be set by CDLV(SC_CTL[26]), CDDBSEL(SC_CTL[25:24]), CDIF(SC_INTSTS[7]), CDIEN(SC_INTEN[7]).
6. CDLV(SC_CTL[26]) determines what kind of pin level change represents the card insertion. CDDBSEL(SC_CTL[25:24]) determines the de-bounce cycles. When the card status CINSERT (SCn_STATUS[12]) or CREMOVE (SCn_STATUS[11]) is detected, CDIF will set to 1. If CDIEN is enable, SC will deliver a interrupt to CPU when CDIF is set to 1. Card Detect Pin is recommend setting before enable SC.

6.15.5.2 Activation, Warm Reset and Deactivation Sequence

Activation

The activation sequence is shown in Figure 6.15-4:

1. Set SCn_RST to low by programming RSTEN (SCn_PINCTL[1]) to ‘0’, and wait SYNC (SCn_PINCTL[31]) is cleared to 0.
2. Set SCn_PWR at high level by programming PWREN (SCn_PINCTL[0]) to ‘1’ and SCn_DATA at high level (reception mode) by programming SCDATA (SCn_PINCTL[9]) to ‘1’, and wait SYNC(SCn_PINCTL[31]) is cleared to 0.
3. Enable SCn_CLK clock by programming CLKKEEP (SCn_PINCTL[6]) to ‘1’, and wait SYNC(SCn_PINCTL[31]) is cleared to 0.
4. De-assert SCn_RST to high by programming RSTEN (SCn_PINCTL[1]) to ‘1’, and wait SYNC(SCn_PINCTL[31]) is cleared to 0.

The activation sequence can be controlled in two ways. The procedure is shown as follows:

- Software Timing Control:

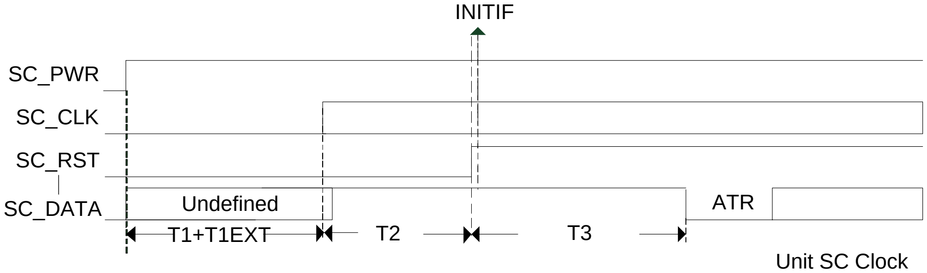
  Set SCn_PINCTL and SCn_TMRCTLx (x = 0, 1, 2) to process the activation sequence. SCn_PWR, SCn_CLK, SCn_RST and SCn_DATA pin state can be programmed by SCn_PINCTL. The programming method is shown in activation sequence. The activation sequence timing can be controlled by setting SCn_TMRCTLx (x = 0, 1, 2). This programming procedure provides user with a flexible timing setting for activation sequence.

- Hardware Timing Control:

  Set ACTEN (SCn_ALTCTL[3]) to ‘1’ and the interface will perform the activation sequence by hardware. The SCn_PWR to SCn_CLK start (T1) and SCn_CLK_start to SCn_RST assert (T2) can be selected by programming INITSEL (SCn_ALTCTL[9:8]). The SCn_PWR to SCn_CLK length can be configure by setting T1EXT(SCn_ACTCTL[4:0]). This programming procedure provides user with a simple setting for activation sequence. During the hardware activation, RX receive is disabled and can not receive data.

The following is activation control sequence in hardware activation mode:

1. Set activation timing by setting INITSEL (SCn_ALTCTL[9:8]).
2. Timer0 can be selected by setting TMRSEL (SCn_CTL[14:13]) is 11.
3. Set operation mode OPMODE (SCn_TMRCTL0[27:24]) to 0011 and give an Answer to Reset (ATR) value by setting CNT (SCn_TMRCTL0[23:0]) register.
4. When hardware de-asserts SCn_RST to high, hardware will generator an interrupt INITIF (SCn_INTSTS[8]) to CPU at the same time if INITIEN (SCn_INTEN[8]) is 1.
5. If the Timer0 decreases the counter to “0” (started from SCn_RST de-assert) and the card does not response ATR before that time, hardware will generate an interrupt flag TMR0IF (SCn_INTSTS[3]).



Time	Comment
T1	SC_PWR to SC_CLK Start
T1EXT	Configurable cycles for T1
T2	SC_CLK Start to SC_RST Assert
T3	SC_CLK Start to ART Appear

Unit SC Clock

INITSEL (SC_ALTCTL[9:8])	T1	T2
00	83.5	491
01	133	537
10	165	569
11	165	42060

Note : The values are measured by chip I / O pin and the real value will depend on system design

Figure 6.15-4 SC Activation Sequence

Warm Reset

The warm reset sequence is shown in Figure 6.15-5 :

1. Set SCn_RST to low by programming RSTEN (SCn_PINCTL[1]) to ‘0’, and wait SYNC(SCn_PINCTL[31]) is cleared to 0.
2. Set SCn_DATA to high by programming SCDATA (SCn_PINCTL[9]) to ‘1’, and wait SYNC(SCn_PINCTL[31]) is cleared to 0.
3. Set SCn_RST to high by programming RSTEN (SCn_PINCTL[1]) to ‘1’, and wait SYNC(SCn_PINCTL[31]) is cleared to 0.

The warm reset sequence can be controlled in two ways. The procedure is shown as follows.

- Software Timing Control:

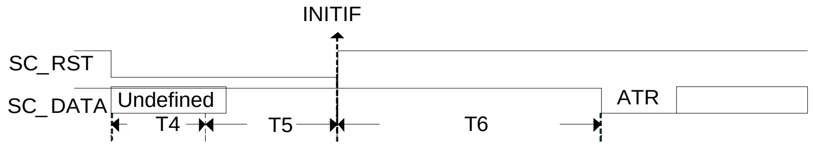
  Set SCn_PINCTL and SCn_TMRCTLx (x = 0, 1, 2) to process the warm reset sequence. The SCn_RST and SCn_DATA pin state can be programmed by SCn_PINCTL. The warm reset sequence timing can be controlled by setting SCn_TMRCTLx (x = 0, 1, 2). This programming procedure provides user with a flexible timing setting for warm reset sequence.

- Hardware Timing Control:

  Set WARSTEN (SCn_ALTCTL[4]) to ‘1’ and the interface will perform the warm reset sequence by hardware. The SCn_RST to SCn_DATA reception mode (T4) and SCn_DATA reception mode to SCn_RST assert (T5) can be selected by programming INITSEL (SCn_ALTCTL[9:8]). This programming procedure provides user with a simple setting for warm reset sequence. During the hardware warm reset, RX receive is disabled and can not receive data.

The following is the warm reset control sequence by hardware:

1. Set warm reset timing by setting INITSEL (SCn_ALTCTL[9:8]).
2. Select Timer0 by setting TMRSEL (SCn_CTL[14:13]) to 11.
3. Set operation mode OPMODE (SCn_TMRCTL0[27:24]) to 0011 and give an Answer to Reset(ATR) value by setting CNT (SCn_TMRCTL0[23:0]) register.
4. Set CNTEN0 (SCn_ALTCTL[5]) and WARSTEN (SCn_ALTCTL[4]) to start counting.
5. When hardware de-asserts SCn_RST to high, hardware will generate an interrupt INITIF (SCn_INTSTS[8]) to CPU at the same time if INITIEN (SCn_INTEN[8]) is 1.
6. If the Timer0 decreases the counter to ‘0’ (start from SCn_RST) and the card does not response ATR before that time, hardware will generate an interrupt flag TMR0IF (SCn_INTSTS[3]).



INITSEL (SC_ALTCTL[9:8])	T4	T5
00	81	483
01	129	531
10	161	563
11	161	42106

Unit: SC Clock

Time	Comment
T4	SC_RST to SC_DATA Reception Mode
T5	SC_DATA Reception Mode to SC_RST Assert
T6	SC_CLK Start to ATR Appear

Note: This value is measured by chip IO pin and the real value will depend on system design

Figure 6.15-5 SC Warm Reset Sequence

Deactivation

The deactivation sequence is shown in Figure 6.15-6.

1. Set SCn_RST to low by programming RSTEN (SCn_PINCTL[1]) to ‘0’, wait SYNC(SCn_PINCTL[31]) is cleared to 0.
1. Stop SCn_CLK by programming CLKKEEP (SCn_PINCTL[6]) to ‘0’, wait SYNC(SCn_PINCTL[31]) is cleared to 0.
2. Set SCn_DATA to low by programming SCDATA (SCn_PINCTL[9]) to ‘0’, wait SYNC(SCn_PINCTL[31]) is cleared to 0.
3. Deactivate SCn_PWR by programming PWREN (SCn_PINCTL[0]) to ‘0’, wait SYNC(SCn_PINCTL[31]) is cleared to 0.

The deactivation sequence can be controlled in two ways. The procedure is shown as follows.

- Software Timing Control:

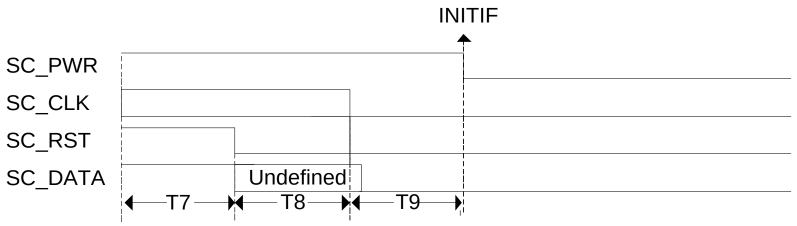
  Set SCn_PINCTL and SCn_TMRCTL0 to process the deactivation sequence. SCn_PWR, SCn_CLK, SC.n_RST and SCn_DATA pin state can be programmed by SCn_PINCTL. The deactivation sequence timing can be controlled by setting SCn_TMRCTL0. This programming procedure provides user with a flexible timing setting for deactivation sequence.

- Hardware Timing Control:

  DACTEN (SCn_ALTCTL[2]) to '1' and the interface will perform the deactivation sequence by hardware. The Deactivation Trigger to SCn_RST low (T7), SMC_RST low to SCn_CLK (T8) and stop SCn_CLK to stop SCn_PWR (T9) time can be selected by programming INITSEL (SCn_ALTCTL[9:8]). This programming procedure provides user with a simple setting for deactivation sequence.

When hardware de-asserts SCn_PWR to low, the SC controller will generate an interrupt INITIF (SCn_INTSTS[8]) to CPU at the same time if INITIEN (SCn_INTEN[8]) is 1.

The SC controller also supports auto deactivation sequence when the card removal detection is enabled by setting ADACEN (SCn_ALTCTL[11]).



Time	Comment
T7	Deactivation Trigger to SC_RST Low
T8	SMC_RST Low to Stop SC_CLK
T9	Stop SC_CLK to Stop SC_PWR

INITSEL (SC_ALTCTL[9:8])	T7	T8	T9
00	97	83	87
01	145	131	135
10	177	163	167
11	177	163	167

Unit: SC Clock

Figure 6.15-6 SC Deactivation Sequence

6.15.5.3 Basic Operation Flow

Basic operation flow of smartcard can be referenced from ISO 7816-3 & EMV.

The Program Sequence Flow is shown as follows:

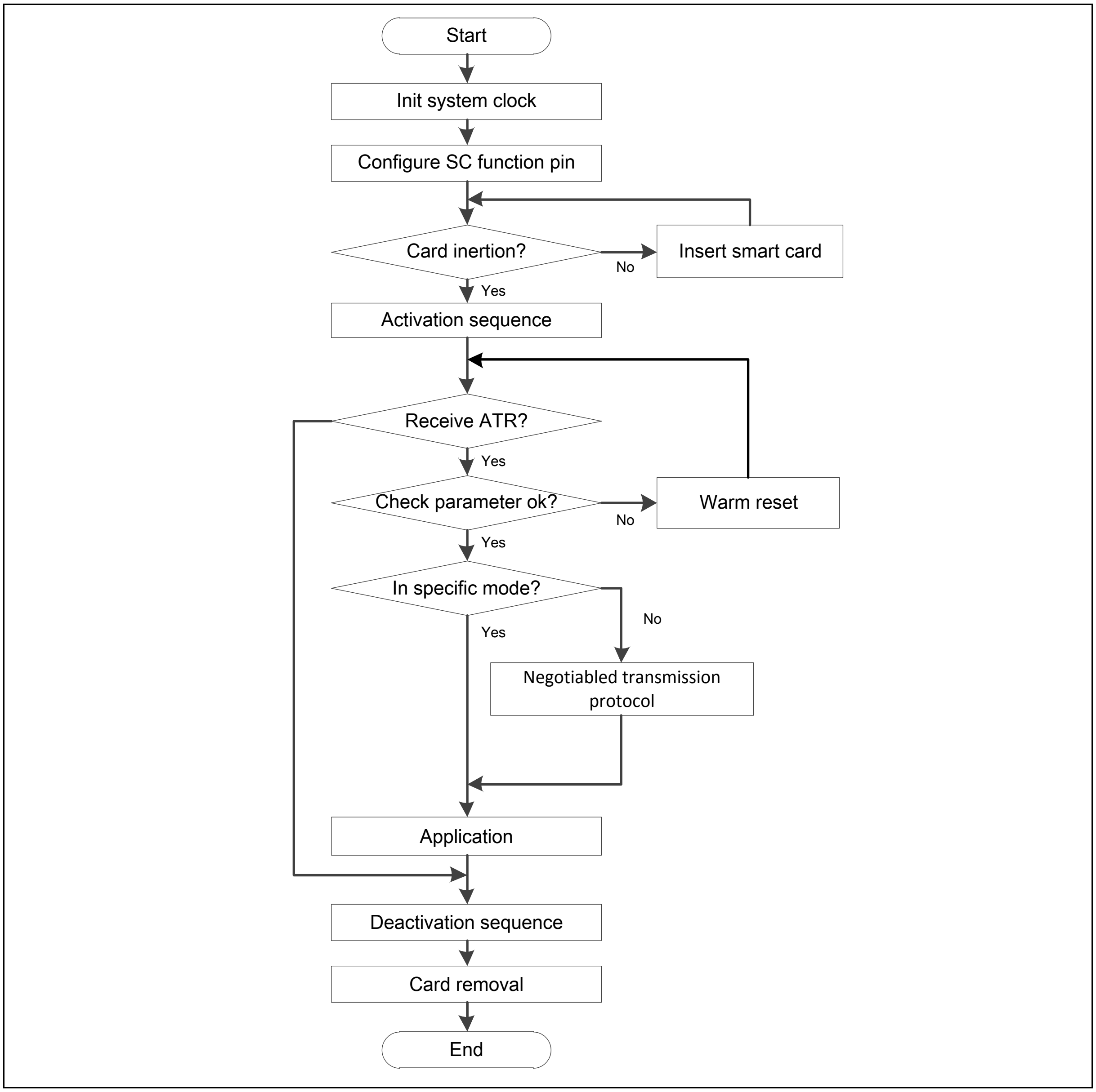


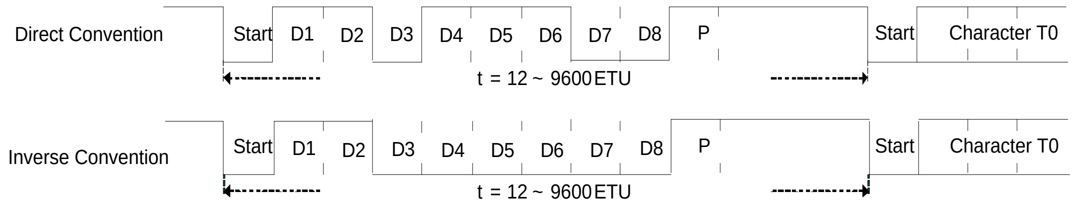
Figure 6.15-7 Basic Operation Flow

6.15.5.4 Initial Character TS

According to ISO 7816-3, the initial character TS has two possible patterns shown in Figure 6.15-8. If the TS pattern is 1100_0000, it is inverse convention. When decoded by inverse convention, the conveyed byte is equal to 0x3F. If the TS pattern is 1101_1100, it is direct convention. When decoded by direct convention, the conveyed byte is equal to 0x3B. User can set AUTOCEN (SCn_CTL[3]) and then the operating convention will be decided by hardware. User can also set the CONSEL (SCn_CTL[5:4]) register (set to '00' or '11') to change the operating convention after SC received TS of answer to reset (ATR).

If auto convention function is enabled by setting AUTOCEN (SCn_CTL[3]) register, the setting step must be done before Answer to Reset (ATR) state and the received first data must be 0x3B or 0x3F. After hardware received first data and stored it at SCn_DAT, the hardware will decided the convention and change the CONSEL (SCn_CTL[5:4]) register automatically. If the received first data is neither

0x3B nor 0x3F, ACERRIF (SCn_INTSTS[10] Auto Convention Error Interrupt Status Flag) will be set and the hardware will generate an interrupt to CPU if ACERRIEN (SCn_INTEN[10]) is 1.



Direct Convention	0_1101_1100_1(0x3B)
Inverse Convention	0_1100_0000_1(0x3F)

Figure 6.15-8 Initial Character TS

6.15.5.5 Transfer Data Flow and Data Buffer Status

After setting initial sequence, SC can start transfering data which format is corresponding to ISO 7816-3. Set ETURDIV(SC_ETUCTL[11:0]) to 273 to make ETU(Element Timing Unit) meet ISO 7816-3. Writing data to SC_DAT, SC will send out an 8-bit data to smart card. Reading data from SC_DAT, SC will return an 8-bit received data from smart card.

Data buffer status show in SC_STATUS. TXPOINT(SC_STATUS[26:24]) and RXPOINT(SC_STATUS[18:16]) represent how many data in transmit buffer and received buffer. TXEMPTY(SC_STATUS[9]), TXFULL(SC_STATUS[10]), TXOV(SC_STATUS[8]), RXEMPTY(SC_STATUS[1]), RXFULL(SC_STATUS[2]), RXOV(SC_STATUS[0]), represent the transmitted/received buffer status is full, empty, or overflow. SC even can generate interrupt for the transmit buffer empty situation by setting TBEIEN(SCn_INTEN[1]) flag. After interrupt is generated, it can be clear by writing 1 to TBEIF(SCn_INTSTS[1]). SC can decide to transfer data or not by polling these status.

TX and RX can be disabled seperately by setting TXOFF(SC_CTL[2]) and RXOFF(SC_CTL[1]). TXACT(SC_STATUS[31]) and RXACT(SC_STATUS[23]) represent the TX transfer/RX transfer is active or not.

6.15.5.6 Receiver Buffer Time-out

The time-out down counter resets and starts counting whenever the RX buffer received a new data. Once the counter decrease to 1 and no new data is received or CPU does not read data by reading SCn_DAT, a receiver time-out flag RXTOIF (SCn_INTSTS[9]) will be set, and hardware will generate an interrupt to CPU when RXTOIEN (SCn_INTEN[9]) is enabled.

6.15.5.7 Error Signal and Character Repetition

According to ISO 7816-3 T=0 mode description, as shown in Figure 6.15-9, if the receiver receives a wrong parity bit, it will pull the SCn_DAT to low by 1.5 bit period to inform the transmitter parity error. Then the transmitter will retransmit the character. The SC interface controller supports hardware error detection function in receiver and supports hardware re-transmit function in transmitter.

User can enable re-transmit function by setting TXRTYEN (SCn_CTL[23]). User can also define the retry (re-transmit) number limitation in TXRTY (SCn_CTL[22:20]). If the re-transmit number is between 1 and TXRTY, TXRERR (SCn_STATUS[29]) flag will be set by hardware. The re-transmit number is up to TXRTY +1 and if the re-transmit number is equal to TXRTY +1, TXOVERR (SCn_STATUS[30]) flag will be set by hardware and if TERRIEN (SCn_INTEN[2]) is enabled, SC controller will generate a transfer error interrupt to CPU, and TERRIF (SC_INTSTS[2]) flag will also be set.

User can also enable re-received function by setting RXRTYEN (SCn_CTL[19]).The received retry number limitation is defined in RXRTY (SCn_CTL[18:16]). If the re-received number is between 1 and RXRTY, RXRERR (SCn_STATUS[21]) flag will be set by hardware. The receiver retry number is up to RXRTY +1, if the number of received errors by receiver is equal to RXRTY +1, receiver will receive this error data to buffer and RXOVERR (SCn_STATUS[22]) flag will be set by hardware and if TERRIEN (SCn_INTEN[2]) is enabled, SC controller will generate a transfer error interrupt to CPU, and TERRIF (SC_INTSTS[2]) flag will also be set.

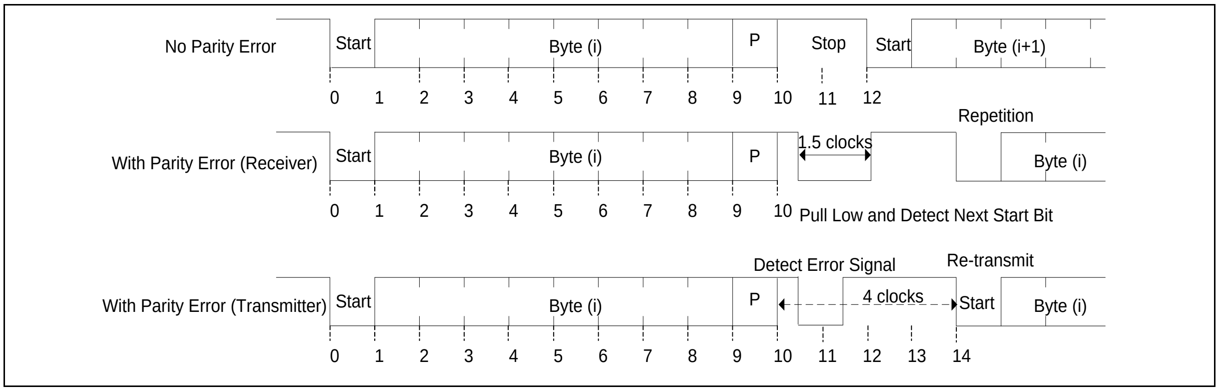


Figure 6.15-9 SC Error Signal

6.15.5.8 Internal Timer Operation Mode

The smart card interface includes a 24-bit time-out counter and two 8 bit time-out counters. These counters help the controller in processing different real-time interval. Each counter can be set to start counting once the trigger enable bit (CNTENx in SCn_ALTCTL[7:5], x = 0, 1, 2) has been written or a START bit has been detected.

The following is the programming flow:

1. Enable counter by setting TMRSEL (SCn_CTL[14:13]) to 11.
2. Select operation mode OPMODE (SCn_TMRCTLx[27:24], x = 0, 1, 2).
3. Give a count value CNT for Timer0, Timer1 and Timer2 by setting CNT0(SCn_TMRCTL0[23:0]), CNT1(SCn_TMRCTL1[7:0]) and CNT2(SCn_TMRCTL2[7:0] register).
4. Set CNTEN0 (SCn_ALTCTL [5]), CNTEN1 (SCn_ALTCTL [6]) or CNTEN2 (SCn_ALTCTL [7]) to enable timer. ACTSTS0(SCn_ALTCTL[13]), ACTSTS1(SCn_ALTCTL[14]) and ACTSTS2(SCn_ALTCTL[15]) represent the status that timer is enable or not.
5. Wait until the counting condition is satisfied, the timer start counting.
6. When internal timer counter satisfied interrupt conditions in different modes and TMR0IEN(SCn_INTEN[3]), TMR1IEN(SCn_INTEN[4]), TMR2IEN(SCn_INTEN[5]) are enable, SC will generate a interrupt to CPU.

The SCn_TMRCTL0, SCn_TMRCTL1 and SCn_TMRCTL2 timer operation mode are listed in Table

6.15-5.

Note1: Only Timer0 (SCn_TMRCTL0 register) supports mode 0011.

Note2: START bit can only be detected when Tx or Rx is idle or finish the last transmission.

OPMODE (SCn_TMRCTLx[27:24]), X = 0, 1, 2	**6.15.5.8..6.1.1 Operation Mode Description**	
0000	The down counter is started when CNTENx (SCn_ALTCTL[7:5]) is enabled and ended when counter time-out. The time-out counter value will be CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) +1.	
	Start	Start counting when CNTENx (SCn_ALTCTL[7:5]) enabled.
	End	When the down counter equals 0, hardware will set TMR0IF, TMR1IF, TMR2IF (SCn_INTSTS[5:3]) and clear CNTENx (SCn_ALTCTL[7:5]) automatically.
0001	The down counter is started when the first START bit (reception or transmission) detected and ended when counter time-out. It takes 2 ETU to detect first START bit after writing data to Tx or receiving data from Rx. The time-out counter value will be CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) +1.	
	Start	Start counting when the first START bit (reception or transmission) detected after CNTENx (SCn_ALTCTL[7:5]) set to 1.
	End	When the down counter equals 0, hardware will set TMR0IF, TMR1IF, TMR2IF (SCn_INTSTS[5:3]) and clear CNTENx (SCn_ALTCTL[7:5]) automatically.
0010	The down counter is started when the first START bit (reception) detected and ended when counter time-out. It takes 2 ETU to detect first START bit after receiving data from Rx. The time-out value will be CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) +1.	
	Start	Start counting when the first START bit (reception) detected bit after CNTENx (SCn_ALTCTL[7:5]) set to 1.
	End	When the down counter equals 0, hardware will set TMR0IF, TMR1IF, TMR2IF (SCn_INTSTS[5:3]) and clear CNTENx (SCn_ALTCTL[7:5]) automatically.
0011	The down counter is only used for hardware activation, warm reset sequence to measure ATR timing. The timing starts when SCn_RST de-assertion and ends when ATR response received or time-out. If the counter decreases to 0 before ATR response received, hardware will set TMR0IF (SCn_INTSTS[3]) and generate an interrupt to CPU if TMR0IEN (SCn_INTEN[3]) enabled. The time-out value will be CNT (SCn_TMRCTL0[23:0]) +1.	
	Start	Start counting when SCn_RST de-assertion after CNTEN0 (SCn_ALTCTL[5]) set to 1. It is only used for hardware activation, warm reset mode.
	End	When the down counter equals 0 before ATR response received, hardware will set TMR0IF and clear CNTEN0 (SCn_ALTCTL[5]) automatically. When ATR received and down counter does not equal to 0, hardware will clear CNTEN0 (SCn_ALTCTL[5]) automatically.
0100	Start	Start down counter counting when CNTENx (SCn_ALTCTL[7:5]) enabled.
	Recount & reload	When ACTSTSx (SCn_ALTCTL[15:13]) is 1, user can change CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL0[7:0], SCn_TMRCTL0[7:0]) value at any time. It will reload the last value which is filled into the CNT(SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) before the counter count to 0. Only when the down counter equals 0, counter reload the CNT (SCn_TMRCTL0[23:0],

		SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) value and start to recount.
	Interrupt	If the counter decreases to 0, hardware will set TMR0IF, TMR1IF, TMR2IF (SCn_INTSTS[5:3]) and generate an interrupt to CPU if TMRxIEN (SCn_INTEN[5:3]) enabled. The time-out value will be CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) +1.
	End	The down counter stopped when use clears CNTENx (SCn_ALTCTL[7:5]) bit.
0101	Start	The down counter is started when the first START bit (reception or transmission) detected after CNTENx (SCn_ALTCTL[7:5]) set to 1. It takes 2 ETU to detect START bit after writing data to Tx or receiving data from Rx.
	Reload	When ACTSTSx (SCn_ALTCTL[15:13]) is 1, user can change CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL0[7:0], SCn_TMRCTL0[7:0]) value at any time. It will reload the last value which is filled into the CNT(SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) before the counter count to 0. Only when the down counter equals 0, counter will reload the CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) value.
	Recount	After down counter reloads the CNT value, timer counter starts to recount only when the next START bit is detected.
	Interrupt	If the counter decreases to 0, hardware will set TMR0IF, TMR1IF, TMR2IF (SCn_INTSTS[5:3]) and generate an interrupt to CPU if TMRxIEN (SCn_INTEN[5:3]) enabled. The time-out value will be CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) +1.
	End	The down counter stopped when user clears CNTENx (SCn_ALTCTL[7:5]) bit.
0110	Start	The down counter is started when the first START bit (reception) detected after CNTENx (SCn_ALTCTL[7:5]) set to 1. It takes 2 ETU to detect START bit after writing data to Tx or receiving data from Rx.
	Reload	When ACTSTSx (SCn_ALTCTL[15:13]) is 1, user can change CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL0[7:0], SCn_TMRCTL0[7:0]) value at any time. It will reload the last value which is filled into the CNT(SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) before the counter counts to 0. Only when the down counter equals 0, counter reload the CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) value.
	Recount	After the down counter reloads the CNT value, timer counter starts to recount only when the next START bit is detected.
	Interrupt	If the counter decreases to 0, hardware will set TMR0IF, TMR1IF, TMR2IF (SCn_INTSTS[5:3]) and generate an interrupt to CPU if TMRxIEN (SCn_INTEN[5:3]) enabled. The time-out value will be CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0])+1.
	End	The down counter stopped when user clears CNTENx (SCn_ALTCTL[7:5]) bit.
0111	Start	The down counter is started when the first START bit (reception or transmission) detected after CNTENx (SCn_ALTCTL[7:5]) set to 1. It takes 2 ETU to detect START bit after writing data to Tx or receiving data from Rx.
	Reload &recount	Only when the next START bit is detected, counter will reload the new value of CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) and recount.
	Interrupt	If the counter decreases to 0 before the next START bit detected, hardware will set TMR0IF, TMR1IF, TMR2IF (SCn_INTSTS[5:3]) and generate an interrupt to CPU if TMRxIEN (SCn_INTEN[5:3]) enabled. The time-out value will be CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) +1.
	End	The down counter stopped when user clears CNTENx (SCn_ALTCTL[7:5]) bit.

1111	Start	The down counter starts counting when user sets CNTENx (SCn_ALTCTL[7:5]) bit and it will count to time-out.
	Reload &recount	Only when the next START bit is detected, counter will reload the new value of CNT (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) and recount.
	Interrupt	If the counter decreases to 0 before the next START bit detected, hardware will generate time-out interrupt flag TMR0IF, TMR1IF, TMR2IF (SCn_INTSTS[5:3]). The time-out value will be CNTx (SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0], SCn_TMRCTL2[7:0]) +1.
	End	The down counter stopped when user clears CNTENx (SCn_ALTCTL[7:5]) bit.

Table 6.15-5 Timer0/Timer1/Timer2 Operation Mode

6.15.5.9 Block Guard Time and Extra Guard Time

Block guard time means the minimum interval between the leading edges of two consecutive characters between different transfer directions. This field indicates the counter for the bit length of block guard time. According to ISO7816-3, in T = 0 mode, user must fill 15 (real block guard time = 16.5) to this field; in T = 1 mode, user must fill 21 (real block guard time = 22.5) to it.

In transmit direction, the smart card sends data to smart card host controller, first. After the period is greater than BGT (SCn_CTL[12:8]), the smart card host controller begin to send the data.

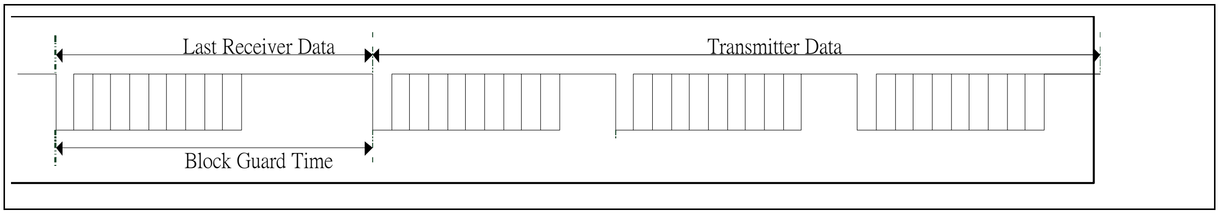


Figure 6.15-10 Transmit Direction Block Guard Time Operation

In receive direction, the smart card host controller sends data to smart card, first. If the smart card responses data to smart card host controller at the time which is less than BGT (SCn_CTL[12:8]), the block guard time interrupt BGTIF (SC_INTSTS[6]) is generated when RXBGTEN (SCn_ALTCTL[12]) and BGTIEN(SCn_INTEN[6]) is enabled.

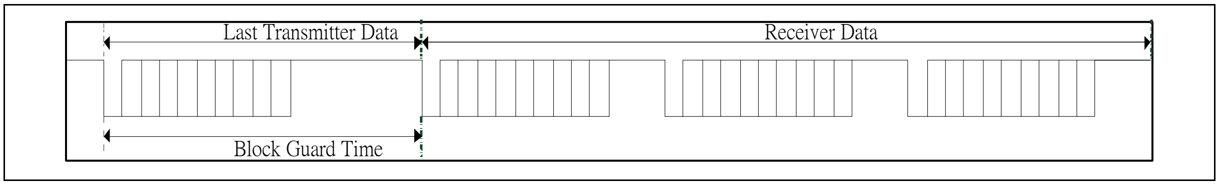


Figure 6.15-11 Receive Direction Block Guard Time Operation

Extra Guard Time is EGT (SCn_EGT[7:0]), it only affects the data transmitted by smart card interface, the format is shown as Figure 6.15-12Figure 6.15-11.

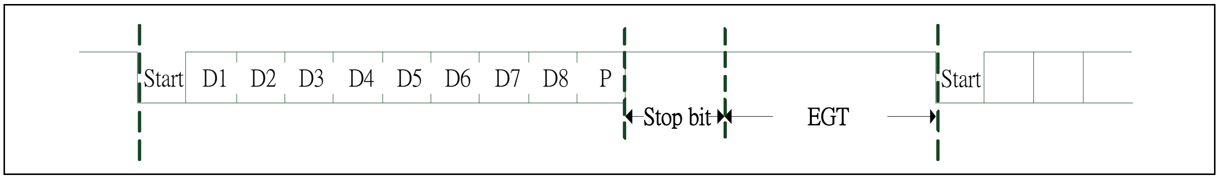


Figure 6.15-12 Extra Guard Time Operation

6.15.5.10 UART Mode

When the UARTEN (SCn_UARTCTL[0]) bit is set, the Smart Card Interface controller can also be used as basic UART function. The following is the program example for UART mode.

- Programming example:

1. Set UARTEN (SCn_UARTCTL[0]) bit to enter UART mode.
2. Do user reset by setting RXRST (SCn_ALTCTL[1]) and TXRST(SCn_ALTCTL[0]) bit to ensure that all state machine return idle state.
3. Fill “0” to CONSEL (SCn_CTL[5:4]) and AUTOCEN (SCn_CTL[3]) field. In UART mode, those fields must be “0”.
4. Select the UART baud rate by setting ETURDIV (SCn_ETUCTL[11:0]) fields. For example, if smartcard module clock is 12 MHz and target baud rate is 115200 bps, ETURDIV should fill with ((12000000 / 115200) - 1).
5. Select the data format include data length (by setting WLS (SCn_UARTCTL[5:4]), parity format (by setting OPE (SCn_UARTCTL[7]) and PBOFF (SCn_UARTCTL[6])) and stop bit length (by setting NSB (SCn_CTL[15] or EGT (SCn_EGT[7:0])).
6. Select the receiver buffer number trigger level by setting RXTRGLV (SCn_CTL[7:6]) field and select the receiver buffer time-out interval by setting RFTM (SCn_RXTOUT[8:0]) field.
7. Write the SCn_DAT (SCn_DAT[7:0]) (TX) register or read the SCn_DAT (SCn_DAT[7:0]) (RX) register can perform UART function.

6.15.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
SC Base Address: **SCn_BA = 0xB009_0000 + (0x1000 * n)** **n=0,1**				
SC_DAT	SCn_BA+0x00	R/W	SC Receive/Transmit Holding Buffer Register	0xXXXX_XXXX
SC_CTL	SCn_BA+0x04	R/W	SC Control Register	0x0000_0000
SC_ALTCTL	SCn_BA+0x08	R/W	SC Alternate Control Register	0x0000_0000
SC_EGT	SCn_BA+0x0C	R/W	SC Extra Guard Time Register	0x0000_0000
SC_RXTOUT	SCn_BA+0x10	R/W	SC Receive Buffer Time-out Counter Register	0x0000_0000
SC_ETUCTL	SCn_BA+0x14	R/W	SC Element Time Unit Control Register	0x0000_0173
SC_INTEN	SCn_BA+0x18	R/W	SC Interrupt Enable Control Register	0x0000_0000
SC_INTSTS	SCn_BA+0x1C	R/W	SC Interrupt Status Register	0x0000_0002
SC_STATUS	SCn_BA+0x20	R/W	SC Transfer Status Register	0x0000_X202
SC_PINCTL	SCn_BA+0x24	R/W	SC Pin Control State Register	0x0000_0000
SC_TMRCTL0	SCn_BA+0x28	R/W	SC Internal Timer0 Control Register	0x0000_0000
SC_TMRCTL1	SCn_BA+0x2C	R/W	SC Internal Timer1 Control Register	0x0000_0000
SC_TMRCTL2	SCn_BA+0x30	R/W	SC Internal Timer2 Control Register	0x0000_0000
SC_UARTCTL	SCn_BA+0x34	R/W	SC UART Mode Control Register	0x0000_0000
SC_ACTCTL	SCn_BA+0x4C	R/W	SC Activation Control Register	0x0000_0000

6.15.7 Register Description

SC Receive/Transmit Holding Buffer Register (SC_DAT)

Register	Offset	R/W	Description	Reset Value
SC_DAT	SCn_BA+0x00	R/W	SC Receive/Transmit Holding Buffer Register	0xXXXX_XXXX

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**DAT**	**Receive/Transmit Holding Buffer** Write Operation: By writing data to DAT, the SC will send out an 8-bit data. **Note:** If SCEN (SCn_CTL[0]) is not enabled, DAT cannot be programmed. Read Operation: By reading DAT, the SC will return an 8-bit received data.

SC Control Register (SC_CTL)

Register	Offset	R/W	Description	Reset Value
SC_CTL	SCn_BA+0x04	R/W	SC Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	**SYNC**	**Reserved**			**CDLV**	**CDDBSEL**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TXRTYEN	**TXRTY**			**RXRTYEN**	**RXRTY**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
NSB	**TMRSEL**		**BGT**				
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RXTRGLV		**CONSEL**		**AUTOCEN**	**TXOFF**	**RXOFF**	**SCEN**

Bits	Description	
[31]	**Reserved**	Reserved.
[30]	**SYNC**	**SYNC Flag Indicator (Read Only)** Due to synchronization, user should check this bit before writing a new value to RXRTY and TXRTY fields. 0 = Synchronizing is completion, user can write new data to RXRTY and TXRTY. 1 = Last value is synchronizing.
[29:27]	**Reserved**	Reserved.
[26]	**CDLV**	**Card Detect Level Selection** 0 = When hardware detects the card detect pin (SCn_CD) from high to low, it indicates a card is detected. 1 = When hardware detects the card detect pin (SCn_CD) from low to high, it indicates a card is detected. **Note:** User must select card detect level before Smart Card controller enabled.
[25:24]	**CDDBSEL**	**Card Detect De-bounce Selection** This field indicates the card detect de-bounce selection. 00 = De-bounce sample card insert once per 384 (128 * 3) SC module clocks and de-bounce sample card removal once per 128 SC module clocks. Other configurations are reserved.
[23]	**TXRTYEN**	**TX Error Retry Enable Bit** This bit enables transmitter retry function when parity error has occurred. 0 = TX error retry function Disabled. 1 = TX error retry function Enabled.
[22:20]	**TXRTY**	**TX Error Retry Count Number** This field indicates the maximum number of transmitter retries that are allowed when parity error has occurred. **Note1:** The real retry number is TXRTY + 1, so 8 is the maximum retry number. **Note2:** This field cannot be changed when TXRTYEN enabled. The change flow is to

		disable TXRTYEN first and then fill in new retry value.
[19]	**RXRTYEN**	**RX Error Retry Enable Bit** This bit enables receiver retry function when parity error has occurred. 0 = RX error retry function Disabled. 1 = RX error retry function Enabled. **Note:** User must fill in the RXRTY value before enabling this bit.
[18:16]	**RXRTY**	**RX Error Retry Count Number** This field indicates the maximum number of receiver retries that are allowed when parity error has occurred **Note1:** The real retry number is RXRTY + 1, so 8 is the maximum retry number. **Note2:** This field cannot be changed when RXRTYEN enabled. The change flow is to disable RXRTYEN first and then fill in new retry value.
[15]	**NSB**	**Stop Bit Length** This field indicates the length of stop bit. 0 = The stop bit length is 2 ETU. 1= The stop bit length is 1 ETU. **Note1:** The default stop bit length is 2. SC and UART adopts NSB to program the stop bit length. **Note2:** In UART mode, RX can receive the data sequence in 1 stop bit or 2 stop bits with NSB is set to 0.
[14:13]	**TMRSEL**	**Timer Channel Selection** 00 = All internal timer function Disabled. 11 = Internal 24 bit timer and two 8 bit timers Enabled. User can configure them by setting SCn_TMRCTL0[23:0], SCn_TMRCTL1[7:0] and SCn_TMRCTL2[7:0]. Other configurations are reserved
[12:8]	**BGT**	**Block Guard Time (BGT)** Block guard time means the minimum interval between the leading edges of two consecutive characters between different transfer directions. This field indicates the counter for the bit length of block guard time. According to ISO 7816-3, in T = 0 mode, user must fill 15 (real block guard time = 16.5) to this field; in T = 1 mode, user must fill 21 (real block guard time = 22.5) to it. **Note:** The real block guard time is BGT + 1.
[7:6]	**RXTRGLV**	**Rx Buffer Trigger Level** When the number of bytes in the receiving buffer equals the RXTRGLV, the RDAIF will be set. If RDAIEN (SCn_INTEN[0]) is enabled, an interrupt will be generated to CPU. 00 = Rx Buffer Trigger Level with 01 bytes. 01 = Rx Buffer Trigger Level with 02 bytes. 10 = Rx Buffer Trigger Level with 03 bytes. 11 = Reserved.
[5:4]	**CONSEL**	**Convention Selection** 00 = Direct convention. 01 = Reserved. 10 = Reserved. 11 = Inverse convention. **Note:** If AUTOCEN (SCn_CTL[3]) is enabled, this field is ignored.
[3]	**AUTOCEN**	**Auto Convention Enable Bit**

		This bit is used for enable auto convention function. 0 = Auto-convention Disabled. 1 = Auto-convention Enabled. If user enables auto convention function, the setting step must be done before Answer to Reset (ATR) state and the first data must be 0x3B or 0x3F. After hardware received first data and stored it at buffer, hardware will decided the convention and change the CONSEL (SCn_CTL[5:4]) bits automatically when received first data is 0x3B or 0x3F. If received first byte is 0x3B, TS is direct convention, CONSEL (SCn_CTL[5:4]) will be set to 00 automatically, otherwise the TS is inverse convention, and CONSEL (SCn_CTL[5:4]) will be set to 11. If the first data is not 0x3B or 0x3F, hardware will set ACERRIF (SCn_INTSTS[10]) and generate an interrupt to CPU when ACERRIEN (SCn_INTEN[10]) is enabled.
[2]	**TXOFF**	**TX Transition Disable Control Bit** This bit is used for disable Tx transition function. 0 = The transceiver Enabled. 1 = The transceiver Disabled.
[1]	**RXOFF**	**RX Transition Disable Control Bit** This bit is used for disable Rx transition function. 0 = The receiver Enabled. 1 = The receiver Disabled. **Note:** If AUTOCEN (SCn_CTL[3]) is enabled, this field is ignored.
[0]	**SCEN**	**SC Controller Enable Bit** Set this bit to 1 to enable SC operation. If this bit is cleared, 0 = SC will force all transition to IDLE state. 1 = SC controller is enabled and all function can work correctly. **Note:** SCEN must be set to 1 before filling in other SC registers, or smart card will not work properly.

SC Alternate Control Register (SC_ALTCTL)

Register	Offset	R/W	Description	Reset Value
SC_ALTCTL	SCn_BA+0x08	R/W	SC Alternate Control Register	0x0000_0000

31	30	29	28	27	26	25	24
SYNC	Reserved						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ACTSTS2	ACTSTS1	ACTSTS0	RXBGTEN	ADACEN	Reserved	INITSEL	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNTEN2	CNTEN1	CNTEN0	WARSTEN	ACTEN	DACTEN	RXRST	TXRST

Bits	Description	
[31]	**SYNC**	**SYNC Flag Indicator (Read Only)** Due to synchronization, user should check this bit when writing a new value to SCn_ALTCTL register. 0 = Synchronizing is completion, user can write new data to SCn_ALTCTL register. 1 = Last value is synchronizing.
[30:16]	**Reserved**	Reserved.
[15]	**ACTSTS2**	**Internal Timer2 Active Status (Read Only)** This bit indicates the timer counter status of timer2. 0 = Timer2 is not active. 1 = Timer2 is active. **Note:** Timer2 is active does not always mean timer2 is counting the CNT (SCn_TMRCTL2[7:0]).
[14]	**ACTSTS1**	**Internal Timer1 Active Status (Read Only)** This bit indicates the timer counter status of timer1. 0 = Timer1 is not active. 1 = Timer1 is active. **Note:** Timer1 is active does not always mean timer1 is counting the CNT (SCn_TMRCTL1[7:0]).
[13]	**ACTSTS0**	**Internal Timer0 Active Status (Read Only)** This bit indicates the timer counter status of timer0. 0 = Timer0 is not active. 1 = Timer0 is active. **Note:** Timer0 is active does not always mean timer0 is counting the CNT (SCn_TMRCTL0[23:0]).
[12]	**RXBGTEN**	**Receiver Block Guard Time Function Enable Bit** This bit enables the receiver block guard time function.

		0 = Receiver block guard time function Disabled. 1 = Receiver block guard time function Enabled.
[11]	**ADACEN**	**Auto Deactivation When Card Removal** This bit is used for enable hardware auto deactivation when smart card is removed. 0 = Auto deactivation Disabled. 1 = Auto deactivation Enabled. **Note:** When the card is removed, hardware will stop any process and then do deactivation sequence if this bit is set. If auto deactivation process completes, hardware will set INITIF (SCn_INTSTS[8]) also.
[10]	**Reserved**	Reserved.
[9:8]	**INITSEL**	**Initial Timing Selection** This fields indicates the initial timing of hardware activation, warm-reset or deactivation. The unit of initial timing is SC module clock. Activation: refer to SC Activation Sequence in Figure 6.15-54. Warm-reset: refer to Warm-Reset Sequence in Figure 6.15-5. Deactivation: refer to Deactivation Sequence in Figure 6.15-56. **Note:** When set activation and warm reset in Timer0 operation mode 0011, it may have deviation at most 128 SC module clock cycles.
[7]	**CNTEN2**	**Internal Timer2 Start Enable Bit** This bit enables Timer 2 to start counting. User can fill 0 to stop it and set 1 to reload and count. The counter unit is ETU base. 0 = Stops counting. 1 = Start counting. **Note1:** This field is used for internal 8 bit timer when TMRSEL (SCn_CTL[14:13]) is 11 only. Do not fill in CNTEN2 when TMRSEL (SCn_CTL[14:13]) is not equal to 11. **Note2:** If the operation mode is not in auto-reload mode (SCn_TMRCTL2[26] = 0), this bit will be auto-cleared by hardware. **Note3:** If SCEN (SCn_CTL[0]) is not enabled, this filed cannot be programmed.
[6]	**CNTEN1**	**Internal Timer1 Start Enable Bit** This bit enables Timer 1 to start counting. User can fill 0 to stop it and set 1 to reload and count. The counter unit is ETU base. 0 = Stops counting. 1 = Start counting. **Note1:** This field is used for internal 8 bit timer when TMRSEL(SCn_CTL[14:13]) is 11 only. Do not fill CNTEN1 when TMRSEL (SCn_CTL[14:13]) is not equal to 11. **Note2:** If the operation mode is not in auto-reload mode (SCn_TMRCTL1[26] = 0), this bit will be auto-cleared by hardware. **Note3:** If SCEN (SCn_CTL[0]) is not enabled, this filed cannot be programmed.
[5]	**CNTEN0**	**Internal Timer0 Start Enable Bit** This bit enables Timer 0 to start counting. User can fill 0 to stop it and set 1 to reload and count. The counter unit is ETU base. 0 = Stops counting. 1 = Start counting. **Note1:** This field is used for internal 24 bit timer when TMRSEL (SCn_CTL[14:13]) is 11 only. **Note2:** If the operation mode is not in auto-reload mode (SCn_TMRCTL0[26] = 0), this bit will be auto-cleared by hardware. **Note3:** If SCEN (SCn_CTL[0]) is not enabled, this filed cannot be programmed.

[4]	**WARSTEN**	**Warm Reset Sequence Generator Enable Bit** This bit enables SC controller to initiate the card by warm reset sequence. 0 = No effect. 1 = Warm reset sequence generator Enabled. **Note1:** When the warm reset sequence completed, this bit will be cleared automatically and the INITIF (SCn_INTSTS[8]) will be set to 1. **Note2:** This field will be cleared by TXRST (SCn_ALTCTL[0]) and RXRST (SCn_ALTCTL[1]). Thus, do not fill in this bit WARSTEN, TXRST and RXRST at the same time. **Note3:** If SCEN (SCn_CTL[0]) is not enabled, this filed cannot be programmed. **Note4**: During the warm reset sequence, RX is disabled automatically and can not receive data. After the warm reset sequence completion, RXOFF (SCn_CTL[1]) keeps the state before perform warm reset sequence.
[3]	**ACTEN**	**Activation Sequence Generator Enable Bit** This bit enables SC controller to initiate the card by activation sequence. 0 = No effect. 1 = Activation sequence generator Enabled. **Note1:** When the activation sequence completed, this bit will be cleared automatically and the INITIF (SCn_INTSTS[8]) will be set to 1. **Note2:** This field will be cleared by TXRST (SCn_ALTCTL[0]) and RXRST (SCn_ALTCTL[1]). Thus, do not fill in this bit ACTEN, TXRST and RXRST at the same time. **Note3:** If SCEN (SCn_CTL[0]) is not enabled, this filed cannot be programmed. **Note4**: During the activation sequence, RX is disabled automatically and can not receive data. After the activation sequence completion, RXOFF (SCn_CTL[1]) keeps the state before hardware activation.
[2]	**DACTEN**	**Deactivation Sequence Generator Enable Bit** This bit enables SC controller to initiate the card by deactivation sequence. 0 = No effect. 1 = Deactivation sequence generator Enabled. **Note1:** When the deactivation sequence completed, this bit will be cleared automatically and the INITIF (SCn_INTSTS[8]) will be set to 1. **Note2:** This field will be cleared by TXRST (SCn_ALTCTL[0]) and RXRST (SCn_ALTCTL[1]). Thus, do not fill in this bit DACTEN, TXRST and RXRST at the same time. **Note3:** If SCEN (SCn_CTL[0]) is not enabled, this filed cannot be programmed.
[1]	**RXRST**	**Rx Software Reset** When RXRST is set, all the bytes in the receive buffer and Rx internal state machine will be cleared. 0 = No effect. 1 = Reset the Rx internal state machine and pointers. **Note:** This bit will be auto cleared after reset is complete.
[0]	**TXRST**	**TX Software Reset** When TXRST is set, all the bytes in the transmit buffer and TX internal state machine will be cleared. 0 = No effect. 1 = Reset the TX internal state machine and pointers. **Note:** This bit will be auto cleared after reset is complete.

SC Extra Guard Time Register (SC_EGT)

Register	Offset	R/W	Description	Reset Value
SC_EGT	SCn_BA+0x0C	R/W	SC Extra Guard Time Register	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**EGT**	**Extra Guard Time** This field indicates the extra guard time value. **Note:** The extra guard time unit is ETU base.

SC Receiver Buffer Time-out Register (SC_RXTOUT)

Register	Offset	R/W	Description	Reset Value
SC_RXTOUT	SCn_BA+0x10	R/W	SC Receive Buffer Time-out Counter Register	0x0000_0000

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

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8:0]	**RFTM**	**SC Receiver FIFO Time-out Counter** The time-out down counter resets and starts counting whenever the RX buffer received a new data. Once the counter decrease to 1 and no new data is received or CPU does not read data by reading SCn_DAT, a receiver time-out flag RXTOIF (SCn_INTSTS[9]) will be set, and hardware will generate an interrupt to CPU when RXTOIEN (SCn_INTEN[9]) is enabled. **Note1:** The counter unit is ETU based and the interval of time-out is RFTM + 0.5. **Note2:** Filling in all 0 to this field indicates to disable this function.

SC Element Time Unit Control Register (SC_ETUCTL)

Register	Offset	R/W	Description	Reset Value
SC_ETUCTL	SCn_BA+0x14	R/W	SC Element Time Unit Control Register	0x0000_0173

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

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:0]	**ETURDIV**	**ETU Rate Divider** The field is used for ETU clock rate divider. The real ETU is ETURDIV + 1. **Note:** User can configure this field, but this field must be greater than 0x04.

SC Interrupt Enable Control Register (SC_INTEN)

Register	Offset	R/W	Description	Reset Value
SC_INTEN	SCn_BA+0x18	R/W	SC 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					ACERRIEN	RXTOIEN	INITIEN
7	6	5	4	3	2	1	0
CDIEN	BGTIEN	TMR2IEN	TMR1IEN	TMR0IEN	TERRIEN	TBEIEN	RDAIEN

Bits	Description	
[31:11]	Reserved	Reserved.
[10]	ACERRIEN	**Auto Convention Error Interrupt Enable Bit** This field is used to enable auto-convention error interrupt. 0 = Auto-convention error interrupt Disabled. 1 = Auto-convention error interrupt Enabled.
[9]	RXTOIEN	**Receiver Buffer Time-out Interrupt Enable Bit** This field is used to enable receiver buffer time-out interrupt. 0 = Receiver buffer time-out interrupt Disabled. 1 = Receiver buffer time-out interrupt Enabled.
[8]	INITIEN	**Initial End Interrupt Enable Bit** This field is used to enable activation (ACTEN (SCn_ALTCTL[3] = 1)), deactivation (DACTEN (SCn_ALTCTL[2] = 1)) and warm reset (WARSTEN (SCn_ALTCTL [4])) sequence complete interrupt. 0 = Initial end interrupt Disabled. 1 = Initial end interrupt Enabled.
[7]	CDIEN	**Card Detect Interrupt Enable Bit** This field is used to enable card detect interrupt. The card detect status is CDPINSTS (SCn_STATUS[13]). 0 = Card detect interrupt Disabled. 1 = Card detect interrupt Enabled.
[6]	BGTIEN	**Block Guard Time Interrupt Enable Bit** This field is used to enable block guard time interrupt in receive direction. 0 = Block guard time interrupt Disabled. 1 = Block guard time interrupt Enabled. **Note:** This bit is valid only for receive direction block guard time.
[5]	TMR2IEN	**Timer2 Interrupt Enable Bit**

		This field is used to enable Timer2 interrupt function. 0 = Timer2 interrupt Disabled. 1 = Timer2 interrupt Enabled.
[4]	**TMR1IEN**	**Timer1 Interrupt Enable Bit** This field is used to enable the Timer1 interrupt function. 0 = Timer1 interrupt Disabled. 1 = Timer1 interrupt Enabled.
[3]	**TMR0IEN**	**Timer0 Interrupt Enable Bit** This field is used to enable Timer0 interrupt function. 0 = Timer0 interrupt Disabled. 1 = Timer0 interrupt Enabled.
[2]	**TERRIEN**	**Transfer Error Interrupt Enable Bit** This field is used to enable transfer error interrupt. The transfer error states is at SCn_STATUS register which includes receiver break error BEF (SCn_STATUS[6]), frame error FEF (SCn_STATUS[5]), parity error PEF (SCn_STATUS[4]), receive buffer overflow error RXOV (SCn_STATUS[0]), transmit buffer overflow error TXOV (SCn_STATUS[8]), receiver retry over limit error RXOVERR (SCn_STATUS[22]) and transmitter retry over limit error TXOVERR (SCn_STATUS[30]). 0 = Transfer error interrupt Disabled. 1 = Transfer error interrupt Enabled.
[1]	**TBEIEN**	**Transmit Buffer Empty Interrupt Enable Bit** This field is used to enable transmit buffer empty interrupt. 0 = Transmit buffer empty interrupt Disabled. 1 = Transmit buffer empty interrupt Enabled.
[0]	**RDAIEN**	**Receive Data Reach Interrupt Enable Bit** This field is used to enable received data reaching trigger level RXTRGLV (SCn_CTL[7:6]) interrupt. 0 = Receive data reach trigger level interrupt Disabled. 1 = Receive data reach trigger level interrupt Enabled.

SC Interrupt Status Register (SC_INTSTS)

Register	Offset	R/W	Description	Reset Value
SC_INTSTS	SCn_BA+0x1C	R/W	SC Interrupt Status Register	0x0000_0002

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					ACERRIF	RXTOIF	INITIF
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CDIF	BGTIF	TMR2IF	TMR1IF	TMR0IF	TERRIF	TBEIF	RDAIF

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10]	**ACERRIF**	**Auto Convention Error Interrupt Status Flag** This field indicates auto convention sequence error. 0 = Received TS at ATR state is 0x3B or 0x3F. 1 = Received TS at ATR state is neither 0x3B nor 0x3F. **Note:** This bit can be cleared by writing 1 to it.
[9]	**RXTOIF**	**Receive Buffer Time-out Interrupt Status Flag (Read Only)** This field is used for indicate receive buffer time-out interrupt status flag. 0 = Receive buffer time-out interrupt did not occur. 1 = Receive buffer time-out interrupt occurred. **Note:** This bit is read only, user must read all receive buffer remaining data by reading SCn_DAT register to clear it.
[8]	**INITIF**	**Initial End Interrupt Status Flag** This field is used for activation (ACTEN (SCn_ALTCTL[3])), deactivation (DACTEN (SCn_ALTCTL[2])) and warm reset (WARSTEN (SCn_ALTCTL[4])) sequence interrupt status flag. 0 = Initial sequence is not complete. 1 = Initial sequence is completed. **Note:** This bit can be cleared by writing 1 to it.
[7]	**CDIF**	**Card Detect Interrupt Status Flag (Read Only)** This field is used for card detect interrupt status flag. The card detect status is CINSERT (SCn_STATUS[12]) and CREMOVE (SCn_STATUS[11]). 0 = Card detect event did not occur. 1 = Card detect event occurred. **Note:** This bit is read only, user must to clear CINSERT or CREMOVE status to clear it.
[6]	**BGTIF**	**Block Guard Time Interrupt Status Flag**

		This field is used for indicate block guard time interrupt status flag in receive direction. 0 = Block guard time interrupt did not occur. 1 = Block guard time interrupt occurred. **Note1:** This bit is valid only when RXBGTEN (SCn_ALTCTL[12]) is enabled. **Note2:** This bit can be cleared by writing 1 to it.
[5]	**TMR2IF**	**Timer2 Interrupt Status Flag** This field is used for Timer2 interrupt status flag. 0 = Timer2 interrupt did not occur. 1 = Timer2 interrupt occurred. **Note:** This bit can be cleared by writing 1 to it.
[4]	**TMR1IF**	**Timer1 Interrupt Status Flag** This field is used for Timer1 interrupt status flag. 0 = Timer1 interrupt did not occur. 1 = Timer1 interrupt occurred. **Note:** This bit can be cleared by writing 1 to it.
[3]	**TMR0IF**	**Timer0 Interrupt Status Flag** This field is used for Timer0 interrupt status flag. 0 = Timer0 interrupt did not occur. 1 = Timer0 interrupt occurred. **Note:** This bit can be cleared by writing 1 to it.
[2]	**TERRIF**	**Transfer Error Interrupt Status Flag** This field is used for transfer error interrupt status flag. The transfer error states is at SCn_STATUS register which includes receiver break error BEF (SCn_STATUS[6]), frame error FEF (SCn_STATUS[5], parity error PEF (SCn_STATUS[4] and receive buffer overflow error RXOV (SCn_STATUS[0]), transmit buffer overflow error TXOV (SCn_STATUS[8]), receiver retry over limit error RXOVERR (SCn_STATUS[22] or transmitter retry over limit error TXOVERR (SCn_STATUS[30]). 0 = Transfer error interrupt did not occur. 1 = Transfer error interrupt occurred. **Note1:** This field is the status flag of BEF, FEF, PEF, RXOV, TXOV, RXOVERR or TXOVERR. **Note2:** This bit can be cleared by writing 1 to it.
[1]	**TBEIF**	**Transmit Buffer Empty Interrupt Status Flag (Read Only)** This field is used for transmit buffer empty interrupt status flag. 0 = Transmit buffer is not empty. 1 = Transmit buffer is empty. **Note:** This bit is read only. If user wants to clear this bit, user must write data to DAT (SCn_DAT[7:0]) and then this bit will be cleared automatically.
[0]	**RDAIF**	**Receive Data Reach Interrupt Status Flag (Read Only)** This field is used for received data reaching trigger level RXTRGLV (SCn_CTL[7:6]) interrupt status flag. 0 = Number of receive buffer is less than RXTRGLV setting. 1 = Number of receive buffer data equals the RXTRGLV setting. **Note:** This bit is read only. If user reads data from SCn_DAT and receiver buffer data byte number is less than RXTRGLV, this bit will be cleared automatically.

SC Transfer Status Register (SC_STATUS)

Register	Offset	R/W	Description	Reset Value
SC_STATUS	SCn_BA+0x20	R/W	SC Transfer Status Register	0x0000_X202

31	30	29	28	27	26	25	24
TXACT	TXOVERR	TXRERR	Reserved		TXPOINT		
23	22	21	20	19	18	17	16
RXACT	RXOVERR	RXRERR	Reserved		RXPOINT		
15	14	13	12	11	10	9	8
Reserved		CDPINSTS	CINSERT	CREMOVE	TXFULL	TXEMPTY	TXOV
7	6	5	4	3	2	1	0
Reserved	BEF	FEF	PEF	Reserved	RXFULL	RXEMPTY	RXOV

Bits	Description	
[31]	TXACT	**Transmit in Active Status Flag (Read Only)** This bit indicates Tx transmit status. 0 = This bit is cleared automatically when Tx transfer is finished or the last byte transmission has completed. 1 = Transmit is active and this bit is set by hardware when Tx transfer is in active and the STOP bit of the last byte has not been transmitted.
[30]	TXOVERR	**Transmitter over Retry Error** This bit is used for transmitter retry counts over than retry number limitation. 0 = Transmitter retries counts is less than TXRTY (SCn_CTL[22:20]) + 1. 1 = Transmitter retries counts is equal or over to TXRTY (SCn_CTL[22:20]) + 1. **Note:** This bit can be cleared by writing 1 to it.
[29]	TXRERR	**Transmitter Retry Error** This bit is used for indicate transmitter error retry and set by hardware.. 0 = No Tx retry transfer. 1 = Tx has any error and retries transfer. **Note1:** This bit can be cleared by writing 1 to it. **Note2:** This bit is a flag and cannot generate any interrupt to CPU.
[28:27]	Reserved	Reserved.
[26:24]	TXPOINT	**Transmit Buffer Pointer Status (Read Only)** This field indicates the Tx buffer pointer status. When CPU writes data into SCn_DAT, TXPOINT increases one. When one byte of Tx buffer is transferred to transmitter shift register, TXPOINT decreases one.
[23]	RXACT	**Receiver in Active Status Flag (Read Only)** This bit indicates Rx transfer status. 0 = This bit is cleared automatically when Rx transfer is finished. 1 = This bit is set by hardware when Rx transfer is in active.

[22]	**RXOVERR**	**Receiver over Retry Error** This bit is used for receiver retry counts over than retry number limitation. 0 = Receiver retries counts is less than RXRTY (SCn_CTL[18:16]) + 1. 1 = Receiver retries counts is equal or over than RXRTY (SCn_CTL[18:16]) + 1. **Note1:** This bit can be cleared by writing 1 to it. **Note2:** If CPU enables receiver retries function by setting RXRTYEN (SCn_CTL[19]), hardware will not set this flag.
[21]	**RXRERR**	**Receiver Retry Error** This bit is used for receiver error retry and set by hardware. 0 = No Rx retry transfer. 1 = Rx has any error and retries transfer. **Note1:** This bit can be cleared by writing 1 to it. **Note2** This bit is a flag and cannot generate any interrupt to CPU. **Note3:** If CPU enables receiver retries function by setting RXRTYEN (SCn_CTL[19]), hardware will not set this flag.
[20:19]	**Reserved**	Reserved.
[18:16]	**RXPOINT**	**Receive Buffer Pointer Status (Read Only)** This field indicates the Rx buffer pointer status. When SC controller receives one byte from external device, RXPOINT increases one. When one byte of Rx buffer is read by CPU, RXPOINT decreases one.
[15:14]	**Reserved**	Reserved.
[13]	**CDPINSTS**	**Card Detect Pin Status (Read Only)** This bit is the pin status of SCn_CD. 0 = The SCn_CD pin state at low. 1 = The SCn_CD pin state at high.
[12]	**CINSERT**	**Card Insert Status of SCn_CD Pin** This bit is set whenever card has been inserted. 0 = No effect. 1 = Card insert. **Note1:** This bit can be cleared by writing “1” to it. **Note2:** The card detect function will start after SCEN (SCn_CTL[0]) set.
[11]	**CREMOVE**	**Card Removal Status of SCn_CD Pin** This bit is set whenever card has been removal. 0 = No effect. 1 = Card removed. **Note1:** This bit can be cleared by writing “1” to it. **Note2:** Card detect function will start after SCEN (SCn_CTL[0]) set.
[10]	**TXFULL**	**Transmit Buffer Full Status Flag (Read Only)** This bit indicates Tx buffer full or not. 0 = Tx buffer count is less than 4. 1 = Tx buffer count equals to 4.
[9]	**TXEMPTY**	**Transmit Buffer Empty Status Flag (Read Only)** This bit indicates TX buffer empty or not. 0 = Tx buffer is not empty. 1 = Tx buffer is empty, it means the last byte of Tx buffer has been transferred to

		Transmitter Shift Register. **Note:** This bit will be cleared when writing data into DAT (SCn_DAT[7:0]).
[8]	**TXOV**	**Transmit Overflow Error Interrupt Status Flag** This bit is set when Tx buffer overflow. 0 = Tx buffer is not overflow. 1 = Tx buffer is overflow when Tx buffer is full and an additional write operation to DAT (SCn_DAT[7:0]). **Note:** This bit can be cleared by writing 1 to it.
[7]	**Reserved**	Reserved.
[6]	**BEF**	**Receiver Break Error Status Flag** This bit is set to logic 1 whenever the received data input (Rx) held in the "spacing state" (logic 0) is longer than a full word transmission time (that is, the total time of “start bit” + “data bits” + “parity bit” + “stop bits”). 0 = Receiver break error flag did not occur. 1 = Receiver break error flag occurred. **Note1:** This bit can be cleared by writing 1 to it. **Note2:** If CPU sets receiver retries function by setting RXRTYEN (SCn_CTL[19]), hardware will not set this flag.
[5]	**FEF**	**Receiver Frame Error Status 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 = Receiver frame error flag did not occur. 1 = Receiver frame error flag occurred. **Note1:** This bit can be cleared by writing 1 to it. **Note2:** If CPU sets receiver retries function by setting RXRTYEN (SCn_CTL[19]), hardware will not set this flag.
[4]	**PEF**	**Receiver Parity Error Status Flag** This bit is set to logic 1 whenever the received character does not have a valid "parity bit". 0 = Receiver parity error flag did not occur. 1 = Receiver parity error flag occurred. **Note1:** This bit can be cleared by writing 1 to it. **Note2:** If CPU sets receiver retries function by setting RXRTYEN (SCn_CTL[19]), hardware will not set this flag.
[3]	**Reserved**	Reserved.
[2]	**RXFULL**	**Receive Buffer Full Status Flag (Read Only)** This bit indicates Rx buffer full or not. 0 = Rx buffer count is less than 4. 1 = Rx buffer count equals to 4.
[1]	**RXEMPTY**	**Receive Buffer Empty Status Flag (Read Only)** This bit indicates Rx buffer empty or not. 0 = Rx buffer is not empty. 1 = Rx buffer is empty, it means the last byte of Rx buffer has read from DAT (SCn_DAT[7:0]) by CPU.
[0]	**RXOV**	**Receive Overflow Error Status Flag** This bit is set when Rx buffer overflow. 0 = Rx buffer is not overflow. 1 = Rx buffer is overflow when the number of received bytes is greater than Rx buffer size

		(4 bytes). **Note:** This bit can be cleared by writing 1 to it.

SC Pin Control State Register (SC_PINCTL)

Register	Offset	R/W	Description	Reset Value
SC_PINCTL	SCn_BA+0x24	R/W	SC Pin Control State Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	SYNC	Reserved					
23	22	21	20	19	18	17	16
Reserved					RSTSTS	PWRSTS	DATASTS
15	14	13	12	11	10	9	8
Reserved				PWRINV	Reserved	SCDATA	Reserved
7	6	5	4	3	2	1	0
Reserved	CLKKEEP	Reserved				RSTEN	PWREN

Bits	Description	
[31]	Reserved	Reserved.
[30]	SYNC	**SYNC Flag Indicator (Read Only)** Due to synchronization, user should check this bit when writing a new value to SCn_PINCTL register. 0 = Synchronizing is completion, user can write new data to SCn_PINCTL register. 1 = Last value is synchronizing.
[29:19]	Reserved	Reserved.
[18]	RSTSTS	**SCn_RST Pin Status (Read Only)** This bit is the pin status of SCn_RST. 0 = SCn_RST pin is low. 1 = SCn_RST pin is high.
[17]	PWRSTS	**SCn_PWR Pin Status (Read Only)** This bit is the pin status of SCn_PWR. 0 = SCn_PWR pin to low. 1 = SCn_PWR pin to high.
[16]	DATASTS	**SCn_DATA Pin Status (Read Only)** This bit is the pin status of SCn_DATA. 0 = The SCn_DATA pin status is low. 1 = The SCn_DATA pin status is high.
[15:12]	Reserved	Reserved.
[11]	PWRINV	**SCn_PWR Pin Inverse** This bit is used for inverse the SCn_PWR pin. There are four kinds of combination for SCn_PWR pin setting by PWRINV (SCn_PINCTL[11]) and PWREN (SCn_PINCTL[0]). PWRINV (SCn_PINCTL[11]) is bit 1 and PWREN (SCn_PINCTL[0]) is bit 0 and all conditions as

NUC980 SERIES TECHNICAL REFERENCE MANUAL

		listed below. 00 = SCn_PWR pin is 0. 01 = SCn_PWR pin is 1. 10 = SCn_PWR pin is 1. 11 = SCn_PWR pin is 0. **Note:** User must select PWRINV (SCn_PINCTL[11]) before smart card is enabled by SCEN (SCn_CTL[0]).
[10]	**Reserved**	Reserved.
[9]	**SCDATA**	**SCn_DATA Pin Signal** This bit is the signal status of SCn_DATA but user can drive SCn_DATA pin to high or low by setting this bit. 0 = Drive SCn_DATA pin to low. 1 = Drive SCn_DATA pin to high. Read this field to get SCn_DATA signal status. 0 = SCn_DATA signal status is low. 1 = SCn_DATA signal status is high. **Note:** When SC is at activation, warm reset or deactivation mode, this bit will be changed automatically. Thus, do not fill in this field when SC is in these modes.
[8:7]	**Reserved**	Reserved.
[6]	**CLKKEEP**	**SC Clock Enable Bit** 0 = SC clock generation Disabled. 1 = SC clock always keeps free running. **Note:** When operating in activation, warm reset or deactivation mode, this bit will be changed automatically. Thus, do not fill in this field when operating in these modes.
[5:2]	**Reserved**	Reserved.
[1]	**RSTEN**	**SCn_RST Pin Signal** User can set RSTEN (SCn_PINCTL[1]) to decide SCn_RST pin is in high or low level. Write this field to drive SCn_RST pin. 0 = Drive SCn_RST pin to low. 1 = Drive SCn_RST pin to high. Read this field to get SCn_RST signal status. 0 = SCn_RST signal status is low. 1 = SCn_RST signal status is high. **Note:** When operating at activation, warm reset or deactivation mode, this bit will be changed automatically. Thus, do not fill in this field when operating in these modes.
[0]	**PWREN**	**SCn_PWR Pin Signal** User can set PWRINV (SCn_PINCTL[11]) and PWREN (SCn_PINCTL[0]) to decide SCn_PWR pin is in high or low level. Write this field to drive SCn_PWR pin Refer PWRINV (SCn_PINCTL[11]) description for programming SCn_PWR pin voltage level. Read this field to get SCn_PWR signal status. 0 = SCn_PWR signal status is low. 1 = SCn_PWR signal status is high. **Note:** When operating at activation, warm reset or deactivation mode, this bit will be changed automatically. Thus, do not fill in this field when operating in these modes.

SC Timer0 Control Register (SC_TMRCTL0)

Register	Offset	R/W	Description	Reset Value
SC_TMRCTL0	SCn_BA+0x28	R/W	SC Internal Timer0 Control Register	0x0000_0000

31	30	29	28	27	26	25	24
SYNC	Reserved			OPMODE			
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]	**SYNC**	**SYNC Flag Indicator (Read Only)** Due to synchronization, user should check this bit when writing a new value to the SCn_TMRCTL0 register. 0 = Synchronizing is completion, user can write new data to SCn_TMRCTL0 register. 1 = Last value is synchronizing.
[30:28]	**Reserved**	Reserved.
[27:24]	**OPMODE**	**Timer0 Operation Mode Selection** This field indicates the internal 24-bit Timer0 operation selection. Refer to Table 6.15-5 for programming Timer0.
[23:0]	**CNT**	**Timer0 Counter Value** This field indicates the internal Timer0 counter values. **Note:** Unit of Timer0 counter is ETU base.

SC Timer1 Control Register (SC_TMRCTL1)

Register	Offset	R/W	Description	Reset Value
SC_TMRCTL1	SCn_BA+0x2C	R/W	SC Internal Timer1 Control Register	0x0000_0000

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

Bits	Description	
[31]	**SYNC**	**SYNC Flag Indicator (Read Only)** Due to synchronization, software should check this bit when writing a new value to SCn_TMRCTL1 register. 0 = Synchronizing is completion, user can write new data to SCn_TMRCTL1 register. 1 = Last value is synchronizing.
[30:28]	**Reserved**	Reserved.
[27:24]	**OPMODE**	**Timer 1 Operation Mode Selection** This field indicates the internal 8-bit Timer1 operation selection. Refer to Table 6.15-5 for programming Timer1.
[23:8]	**Reserved**	Reserved.
[7:0]	**CNT**	**Timer 1 Counter Value** This field indicates the internal Timer1 counter values. **Note:** Unit of Timer1 counter is ETU base.

SC Timer2 Control Register (SC_TMRCTL2)

Register	Offset	R/W	Description	Reset Value
SC_TMRCTL2	SCn_BA+0x30	R/W	SC Internal Timer2 Control Register	0x0000_0000

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

Bits	Description	
[31]	SYNC	**SYNC Flag Indicator (Read Only)** Due to synchronization, user should check this bit when writing a new value to SCn_TMRCTL2 register. 0 = Synchronizing is completion, user can write new data to SCn_TMRCTL2 register. 1 = Last value is synchronizing.
[30:28]	Reserved	Reserved.
[27:24]	OPMODE	**Timer 2 Operation Mode Selection** This field indicates the internal 8-bit Timer2 operation selection Refer to Table 6.15-5 for programming Timer2.
[23:8]	Reserved	Reserved.
[7:0]	CNT	**Timer 2 Counter Value** This field indicates the internal Timer2 counter values. **Note:** Unit of Timer2 counter is ETU base.

SC UART Mode Control Register (SC_UARTCTL)

Register	Offset	R/W	Description	Reset Value
SC_UARTCTL	SCn_BA+0x34	R/W	SC UART Mode 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
OPE	PBOFF	WLS		Reserved			UARTEN

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**OPE**	**Odd Parity Enable Bit** This is used for odd/even parity selection. 0 = Even number of logic 1 are transmitted or check the data word and parity bits in receiving mode. 1 = Odd number of logic 1 are transmitted or check the data word and parity bits in receiving mode. **Note:** This bit has effect only when PBOFF bit is 0.
[6]	**PBOFF**	**Parity Bit Disable Bit** Sets this bit is used for disable parity check function. 0 = Parity bit is generated or checked between the “last data word bit” and “stop bit” of the serial data. 1 = Parity bit is not generated (transmitting data) or checked (receiving data) during transfer. **Note:** In smart card mode, this field must be 0 (default setting is with parity bit).
[5:4]	**WLS**	**Word Length Selection** This field is used for select UART data length. 00 = Word length is 8 bits. 01 = Word length is 7 bits. 10 = Word length is 6 bits. 11 = Word length is 5 bits. **Note:** In smart card mode, this WLS must be 00.
[3:1]	**Reserved**	Reserved.
[0]	**UARTEN**	**UART Mode Enable Bit** Sets this bit to enable UART mode function. 0 = Smart Card mode.

Bits	Description	
		1 = UART mode. **Note1:** When operating in UART mode, user must set CONSEL (SCn_CTL[5:4]) = 00 and AUTOCEN (SCn_CTL[3]) = 0. **Note2:** When operating in Smart Card mode, user must set UARTEN (SCn_UARTCTL[0]) = 0. **Note3:** When UART mode is enabled, hardware will generate a reset to reset FIFO and internal state machine.

SC Activation Control Register (SC_ACTCTL)

Register	Offset	R/W	Description	Reset Value
SC_ACTCTL	SCn_BA+0x4C	R/W	SC Activation 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			**T1EXT**				

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4:0]	**T1EXT**	**T1 Extend Time of Hardware Activation** This field provide the configurable cycles to extend the activation time T1 period. The cycle scaling factor is 2048. Extend cycles = (filled value * 2048) cycles. Refer to SC activation sequence in Figure 6.15-4. For example, SCLK = 4 MHz, each cycle = 0.25us,. Filled 20 to this field Extend time = 20 * 2048 * 0.25us = 10.24 ms. **Note:** Setting 0 to this field conforms to the protocol ISO/IEC 7816-3

6.16 I^{2}C Serial Interface Controller (I^{2}C)

6.16.1 Overview

I^{2}C is a two-wire, bi-directional serial bus that provides a simple and efficient method of data exchange between devices. The I^{2}C standard is a true multi-master bus including collision detection and arbitration that prevents data corruption if two or more masters attempt to control the bus simultaneously.

There are four sets of I^{2}C controllers which support Power-down wake-up function.

6.16.2 Features

The I^{2}C bus uses two wires (SDA and SCL) to transfer information between devices connected to the bus. The main features of the I^{2}C bus include:

- Supports up to three I^{2}C 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 I^{2}C interrupt if the I^{2}C bus hangs up and timer-out counter overflows
- Programmable clocks allow for versatile rate control
- Supports 7-bit addressing and 10-bit addressing mode
- Supports multiple address recognition (four slave address with mask option)
- Supports Power-down wake-up function
- Supports setup/hold time programmable

6.16.3 Block Diagram

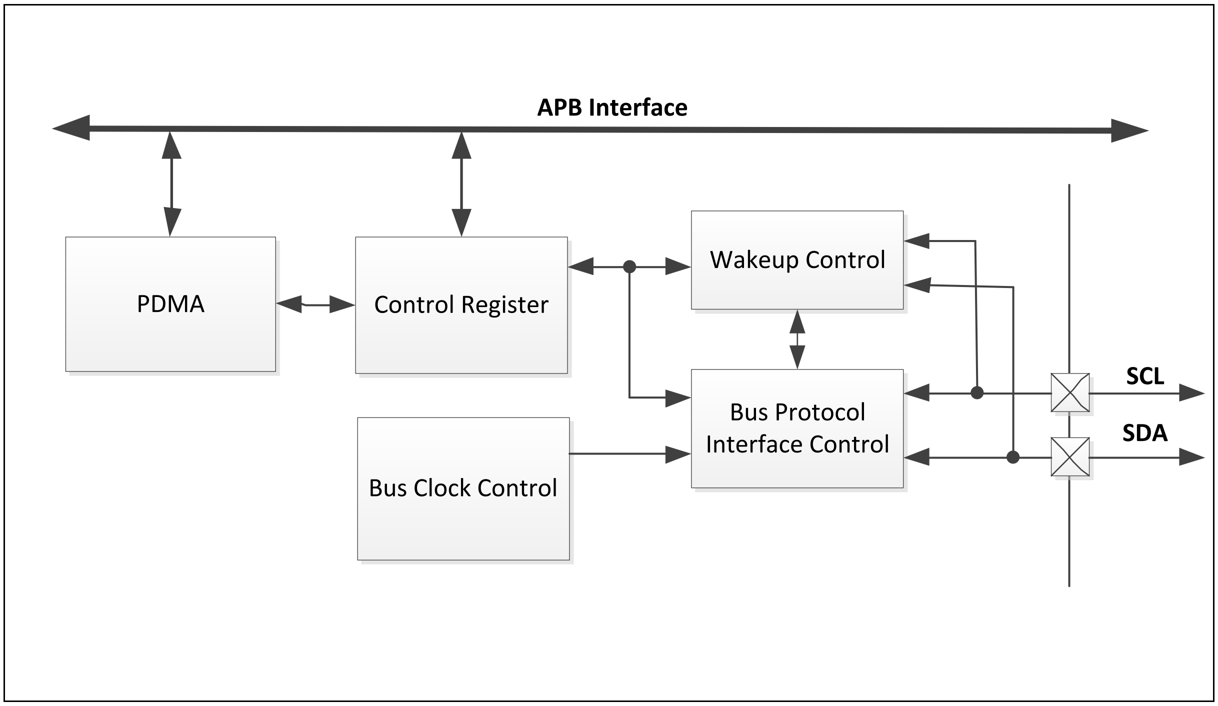


Figure 6.16-1 I^{2}C Controller Block Diagram

6.16.4 Basic Configuration

I^{2}C0 basic configurations

- Clock source Configuration
  - Enable I^{2}C0 peripheral clock in I^{2}C0CKEN (CLK_APBCLK1[0]).
- Reset Configuration
  - Reset I^{2}C0 controller in I^{2}C0RST (SYS_APBIPRST1[0]).
- Pin configuration

Group	Pin Name	GPIO	MFP
I^{2}C0	I^{2}C0_SCL	PG.10	MFP2
		PA.1	MFP3
		PE.12	MFP6
	I^{2}C0_SDA	PA.15	MFP2
		PA.0	MFP3
		PE.10	MFP6

I^{2}C1 Basic Configurations

- Clock Source Configuration
  - Enable I^{2}C1 peripheral clock in I^{2}C1CKEN (CLK_APBCLK1[1]).
- Reset Configuration
  - Reset I^{2}C1 controller in I^{2}C1RST (SYS_APBIPRST1[1]).

- Pin Configuration

Group	Pin Name	GPIO	MFP
I^{2}C1	I^{2}C1_SCL	PA.14, PB.4	MFP2
		PC.3	MFP4
	I^{2}C1_SDA	PA.13, PB.6	MFP2
		PC.4	MFP4

I^{2}C2 Basic Configurations

- Clock source Configuration
  - Enable I^{2}C2 peripheral clock in I^{2}C2CKEN (CLK_APBCLK1[2]).
- Reset Configuration
  - Reset I^{2}C2 controller in I^{2}C2RST (SYS_APBIPRST1 [2]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
I^{2}C2	I^{2}C2_SCL	PB.5, PB.8	MFP2
	I^{2}C2_SDA	PB.7, PC.0	MFP2

I^{2}C3 Basic Configurations

- Clock source Configuration
  - Enable I^{2}C3 peripheral clock in I^{2}C3CKEN (CLK_APBCLK1[3]).
- Reset Configuration
  - Reset I^{2}C3 controller in I^{2}C3RST (SYS_APBIPRST1[3]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
I^{2}C3	I^{2}C3_SCL	PB.3	MFP2
		PD.14	MFP3
	I^{2}C3_SDA	PB.1	MFP2
		PD.15	MFP3

6.16.5 Functional Description

On I^{2}C bus, data is transferred between a Master and a Slave. Data bits transfer on the SCL and SDA lines are synchronously on a byte-by-byte basis. Each data byte is 8-bit 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^{2}C BUS Timing.

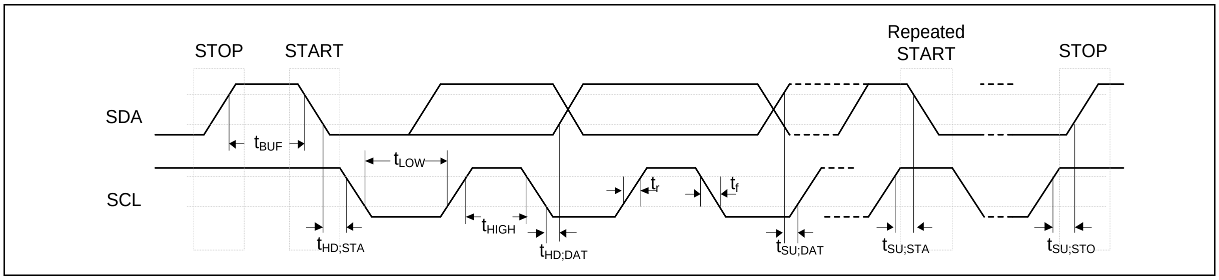


Figure 6.16-2 I^{2}C Bus Timing

The device’s on-chip I^{2}C provides the serial interface that meets the I^{2}C bus standard mode specification. The I^{2}C port handles byte transfers autonomously. To enable this port, the bit I^{2}CEN in I^{2}C_CTL0 should be set to '1'. The I^{2}C hardware interfaces to the I^{2}C bus via two pins: SDA and SCL. When I/O pins are used as I^{2}C ports, user must set the pins function to I^{2}C in advance.

Note: Pull-up resistor is needed for I^{2}C operation as the SDA and SCL are open-drain pins.

6.16.5.1 I^{2}C Protocol

Figure 6.16-3 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

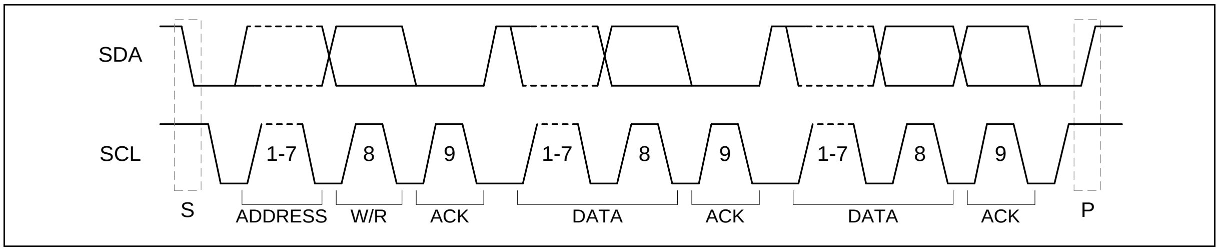


Figure 6.16-3 I^{2}C Protocol

- START or Repeated START signal

  When the bus is free/idle, which means no master device is engaging the bus (both SCL and SDA lines are high), a master can initiate a transfer by sending a START signal. A START signal, usually referred to as the “S” bit, is defined as a HIGH to LOW transition on the SDA line while SCL is HIGH. The START signal denotes the beginning of a new data transmission.

  After having sent the address byte (address and read/write bit), the master may send any number of bytes followed by a stop condition. Instead of sending the stop condition it is also allowed to send another start condition again followed by an address (and of course including a read/write bit) and more data. The start condition is called as Repeat START (Sr). This is defined recursively allowing any number of start conditions to be sent. The purpose of this is to allow combined write/read operations to one or more devices without releasing the bus and thus with the guarantee that the operation is not interrupted. The controller uses this method to communicate with another slave or the same slave in a different transfer direction (e.g. from writing to a device to reading from a device) without releasing the bus.

- STOP signal

  The master can terminate the communication by generating a STOP signal. A STOP signal, usually referred to as the “P” bit, is defined as a LOW to HIGH transition on the SDA line while SCL is HIGH.

  Figure 6.16-4 shows the waveform of START, Repeat START and STOP.

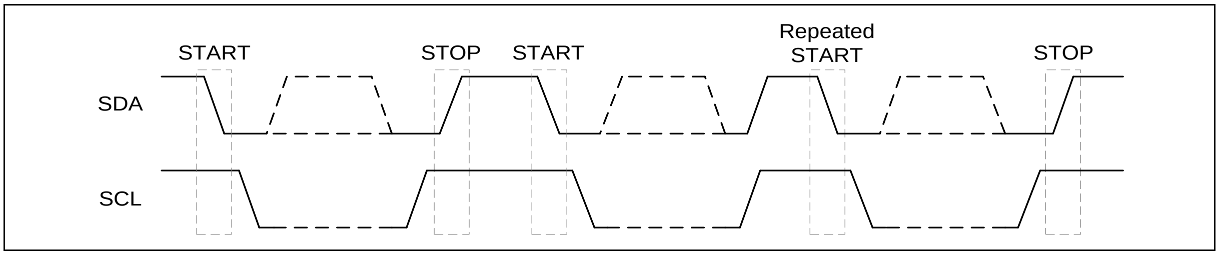


Figure 6.16-4 START and STOP Conditions

- Slave Address Transfer

  After a (Repeated) START condition, the master sends a slave address to identify the target device of the communication. The start address can comprise one or two address bytes (for 7-bit or for 10-bit addressing schemes). After an address byte, a slave sensitive to the transmitted address has to acknowledge the reception.

  Therefore, the slave’s address can be programmed in the device, where it is compared to the received address. In case of a match, the slave answers with an acknowledge (SDA = 0). Slaves that are not targeted answer with a non-acknowledge (SDA = 1). In addition to the match of the programmed address, another address byte value has to be answered with an acknowledge if the slave is capable to handle the corresponding requests.

- Data Transfer

  When a slave receives a correct address with an R/W bit, the data will follow R/W bit specified to transfer. Each transferred byte is followed by an acknowledge bit on the 9th SCL clock cycle. If the slave signals a Not Acknowledge (NACK), the master can generate a STOP signal to abort the data transfer or generate a Repeated START signal and start a new transfer cycle.

  If the master, as a receiving device, does Not Acknowledge (NACK) the slave, the slave releases the SDA line for the master to generate a STOP or Repeated START signal. The Figure 6.16-5 and Figure 6.16-6 shows the waveform of bit transfer and acknowledge.

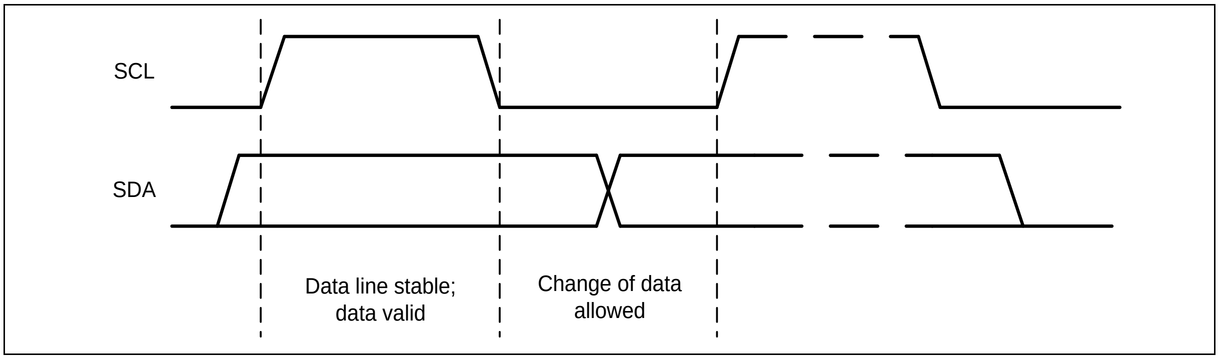


Figure 6.16-5 Bit Transfer on the I^{2}C Bus

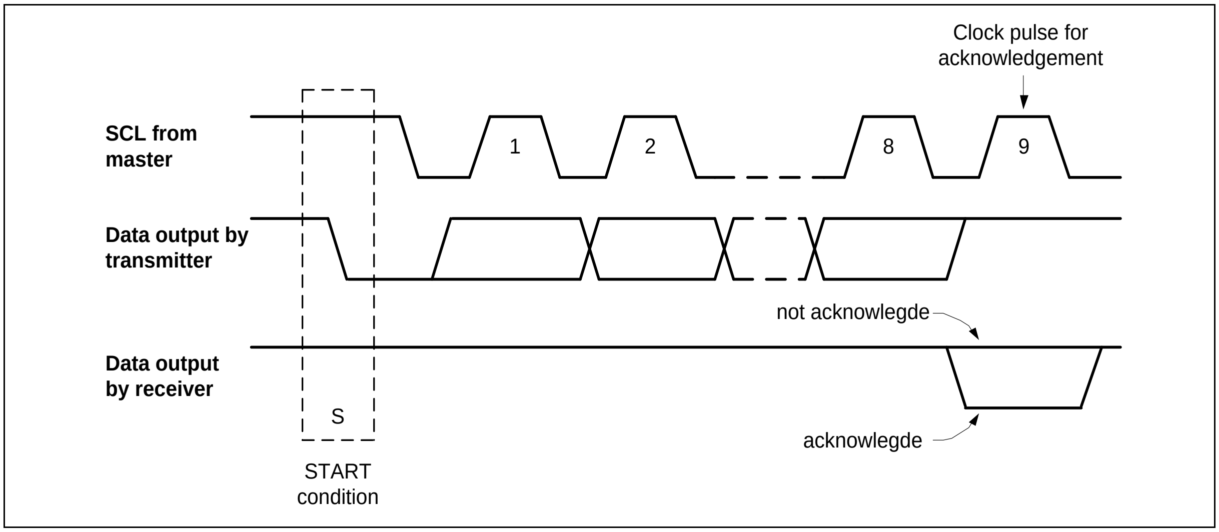


Figure 6.16-6 Acknowledge on the I^{2}C Bus

- Data transfer on I^{2}C bus

  Figure 6.16-7 shows a master transmits data to slave by 7-bit. A master addresses a slave with a 7-bit address and 1-bit write index to denote that the master wants to transmit data to the slave. The master keeps transmitting data after the slave returns acknowledge to the master.

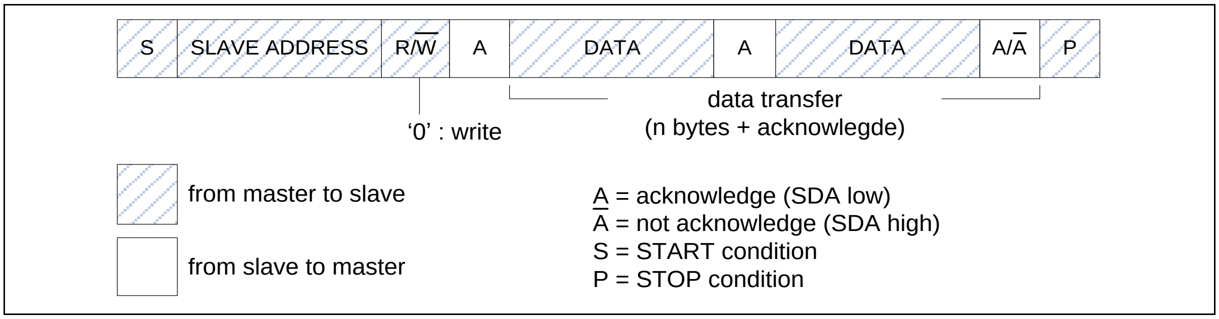


Figure 6.16-7 Master Transmits Data to Slave by 7-bit

Figure 6.16-8 shows a master read data from slave by 7-bit. A master addresses a slave with a 7-bit address and 1-bit read index to denote that the master wants to read data from the slave. The slave will start transmitting data after the slave returns acknowledge to the master.

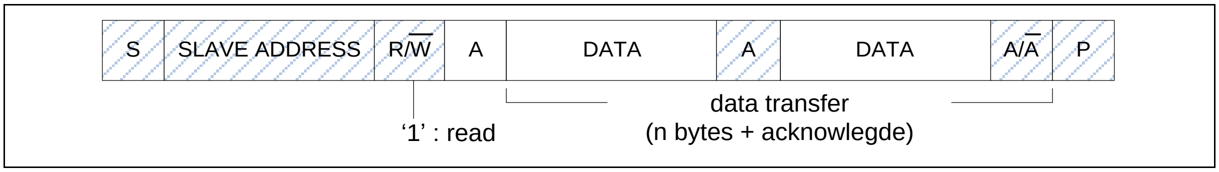


Figure 6.16-8 Master Reads Data from Slave by 7-bit

Figure 6.16-9 shows a master transmits data to slave by 10-bit. A master addresses a slave with a 10-bit address. First byte contains 10-bit address indicator (5'b11110) and 2-bit address with write index, second byte contains 8-bit address. The master keeps transmitting data after the second byte end. Note that 7-bit and 10-bit address device can work on the same bus.

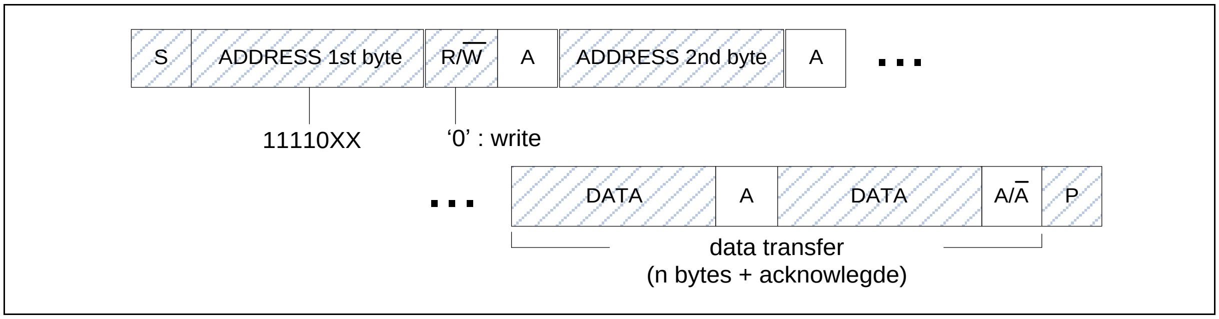


Figure 6.16-9 Master Transmits Data to Slave by 10-bit

Figure 6.16-10 shows a master read data from slave by 10-bit. A master addresses a slave with a 10-bit address. First mater transmits 10-bit address to slave, after that master transmits first byte with read index. The slave will start transmitting data after the first byte with read index.

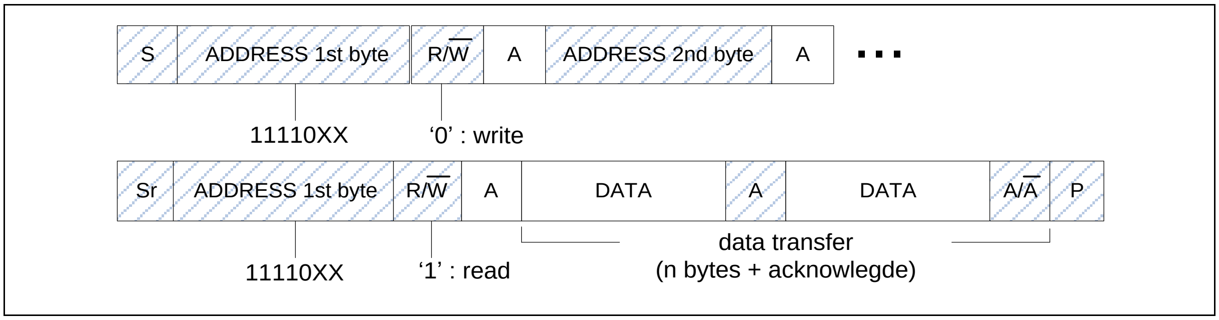


Figure 6.16-10 Master Reads Data from Slave by 10-bit

6.16.5.2 Operation Modes

The on-chip I^{2}C ports support three operation modes, Master, Slave, and General Call Mode.

In a given application, I^{2}C port may operate as a master or as a slave. In Slave mode, the I^{2}C port hardware looks for its own slave address and the general call address. If one of these addresses is detected, and if the slave is willing to receive or transmit data from/to master(by setting the AA bit), acknowledge pulse will be transmitted out on the 9th clock, hence an interrupt is requested on both master and slave devices if interrupt is enabled. When the microprocessor wishes to become the bus master, hardware waits until the bus is free before entering Master mode so that a possible slave action is not be interrupted. If bus arbitration is lost in Master mode, I^{2}C port switches to Slave mode immediately and can detect its own slave address in the same serial transfer.

To control the I^{2}C bus transfer in each mode, user needs to set I^{2}C_CTL0, I^{2}C_DAT registers according to current status code of I^{2}C_STATUS0 register. In other words, for each I^{2}C bus action, user needs to check current status by I^{2}C_STATUS0 register, and then set I^{2}C_CTL0, I^{2}C_DAT registers to take bus action. Finally, check the response status by I^{2}C_STATUS0.

The bits, STA, STO and AA in I^{2}C_CTL0 register are used to control the next state of the I^{2}C hardware after SI flag of I^{2}C_CTL0 [3] register is cleared. Upon completion of the new action, a new status code will be updated in I^{2}C_STATUS0 register and the SI flag of I^{2}C_CTL0 register will be set. But the SI flag will not be set when I^{2}C STOP. If the I^{2}C interrupt control bit INTEN (I^{2}C_CTL0 [7]) is set, appropriate action or software branch of the new status code can be performed in the Interrupt service routine.

Figure 6.16-11 shows the current I^{2}C status code is 0x08, and then set I^{2}C_DATA=SLA+W and (STA,STO,SI,AA) = (0,0,1,x) to send the address to I^{2}C bus. If a slave on the bus matches the address and response ACK, the I^{2}C_STATUS0 will be updated by status code 0x18.

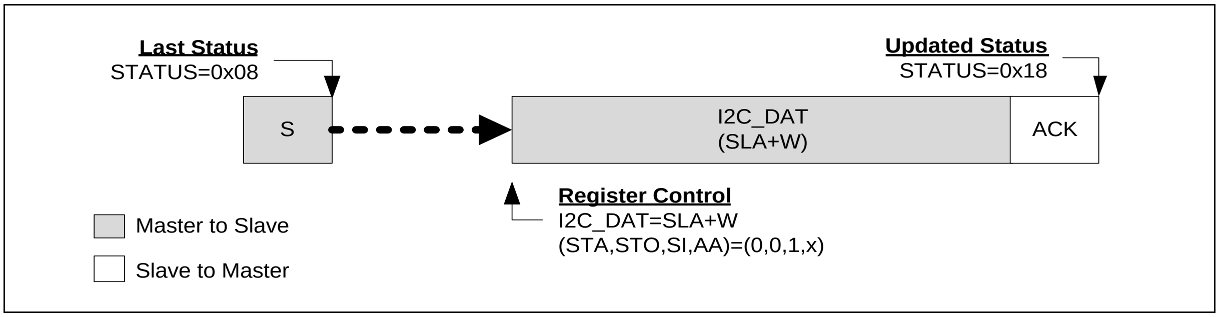


Figure 6.16-11 Control I^{2}C Bus according to the Current I^{2}C Status

Master Mode

In Figure 6.16-12 and Figure 6.16-13, all possible protocols for I^{2}C master are shown. User needs to follow proper path of the flow to implement required I^{2}C protocol.

In other words, user can send a START signal to bus and I^{2}C will be in Master Transmitter (MT) mode (Figure 6.16-12) or Master receiver (MR) mode (Figure 6.16-13) after START signal has been sent successfully and new status code would be 0x08. Followed by START signal, user can send slave address, read/write bit, data and Repeat START, STOP to perform I^{2}C protocol.

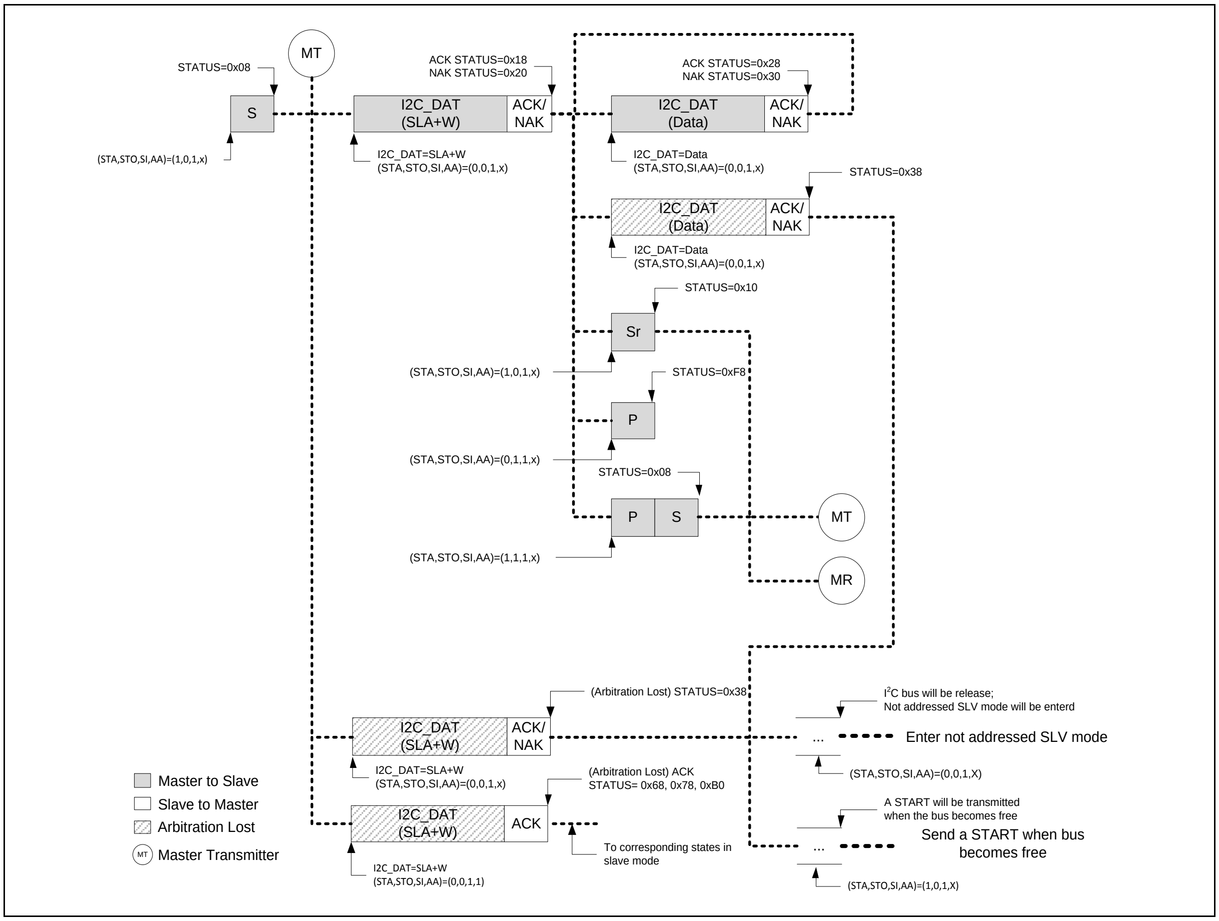


Figure 6.16-12 Master Transmitter Mode Control Flow

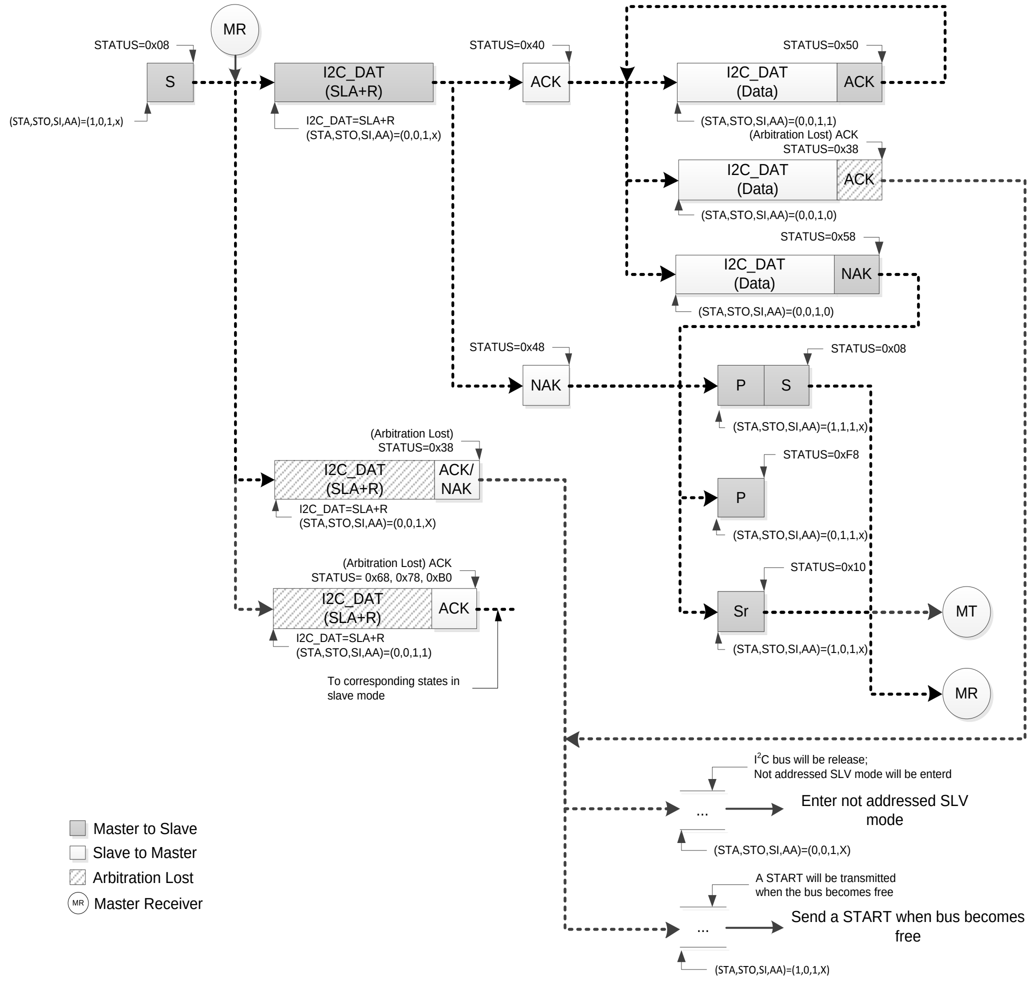

MR
STATUS=0x08
S
(STA,STO,SI,AA)=(1,0,1,x)
I2C_DAT
(SLA+R)
I2C_DAT=SLA+R
(STA,STO,SI,AA)=(0,0,1,x)
STATUS=0x40
ACK
STATUS=0x50
I2C_DAT
(Data)
ACK
(STA,STO,SI,AA)=(0,0,1,1)
(Arbitration Lost) ACK
STATUS=0x38
I2C_DAT
(Data)
ACK
(STA,STO,SI,AA)=(0,0,1,0)
STATUS=0x58
I2C_DAT
(Data)
NAK
(STA,STO,SI,AA)=(0,0,1,0)
STATUS=0x48
NAK
STATUS=0x08
P
S
(STA,STO,SI,AA)=(1,1,1,x)
STATUS=0xF8
P
(STA,STO,SI,AA)=(0,1,1,x)
STATUS=0x10
Sr
(STA,STO,SI,AA)=(1,0,1,x)
MT
MR
(Arbitration Lost)
STATUS=0x38
I2C_DAT
(SLA+R)
ACK/
NAK
I2C_DAT=SLA+R
(STA,STO,SI,AA)=(0,0,1,X)
(Arbitration Lost) ACK
STATUS= 0x68, 0x78, 0xB0
I2C_DAT
(SLA+R)
ACK
I2C_DAT=SLA+R
(STA,STO,SI,AA)=(0,0,1,1)
To corresponding states in slave mode
I²C bus will be release;
Not addressed SLV mode will be enterd
...
Enter not addressed SLV mode
(STA,STO,SI,AA)=(0,0,1,X)
A START will be transmitted
when the bus becomes free
...
Send a START when bus becomes free
(STA,STO,SI,AA)=(1,0,1,X)
■ Master to Slave
□ Slave to Master
▨ Arbitration Lost
MR Master Receiver

Figure 6.16-13 Master Receiver Mode Control Flow

If the I^{2}C is in Master mode and gets arbitration lost, the status code will be 0x38. In status 0x38, user may set (STA, STO, SI, AA) = (1, 0, 1, X) to send START to re-start Master operation when bus become free. Otherwise, user may set (STA, STO, SI, AA) = (0, 0, 1, X) to release I^{2}C bus and enter not addressed Slave mode.

Slave Mode

When reset default, I^{2}C is not addressed and will not recognize the address on I^{2}C bus. User can set slave address by I^{2}C_ADDRn (n=0~3) and set (STA, STO, SI, AA) = (0, 0, 1, 1) to let I^{2}C recognize the address sent by master. Figure 6.16-14 shows all the possible flow for I^{2}C in Slave mode. Users need to follow a proper flow (as shown in Figure 6.16-14 to implement their own I^{2}C protocol.

If bus arbitration is lost in Master mode, I^{2}C port switches to Slave mode immediately and can detect its own slave address in the same serial transfer. If the detected address is SLA+W (Master want to write data to Slave) after arbitration lost, the status code is 0x68. If the detected address is SLA+R (Master want to read data from Slave) after arbitration lost, the status code is 0xB0.

Note: During I^{2}C communication, the SCL clock will be released when writing ‘1’ to clear SI flag in

Slave mode.

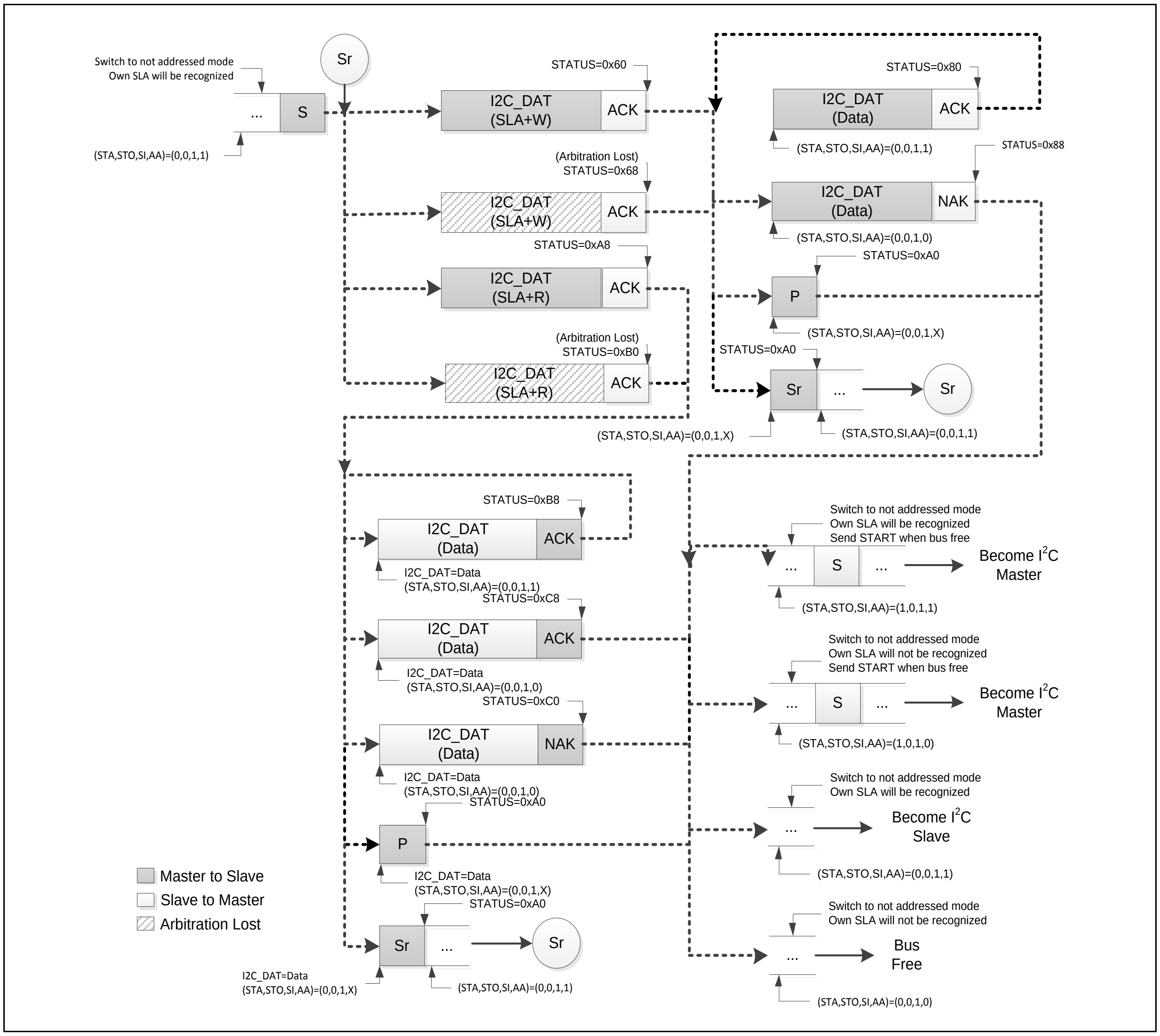

Switch to not addressed mode
Own SLA will be recognized
...
S
(STA,STO,SI,AA)=(0,0,1,1)
Sr
STATUS=0x60
I2C_DAT
(SLA+W)
ACK
(Arbitration Lost)
STATUS=0x68
I2C_DAT
(SLA+W)
ACK
STATUS=0xA8
I2C_DAT
(SLA+R)
ACK
(Arbitration Lost)
STATUS=0xB0
I2C_DAT
(SLA+R)
ACK
(STA,STO,SI,AA)=(0,0,1,X)
STATUS=0x80
I2C_DAT
(Data)
ACK
(STA,STO,SI,AA)=(0,0,1,1)
STATUS=0x88
I2C_DAT
(Data)
NAK
(STA,STO,SI,AA)=(0,0,1,0)
STATUS=0xA0
P
(STA,STO,SI,AA)=(0,0,1,X)
STATUS=0xA0
Sr
...
Sr
(STA,STO,SI,AA)=(0,0,1,1)
STATUS=0xB8
I2C_DAT
(Data)
ACK
I2C_DAT=Data
(STA,STO,SI,AA)=(0,0,1,1)
STATUS=0xC8
I2C_DAT
(Data)
ACK
I2C_DAT=Data
(STA,STO,SI,AA)=(0,0,1,0)
STATUS=0xC0
I2C_DAT
(Data)
NAK
I2C_DAT=Data
(STA,STO,SI,AA)=(0,0,1,0)
STATUS=0xA0
P
I2C_DAT=Data
(STA,STO,SI,AA)=(0,0,1,X)
STATUS=0xA0
Sr
...
Sr
I2C_DAT=Data
(STA,STO,SI,AA)=(0,0,1,X)
(STA,STO,SI,AA)=(0,0,1,1)
Switch to not addressed mode
Own SLA will be recognized
Send START when bus free
...
S
...
Become I2C Master
(STA,STO,SI,AA)=(1,0,1,1)
Switch to not addressed mode
Own SLA will not be recognized
Send START when bus free
...
S
...
Become I2C Master
(STA,STO,SI,AA)=(1,0,1,0)
Switch to not addressed mode
Own SLA will be recognized
...
Become I2C Slave
(STA,STO,SI,AA)=(0,0,1,1)
Switch to not addressed mode
Own SLA will not be recognized
...
Bus Free
(STA,STO,SI,AA)=(0,0,1,0)
Master to Slave
Slave to Master
Arbitration Lost

Figure 6.16-14 Slave Mode Control Flow

If I^{2}C is still receiving data in addressed Slave mode but got a STOP or Repeat START, the status code will be 0xA0. User could follow the action for status code 0x88 as shown in the above figure when getting 0xA0 status.

If I^{2}C is still transmitting data in addressed Slave mode but got a STOP or Repeat START, the status code will be 0xA0. User could follow the action for status code 0xC8 as shown in the above figure when getting 0xA0 status.

Note: After slave gets status of 0x88, 0xC8, 0xC0 and 0xA0, slave can switch to not address mode and own SLA will not be recognized. If entering this status, slave will not receive any I^{2}C signal or address from master. At this status, I^{2}C should enter idle mode.

General Call (GC) Mode

If the GC bit (I^{2}C_ADDRn [0]) is set, the I^{2}C port hardware will respond to General Call address (00H).

User can clear GC bit to disable general call function. When the GC bit is set and the I^{2}C in Slave mode, it can receive the general call address by 0x00 after master send general call address to I^{2}C bus, then it will follow status of GC mode.

The GC mode can wake up when address matched. Note that the default address is 0x00, but user must set an address except for 0x00.

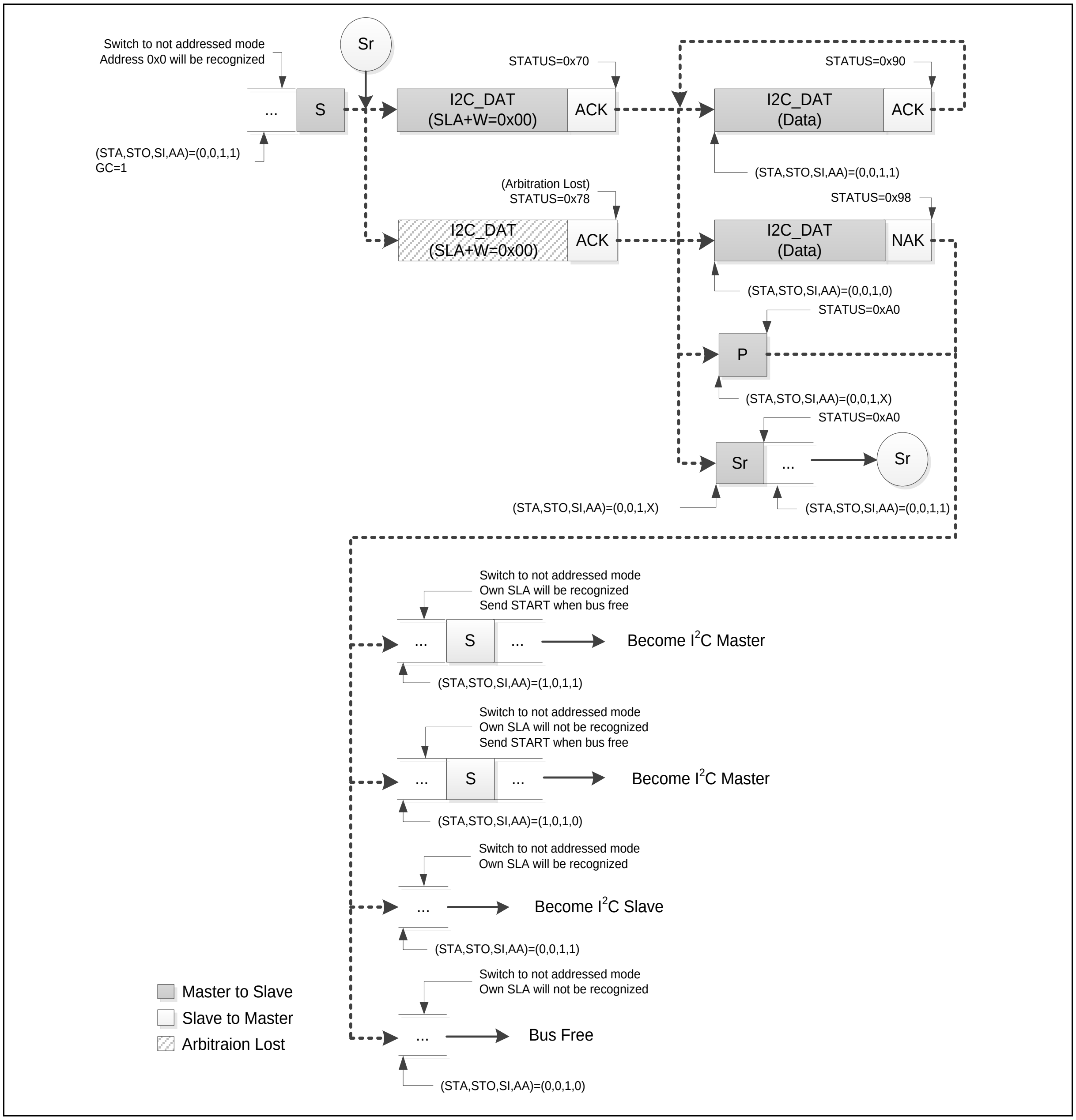

Switch to not addressed mode
Address 0x0 will be recognized
Sr
STATUS=0x70
STATUS=0x90
...
S
I2C_DAT
(SLA+W=0x00)
ACK
I2C_DAT
(Data)
ACK
(STA,STO,SI,AA)=(0,0,1,1)
GC=1
(STA,STO,SI,AA)=(0,0,1,1)
(Arbitration Lost)
STATUS=0x78
STATUS=0x98
I2C_DAT
(SLA+W=0x00)
ACK
I2C_DAT
(Data)
NAK
(STA,STO,SI,AA)=(0,0,1,0)
STATUS=0xA0
P
(STA,STO,SI,AA)=(0,0,1,X)
STATUS=0xA0
Sr
...
Sr
(STA,STO,SI,AA)=(0,0,1,X)
(STA,STO,SI,AA)=(0,0,1,1)
Switch to not addressed mode
Own SLA will be recognized
Send START when bus free
...
S
...
Become I2C Master
(STA,STO,SI,AA)=(1,0,1,1)
Switch to not addressed mode
Own SLA will not be recognized
Send START when bus free
...
S
...
Become I2C Master
(STA,STO,SI,AA)=(1,0,1,0)
Switch to not addressed mode
Own SLA will be recognized
...
Become I2C Slave
(STA,STO,SI,AA)=(0,0,1,1)
Switch to not addressed mode
Own SLA will not be recognized
...
Bus Free
(STA,STO,SI,AA)=(0,0,1,0)
Master to Slave
Slave to Master
Arbitraion Lost

Figure 6.16-15 GC Mode

If I^{2}C is still receiving data in GC mode but got a STOP or Repeat START, the status code will be 0xA0. User could follow the action for status code 0x98 in above figure when getting 0xA0 status.

Note: After slave gets status of 0x98 and 0xA0, slave can switch to not address mode and own SLA will not be recognized. If entering this status, slave will not receive any I^{2}C signal or address from master. At this time, the I^{2}C controller should enter idle mode.

Multi-Master

In some applications, there are two or more masters on the same I^{2}C bus to access slaves, and the masters may transmit data simultaneously. The I^{2}C supports multi-master by including collision detection and arbitration to prevent data corruption.

If for some reason two masters initiate command at the same time, the arbitration procedure determines which master wins and can continue with the command. Arbitration is performed on the SDA signal while the SCL signal is high. Each master checks if the SDA signal on the bus corresponds to the generated SDA signal. If the SDA signal on the bus is low but it should be high, then this master has lost arbitration. The device that has lost arbitration can generate SCL pulses until the byte ends and must then release the bus and go into slave mode. The arbitration procedure can continue until all the data is transferred. This means that in multi-master system each master must monitor the bus for collisions and act accordingly.

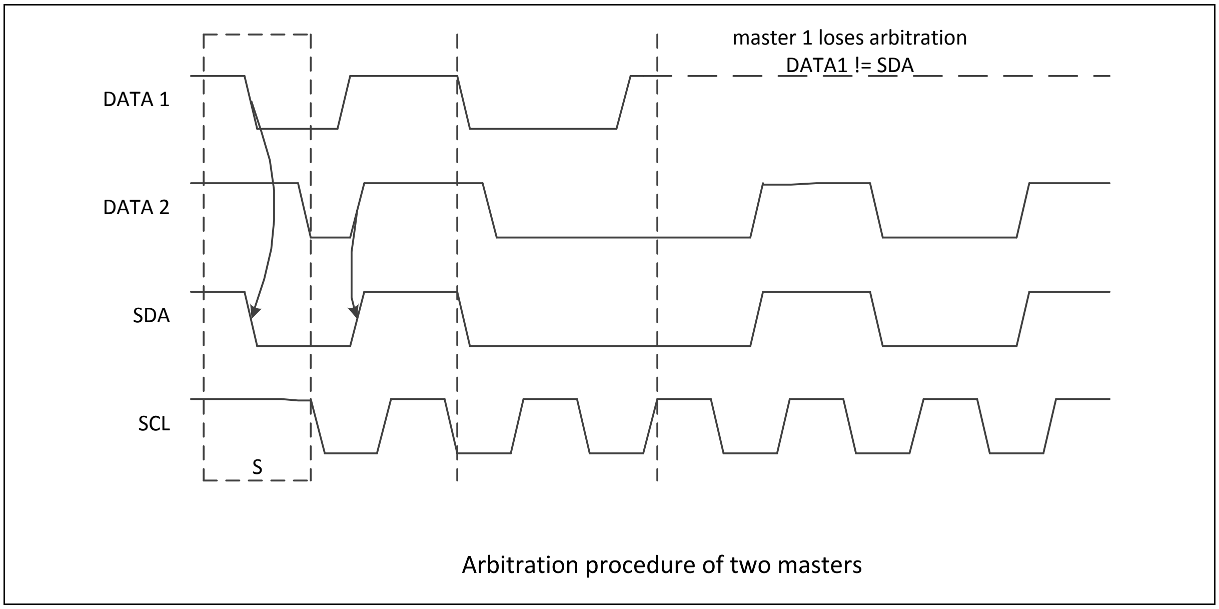

master 1 loses arbitration
DATA1 != SDA
DATA 1
DATA 2
SDA
SCL
S
Arbitration procedure of two masters

Figure 6.16-16 Arbitration Lost

- When I^{2}C_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 (I^{2}C_STATUS1 [8]).
- When I^{2}C_STATUS0 = 0x00, a “Bus Error” is received. To recover I^{2}C bus from a bus error, STO should be set and SI should be cleared, and then STO is cleared to release bus.

- Set (STA, STO, SI, AA) = (0, 1, 1, X) to stop current transfer
- Set (STA, STO, SI, AA) = (0, 0, 1, X) to release bus

6.16.5.3 Programmable setup and hold times

To guarantee a correct data setup and hold time, the timing must be configured. By programming HTCTL (I^{2}C_TMCTL[24:16]) to configure hold time and STCTL (I^{2}C_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 6.16-1 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**	**1000k**
12 MHz	120	60	30	15	12
24 MHz	240	120	60	30	24
48 MHz	480	240	120	60	48
72 MHz	720	360	180	90	72

Table 6.16-1 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 (I^{2}C_TMCTL[8:0]) to 3 that stretch three PCLKs for setup time setting. The setup time maximum setting value: ST_{limit} = (I^{2}C_CLKDIV[7:0]+1) X 2 - 6.

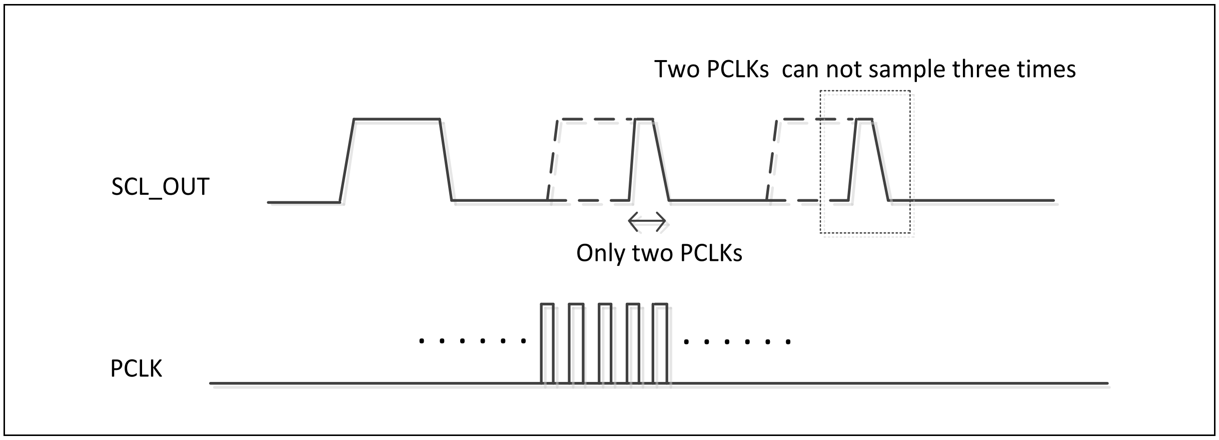


Figure 6.16-17 Setup Time Wrong Adjustment

For hold time wrong adjustment example, use I^{2}C Baud Rate = 1200k and PCLK = 72 MHz, the SCL high/low duty = 60 PCLK. When HTCTL (I^{2}C_TMCTL[24:16]) is set to 61 and STCTL (I^{2}C_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} = (I^{2}C_CLKDIV[7:0]+1) X 2 - 9.

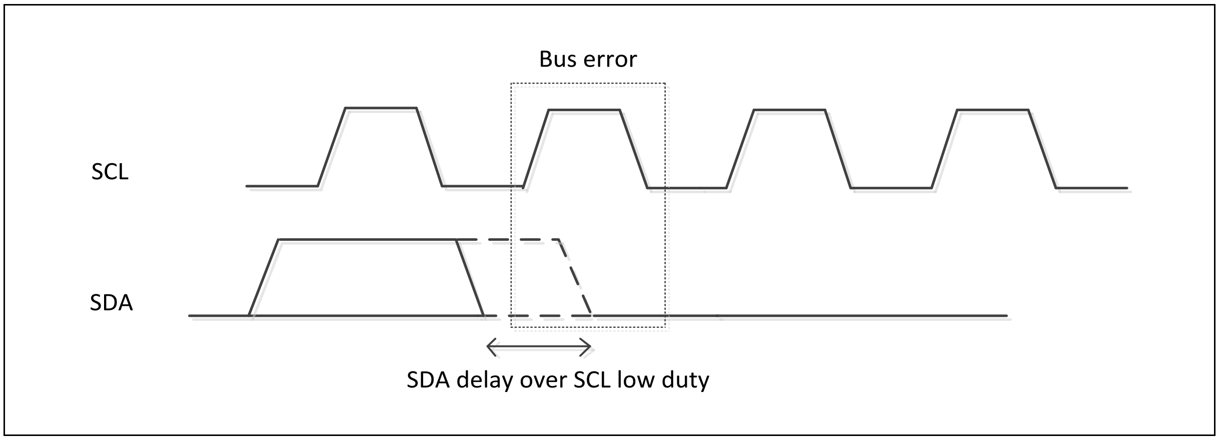


Figure 6.16-18 Hold Time Wrong Adjustment

6.16.5.4 I^{2}C Protocol Registers

To control I^{2}C port through the following fifteen special function registers: I^{2}C_CTL0 (control register), I^{2}C_STATUS0 (status register), I^{2}C_DAT (data register), I^{2}C_ADDRn (address registers, n=0~3), I^{2}C_ADDRMSKn (address mask registers, n=0~3), I^{2}C_CLKDIV (clock rate register), I^{2}C_TOCTL (Time-out control register), I^{2}C_WKCTL(wake up control register) and I^{2}C_WKSTS(wake up status register).

Address Registers (I^{2}C_ADDR)

The I^{2}C port is equipped with four slave address registers, I^{2}C_ADDRn (n=0~3). The contents of the register are irrelevant when I^{2}C is in Master mode. In Slave mode, the bit field ADDR(I^{2}C_ADDRn[7:1]) must be loaded with the chip's own slave address. The I^{2}C hardware will react if the contents of I^{2}C_ADDRn are matched with the received slave address.

The I^{2}C ports support the "General Call" function. If the GC bit (I^{2}C_ADDRn [0]) is set the I^{2}C port hardware will respond to General Call address (00H). Clear GC bit to disable general call function.

When the GC bit is set and the I^{2}C is in Slave mode, it can receive the general call address by 00H after Master send general call address to I^{2}C bus, then it will follow status of GC mode.

Slave Address Mask Registers (I^{2}C_ADDRMSK)

The I^{2}C bus controller supports multiple address recognition with four address mask registers I^{2}C_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 (I^{2}C_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 (I^{2}C_DAT [7:0]) directly while it is not in the process of shifting a byte. When I^{2}C is in a defined state and the serial interrupt flag (SI) is set, data in I^{2}C_DAT [7:0] remains stable. While data is being shifted out, data on the bus is simultaneously being shifted in; I^{2}C_DAT [7:0] always contains the last data byte presented on the bus.

The acknowledge bit is controlled by the I^{2}C hardware and cannot be accessed by the CPU. Serial data is shifted into I^{2}C_DAT [7:0] on the rising edges of serial clock pulses on the SCL line. When a byte has been shifted into I^{2}C_DAT [7:0], the serial data is available in I^{2}C_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 I^{2}C_DAT[7:0] when sending I^{2}C_DAT[7:0] to bus. In the case of sending data, serial data bits are shifted out from I^{2}C_DAT [7:0] on the falling edge of SCL clocks, and is shifted to I^{2}C_DAT [7:0] on the rising edge of SCL clocks. Figure 6.16-19 shows I^{2}C Data Shifting Direction.

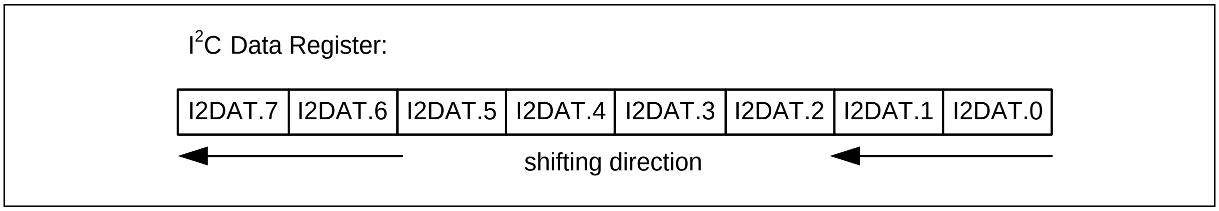


Figure 6.16-19 I^{2}C Data Shifting Direction

Control Register (I^{2}C_CTL0)

The CPU can be read from and written to I^{2}C_CTL0 [7:0] directly. When the I^{2}C port is enabled by setting I^{2}CEN (I^{2}C_CTL0 [6]) to high, the internal states will be controlled by I^{2}C_CTL0 and I^{2}C logic hardware.

There are two bits are affected by hardware: the SI bit is set when the I^{2}C hardware requests a serial interrupt, and the STO bit is cleared when a STOP condition is present on the bus. The STO bit is also cleared when I^{2}CEN = 0.

Once a new status code is generated and stored in I^{2}C_STATUS0, the I^{2}C Interrupt Flag bit SI (I^{2}C_CTL0 [3]) will be set automatically. If the Enable Interrupt bit INTEN (I^{2}C_CTL0 [7]) is set at this time, the I^{2}C interrupt will be generated. The bit field I^{2}C_STATUS0[7:0] stores the internal state code, the content keeps stable until SI is cleared by software.

Status Register (I^{2}C_STATUS0)

I^{2}C_STATUS0 [7:0] is an 8-bit read-only register. The bit field I^{2}C_STATUS0 [7:0] contains the status code and there 26 possible status codes. All states are listed in Table 6.16-2. When I^{2}C_STATUS0 [7:0] is F8H, no serial interrupt is requested. All other I^{2}C_STATUS0 [7:0] values correspond to the defined I^{2}C states. When each of these states is entered, a status interrupt is requested (SI = 1). A valid status code is present in I^{2}C_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 I^{2}C format frame. A Bus Error may occur during the serial transfer of an address byte, a data byte or an acknowledge bit. To recover I^{2}C from bus error, STO should be set and SI should be cleared to enter Not Addressed Slave mode. Then STO is cleared to release bus and to wait for a new communication. The I^{2}C bus cannot recognize stop condition during this action when a bus error occurs.

Master Mode		**Slave Mode**	
STATUS	Description	STATUS	Description
0x08[1]	Start	0xA0	Slave Transmit Repeat Start or Stop
0x10[1]	Master Repeat Start	0xA8[1]	Slave Transmit Address ACK
0x18[1]	Master Transmit Address ACK	0xB8[1]	Slave Transmit Data ACK
0x20	Master Transmit Address NACK	0xC0	Slave Transmit Data NACK
0x28[1]	Master Transmit Data ACK	0xC8[1]	Slave Transmit Last Data ACK
0x30	Master Transmit Data NACK	0x60[1]	Slave Receive Address ACK
0x38	Master Arbitration Lost	0x68[1]	Slave Receive Arbitration Lost
0x40[1]	Master Receive Address ACK	0x80[1]	Slave Receive Data ACK

0x48	Master Receive Address NACK	0x88	Slave Receive Data NACK
0x50[1]	Master Receive Data ACK	0x70[1]	GC mode Address ACK
0x58	Master Receive Data NACK	0x78[1]	GC mode Arbitration Lost
0x00	Bus error	0x90[1]	GC mode Data ACK
		0x98	GC mode Data NACK
		0xB0[1]	Address Transmit Arbitration Lost
0xF8	Bus Released **Note:** Status “0xF8” exists in both master/slave modes, and it won’t raise interrupt.		

Table 6.16-2 I^{2}C Status Code Description

Clock Baud Rate Bits (I^{2}C_CLKDIV)

The data baud rate of I^{2}C is determines by DIVIDER(I^{2}C_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 (I^{2}C_CLKDIV [7:0] +1)). If system clock = 16 MHz, the I^{2}C_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 (I^{2}C_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 I^{2}C_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 6.16-20 for the 14-bit time-out counter. User may write 1 to clear TOIF to 0.

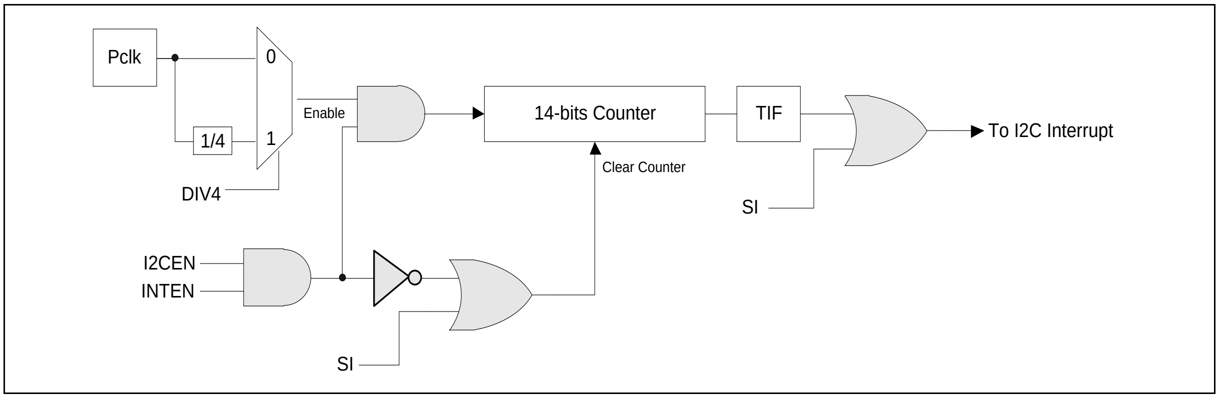


Figure 6.16-20 I^{2}C Time-out Count Block Diagram

Wake-up Control Register (I^{2}C_WKCTL)

When chip enters Power-down mode and set WKEN (I^{2}C_WKCTL [0]) to 1, other I^{2}C master can wake up the chip by addressing the I^{2}C device, user must configure the related setting before entering sleep mode. The ACK bit cycle of address match frame is done in power-down. The controller will stretch the SCL to low when the address is matched the device’s address and the ACK cycle done, then the I^{2}C controller will go ahead. If NHDBUSEN (I^{2}C_WKCTL [7]) is set, the controller will not stretch the

SCL to low. Note that when the controller does not stretch the SCL to low, transmit or receive data will be performed immediately. If data transmitted or received when SI event is not clear, user must reset the I^{2}C controller and execute the original operation again.

<u>Wake-up Status Register (I^{2}C_WKSTS)</u>

When system is woken up by other I^{2}C master device, WKIF is set to indicate this event. User needs write “1” to clear this bit.

When the chip is woken up by address match with one of the device address register (I^{2}C_ADDRn), the user shall check the WKAKDONE (I^{2}C_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 (I^{2}C_WKSTS [2]) bit records the Read/Write command before the I^{2}C controller sends address. The user can read this bit’s status to prepare the next transmitted data (WRSTSWK = 0) or to wait the incoming data (WRSTSWK = 1) can be stored in time after the system is wake-up by the address match frame. Note that the WRSTSWK (I^{2}C_WKSTS [2]) bit is cleared when writing one to the WKAKDONE (I^{2}C_WKSTS [1]) bit.

When system is woken up by other I^{2}C master device, WKIF is set to indicate this event. User needs to write “1” to clear this bit.

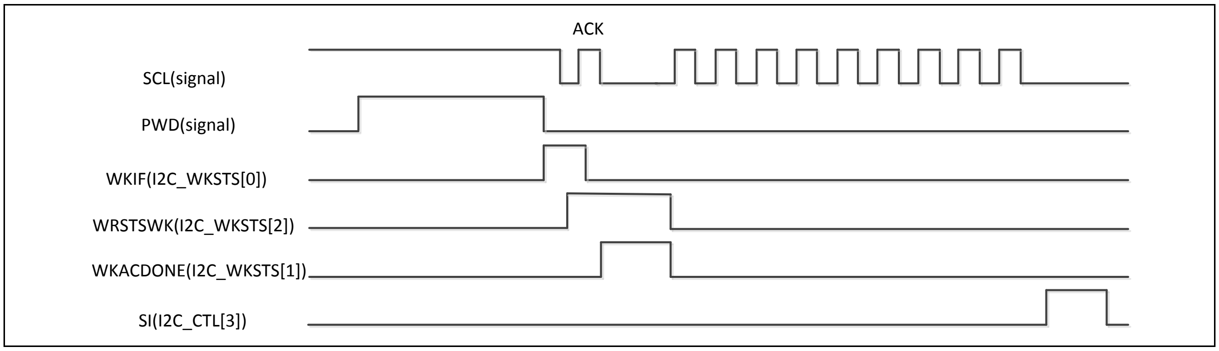


Figure 6.16-21 I^{2}C Wake-Up Related Signals Waveform

6.16.5.5 Example for Random Read on EEPROM

The following steps are used to configure the I^{2}C0 related registers when using I^{2}C to read data from EEPROM.

1. Set I^{2}C0 the multi-function pin as SCL and SDA pins. The muti-function configuration reference Basic Configuration.
2. Enable I^{2}C0 APB clock. The clock configuration reference Basic Configuration.
3. Set I^{2}C0RST=1 to reset I^{2}C0 controller then set I^{2}C0 controller to normal operation. The reset controller configuration reference Basic Configuration.
4. Set I^{2}CEN=1 to enable I^{2}C0 controller in the “I^{2}C_CTL0” register.
5. Give I^{2}C0 clock a divided register value for I^{2}C clock rate in the “I^{2}C_CLKDIV”.
6. Enable system I^{2}C0 IRQ in system “NVIC” control register.

7. Set INTEN=1 to enable I^{2}C0 Interrupt in the "I^{2}C_CTL0" register.
8. Set I^{2}C0 address registers "I^{2}C_ADDR0 ~ I^{2}C_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 6.16-22 shows the EEPROM random read operation.

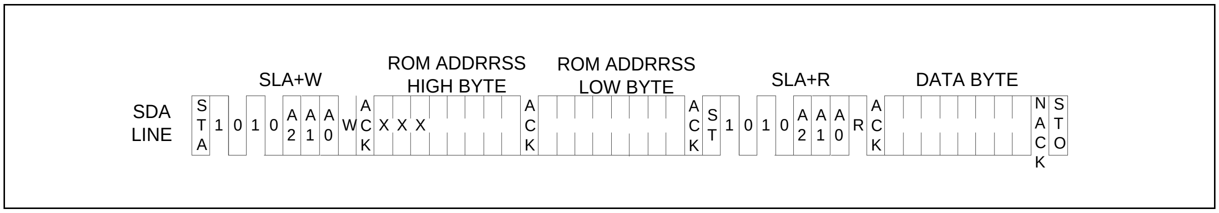


Figure 6.16-22 EEPROM Random Read

Figure 6.16-23 shows how to use the I^{2}C controller to implement the protocol of EEPROM random read.

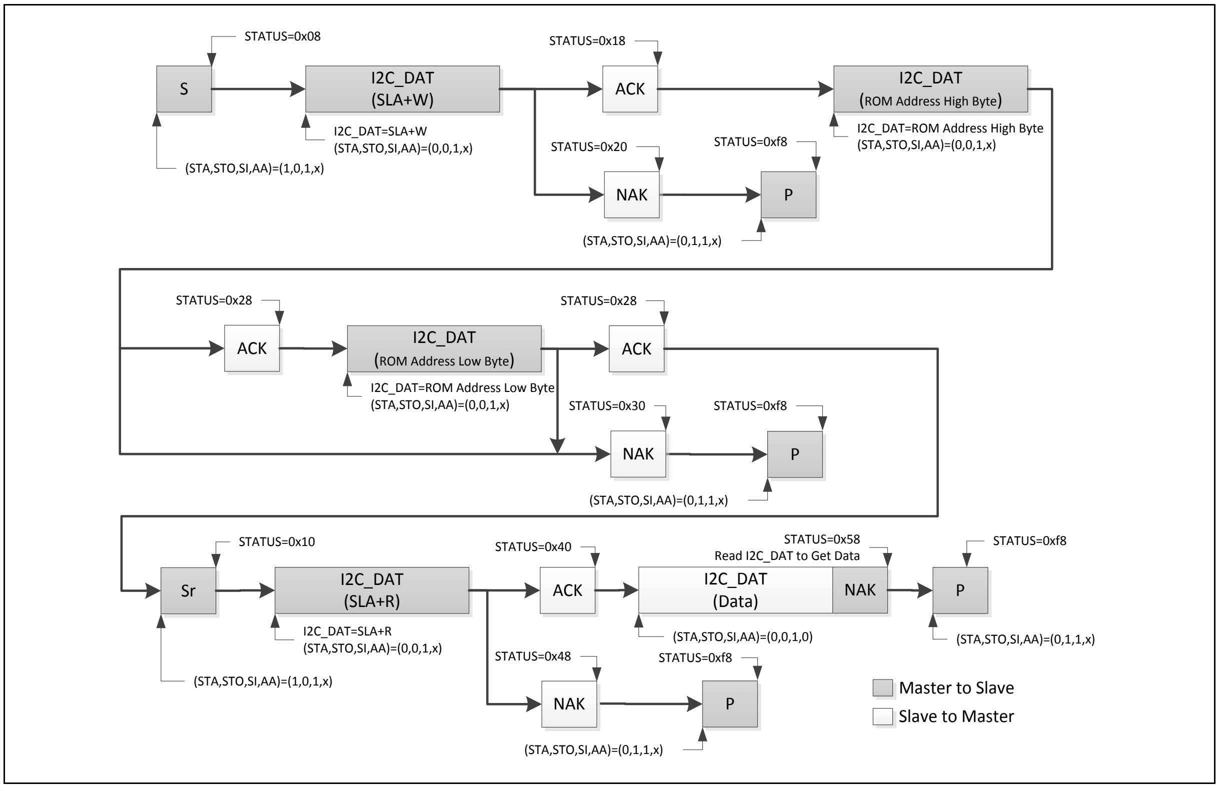

STATUS=0x08
S
I2C_DAT (SLA+W)
I2C_DAT=SLA+W (STA,STO,SI,AA)=(0,0,1,x)
(STA,STO,SI,AA)=(1,0,1,x)
STATUS=0x18
ACK
I2C_DAT (ROM Address High Byte)
I2C_DAT=ROM Address High Byte (STA,STO,SI,AA)=(0,0,1,x)
STATUS=0x20
NAK
STATUS=0xf8
P
(STA,STO,SI,AA)=(0,1,1,x)
STATUS=0x28
ACK
I2C_DAT (ROM Address Low Byte)
I2C_DAT=ROM Address Low Byte (STA,STO,SI,AA)=(0,0,1,x)
STATUS=0x28
ACK
STATUS=0x30
NAK
STATUS=0xf8
P
(STA,STO,SI,AA)=(0,1,1,x)
STATUS=0x10
Sr
I2C_DAT (SLA+R)
I2C_DAT=SLA+R (STA,STO,SI,AA)=(0,0,1,x)
(STA,STO,SI,AA)=(1,0,1,x)
STATUS=0x40
ACK
STATUS=0x58 Read I2C_DAT to Get Data
I2C_DAT (Data)
NAK
(STA,STO,SI,AA)=(0,0,1,0)
STATUS=0xf8
P
(STA,STO,SI,AA)=(0,1,1,x)
STATUS=0x48
NAK
STATUS=0xf8
P
(STA,STO,SI,AA)=(0,1,1,x)
Master to Slave
Slave to Master

Figure 6.16-23 Protocol of EEPROM Random Read

The I^{2}C controller, which is a master, sends START to bus. Then, it sends a SLA+W (Slave address + Write bit) to EERPOM followed by two bytes data address to set the EEPROM address to read. Finally, a Repeat START followed by SLA+R is sent to read the data from EEPROM.

6.16.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
I^{2}C Base Address: **I2Cn_BA = 0xB008_0000 + (0x1000 *n)** **n= 0,1,2,3**				
I2C_CTL0	I2Cn_BA+0x00	R/W	I^{2}C Control Register 0	0x0000_0000
I2C_ADDR0	I2Cn_BA+0x04	R/W	I^{2}C Slave Address Register0	0x0000_0000
I2C_DAT	I2Cn_BA+0x08	R/W	I^{2}C Data Register	0x0000_0000
I2C_STATUS0	I2Cn_BA+0x0C	R	I^{2}C Status Register 0	0x0000_00F8
I2C_CLKDIV	I2Cn_BA+0x10	R/W	I^{2}C Clock Divided Register	0x0000_0000
I2C_TOCTL	I2Cn_BA+0x14	R/W	I^{2}C Time-out Control Register	0x0000_0000
I2C_ADDR1	I2Cn_BA+0x18	R/W	I^{2}C Slave Address Register1	0x0000_0000
I2C_ADDR2	I2Cn_BA+0x1C	R/W	I^{2}C Slave Address Register2	0x0000_0000
I2C_ADDR3	I2Cn_BA+0x20	R/W	I^{2}C Slave Address Register3	0x0000_0000
I2C_ADDRMSK0	I2Cn_BA+0x24	R/W	I^{2}C Slave Address Mask Register0	0x0000_0000
I2C_ADDRMSK1	I2Cn_BA+0x28	R/W	I^{2}C Slave Address Mask Register1	0x0000_0000
I2C_ADDRMSK2	I2Cn_BA+0x2C	R/W	I^{2}C Slave Address Mask Register2	0x0000_0000
I2C_ADDRMSK3	I2Cn_BA+0x30	R/W	I^{2}C Slave Address Mask Register3	0x0000_0000
I2C_WKCTL	I2Cn_BA+0x3C	R/W	I^{2}C Wake-up Control Register	0x0000_0000
I2C_WKSTS	I2Cn_BA+0x40	R/W	I^{2}C Wake-up Status Register	0x0000_0000
I2C_CTL1	I2Cn_BA+0x44	R/W	I^{2}C Control Register 1	0x0000_0000
I2C_STATUS1	I2Cn_BA+0x48	R/W	I^{2}C Status Register 1	0x0000_0000
I2C_TMCTL	I2Cn_BA+0x4C	R/W	I^{2}C Timing Configure Control Register	0x0000_0000

6.16.7 Register Description

I^{2}C Control Register (I2C_CTL0)

Register	Offset	R/W	Description	Reset Value
I2C_CTL0	I2Cn_BA+0x00	R/W	I^{2}C Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
INTEN	**I2CEN**	**STA**	**STO**	**SI**	**AA**	**Reserved**	

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**INTEN**	**Enable Interrupt** 0 = I^{2}C interrupt Disabled. 1 = I^{2}C interrupt Enabled.
[6]	**I2CEN**	**I^{2}C Controller Enable Bit** Set to enable I^{2}C serial function controller. When I2CEN=1 the I^{2}C serial function enable. The multi-function pin function must set to SDA, and SCL of I^{2}C function first. 0 = I^{2}C controller Disabled. 1 = I^{2}C controller Enabled.
[5]	**STA**	**I^{2}C START Control** Setting STA to logic 1 to enter Master mode, the I^{2}C hardware sends a START or repeat START condition to bus when the bus is free.
[4]	**STO**	**I^{2}C STOP Control** In Master mode, setting STO to transmit a STOP condition to bus then I^{2}C controller will check the bus condition if a STOP condition is detected. This bit will be cleared by hardware automatically.
[3]	**SI**	**I^{2}C Interrupt Flag** When a new I^{2}C state is present in the I2C_STATUS0 register, the SI flag is set by hardware. If bit INTEN (I2C_CTL0 [7]) is set, the I^{2}C interrupt is requested. SI must be cleared by software. Clear SI by writing 1 to this bit.
[2]	**AA**	**Assert Acknowledge Control** When AA =1 prior to address or data is received, an acknowledged (low level to SDA) will be returned during the acknowledge clock pulse on the SCL line when 1.) A slave is acknowledging the address sent from master, 2.) The receiver devices are acknowledging the data sent by transmitter. When AA=0 prior to address or data received, a Not acknowledged (high level to SDA) will be returned during the acknowledge clock pulse on

		the SCL line.
[1:0]	**Reserved**	Reserved.

I^{2}C Data Register (I2C_DAT)

Register	Offset	R/W	Description	Reset Value
I2C_DAT	I2Cn_BA+0x08	R/W	I^{2}C Data Register	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**DAT**	**I^{2}C Data** Bit [7:0] is located with the 8-bit transferred/received data of I^{2}C serial port.

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.

I^{2}C Clock Divided Register (I2C_CLKDIV)

Register	Offset	R/W	Description	Reset Value
I2C_CLKDIV	I2Cn_BA+0x10	R/W	I^{2}C 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**	**I^{2}C Clock Divided** Indicates the I^{2}C clock rate: Data Baud Rate of I^{2}C = (system clock) / (4x (I2C_CLKDIV+1)). **Note:** The minimum value of I2C_CLKDIV is 4.

I^{2}C Time-out Control Register (I2C_TOCTL)

Register	Offset	R/W	Description	Reset Value
I2C_TOCTL	I2Cn_BA+0x14	R/W	I^{2}C Time-out Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**TOCEN**	**TOCDIV4**	**TOIF**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**TOCEN**	**Time-out Counter Enable Bit** When enabled, the 14-bit time-out counter will start counting when SI is 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 I^{2}C time-out happened and it can interrupt CPU if I^{2}C interrupt enable bit (INTEN) is set to 1. **Note:** Software can write 1 to clear this bit.

I^{2}C Slave Address Register (ADDRx)

Register	Offset	R/W	Description	Reset Value
I2C_ADDR0	I2Cn_BA+0x04	R/W	I^{2}C Slave Address Register0	0x0000_0000
I2C_ADDR1	I2Cn_BA+0x18	R/W	I^{2}C Slave Address Register1	0x0000_0000
I2C_ADDR2	I2Cn_BA+0x1C	R/W	I^{2}C Slave Address Register2	0x0000_0000
I2C_ADDR3	I2Cn_BA+0x20	R/W	I^{2}C Slave Address Register3	0x0000_0000

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

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10:1]	**ADDR**	**I^{2}C Address** The content of this register is irrelevant when I^{2}C is in Master mode. In the slave mode, the seven most significant bits must be loaded with the chip's own address. The I^{2}C hardware will react if either of the address is matched. **Note:** When software set 10'h000, the address can not be used.
[0]	**GC**	**General Call Function** 0 = General Call Function Disabled. 1 = General Call Function Enabled.

I^{2}C Slave Address Mask Register (ADDRMSKx)

Register	Offset	R/W	Description	Reset Value
I2C_ADDRMSK0	I2Cn_BA+0x24	R/W	I^{2}C Slave Address Mask Register0	0x0000_0000
I2C_ADDRMSK1	I2Cn_BA+0x28	R/W	I^{2}C Slave Address Mask Register1	0x0000_0000
I2C_ADDRMSK2	I2Cn_BA+0x2C	R/W	I^{2}C Slave Address Mask Register2	0x0000_0000
I2C_ADDRMSK3	I2Cn_BA+0x30	R/W	I^{2}C Slave Address Mask Register3	0x0000_0000

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

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10:1]	**ADDRMSK**	**I^{2}C Address Mask** 0 = Mask Disabled (the received corresponding register bit should be exact the same as address register.). 1 = Mask Enabled (the received corresponding address bit is don’t care.). I^{2}C bus controllers support multiple address recognition with four address mask register. When the bit in the address mask register is set to one, it means the received corresponding address bit is don’t-care. If the bit is set to zero, that means the received corresponding register bit should be exact the same as address register. **Note:** The wake-up function can not use address mask.
[0]	**Reserved**	Reserved.

I^{2}C Wake-up Control Register (I2C_WKCTL)

Register	Offset	R/W	Description	Reset Value
I2C_WKCTL	I2Cn_BA+0x3C	R/W	I^{2}C Wake-up Control Register	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**NHDBUSEN**	**I^{2}C No Hold BUS Enable Bit** 0 = I^{2}C hold bus after wake-up. 1= I^{2}C don't hold bus after wake-up. **Note:** The I^{2}C 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 I^{2}C controller and execute the original operation again.
[6:1]	**Reserved**	Reserved.
[0]	**WKEN**	**I^{2}C Wake-up Enable Bit** 0 = I^{2}C wake-up function Disabled. 1= I^{2}C wake-up function Enabled.

I^{2}C Wake-up Status Register (I2C_WKSTS)

Register	Offset	R/W	Description	Reset Value
I2C_WKSTS	I2Cn_BA+0x40	R/W	I^{2}C Wake-up Status Register	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**WRSTSWK**	**Read/Write Status Bit in Address Wakeup Frame** 0 = Write command be record on the address match wakeup frame. 1 = Read command be record on the address match wakeup frame. **Note:** This bit will be cleared when software can write 1 to WKAKDONE bit.
[1]	**WKAKDONE**	**Wakeup Address Frame Acknowledge Bit Done** 0 = The ACK bit cycle of address match frame isn't done. 1 = The ACK bit cycle of address match frame is done in power-down. **Note:** This bit can't release WKIF. Software can write 1 to clear this bit.
[0]	**WKIF**	**I^{2}C Wake-up Flag** When chip is woken up from Power-down mode by I^{2}C, this bit is set to 1. Software can write 1 to clear this bit.

I^{2}C Control Register 1 (I2C_CTL1)

Register	Offset	R/W	Description	Reset Value
I2C_CTL1	I2Cn_BA+0x44	R/W	I^{2}C Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**ADDR10EN**	**Reserved**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[9]	**ADDR10EN**	**Address 10-bit Function Enable Bit** 0 = Address match 10-bit function Disabled. 1 = Address match 10-bit function Enabled.
[8:0]	**Reserved**	Reserved.

I^2C Status Register 1 (I2C_STATUS1)

Register	Offset	R/W	Description	Reset Value
I2C_STATUS1	I2Cn_BA+0x48	R/W	I^2C Status Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**ONBUSY**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**ADMAT3**	**ADMAT2**	**ADMAT1**	**ADMAT0**

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**ONBUSY**	**On Bus Busy (Read Only)** Indicates that a communication is in progress on the bus. It is set by hardware when a START condition is detected. It is cleared by hardware when a STOP condition is detected. 0 = The bus is IDLE (both SCLK and SDA High). 1 = The bus is busy.
[7:4]	**Reserved**	Reserved.
[3]	**ADMAT3**	**I^2C Address 3 Match Status** When address 3 is matched, hardware will inform which address used. This bit will set to 1, and software can write 1 to clear this bit.
[2]	**ADMAT2**	**I^2C Address 2 Match Status** When address 2 is matched, hardware will inform which address used. This bit will set to 1, and software can write 1 to clear this bit.
[1]	**ADMAT1**	**I^2C Address 1 Match Status** When address 1 is matched, hardware will inform which address used. This bit will set to 1, and software can write 1 to clear this bit.
[0]	**ADMAT0**	**I^2C Address 0 Match Status** When address 0 is matched, hardware will inform which address used. This bit will set to 1, and software can write 1 to clear this bit.

I^{2}C Timing Configure Control Register (I2C_TMCTL)

Register	Offset	R/W	Description	Reset Value
I2C_TMCTL	I2Cn_BA+0x4C	R/W	I^{2}C Timing Configure Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							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.

6.17 Serial Peripheral Interface (SPI)

6.17.1 Overview

The Serial Peripheral Interface (SPI) applies to synchronous serial data communication and allows full duplex transfer. Devices communicate in Master/Slave mode with the 4-wire bi-direction interface. The chip contains up to one set of SPI controllers performing a serial-to-parallel conversion on data received from a peripheral device, and a parallel-to-serial conversion on data transmitted to a peripheral device. Each SPI controller can be configured as a master or a slave device and supports the PDMA function to access the data buffer.

6.17.2 Features

- SPI Mode
  - Up to two sets of SPI controllers
  - Supports Master or Slave mode operation
  - Master mode up to 100 MHz and Slave mode up to 30 MHz (when chip works at V_{DD} = 2.7~3.6V)
  - Configurable bit length of a transaction word from 8 to 32-bit
  - Provides separate 4-level depth transmit and receive FIFO buffers
  - Supports MSB first or LSB first transfer sequence
  - Supports Byte Reorder function
  - Supports Byte or Word Suspend mode
  - Supports PDMA transfer
  - Supports one data channel half-duplex transfer
  - Supports receive-only mode

6.17.3 Block Diagram

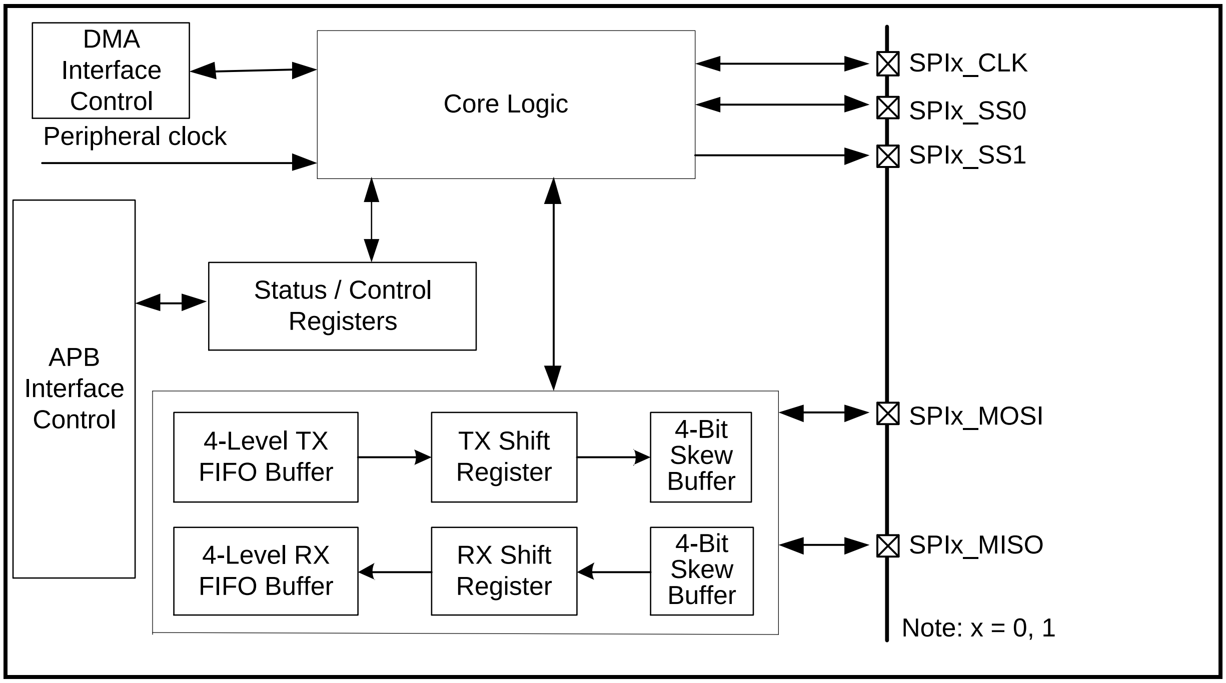


Figure 6.17-1 SPI Block Diagram

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 in transmitting and received side. In received side, it is used to shift bits into RX shift register from SPI bus. In transmitting side, it is used to shift bits into SPI bus from TX shift register.

6.17.4 Basic Configuration

6.17.4.1 *SPI0 Basic Configuration*

- Clock source Configuration
  - Select the source of SPI0 peripheral clock on SPI0_S (CLK_DIVCTL2[11:10]).
  - Enable SPI0 peripheral clock in SPI0CKEN (CLK_PCLKEN1[5]).
- Reset Configuration
  - Reset SPI0 controller in SPI0RST (SYS_APBIPRST1[5]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
SPI0	SPI0_CLK	PD.9	MFP1
		PC.6	MFP5
	SPI0_MISO	PD.11	MFP1
		PC.8	MFP5
	SPI0_MOSI	PD.10	MFP1
		PC.7	MFP5
		PC.4	MFP6
	SPI0_SS0	PD.8	MFP1
		PC.5	MFP5
	SPI0_SS1	PD.1, PG.15	MFP1
		PC.0	MFP5

6.17.4.2 *SPI1 Basic Configuration*

- Clock source Configuration
  - Select the source of SPI1 peripheral clock on SPI1_S (CLK_DIVCTL2[13:12]).
  - Enable SPI1 peripheral clock in SPI1CKEN (CLK_ PCLKEN1[6]).
- Reset Configuration
  - Reset SPI1 controller in SPI1RST (SYS_ APBIPRST1[6]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
SPI1	SPI1_CLK	PG.12	MFP2
		PB.10	MFP5
		PB.4, PG.10	MFP6
	SPI1_MISO	PG.14	MFP2
		PB.12	MFP5
		PB.5	MFP6
	SPI1_MOSI	PG.13	MFP2
		PB.11	MFP5

		PB.7	MFP6
	SPI1_SS0	PG.11	MFP2
		PB.9	MFP5
		PA.15, PB.6	MFP6
	SPI1_SS1	PG.15	MFP2
		PB.1	MFP6

SPI (SPI0 and SPI1) Interface Controller Pin description is shown as follows:

Pin	SPI Mode	
SPIx_SS0 / SPIx_SS1	SPI slave selection pin	
SPIx_CLK	SPI clock pin	
SPIx_MISO	SPI master input or slave output pin	
SPIx_MOSI	SPI master output or slave input pin	

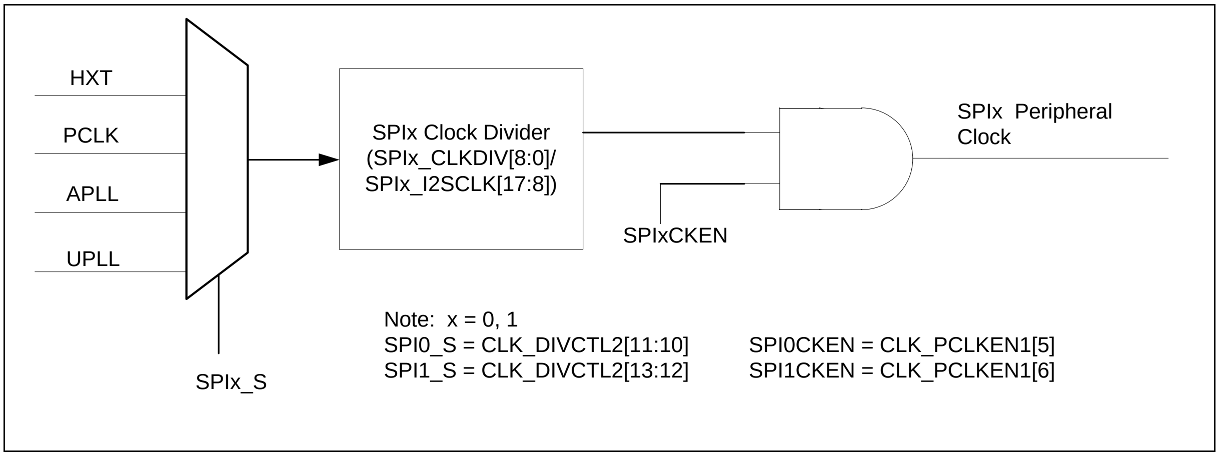
Table 6.17-1 SPI Interface Controller Pin Description (SPI0 and SPI1)

6.17.5 Functional Description

6.17.5.1 Terminology

<u>SPI Peripheral Clock and SPI Bus Clock</u>

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, PCLK, APLL, or UPLL. SPIx_S of CLK_DIVCTL2 register determines the clock source of the peripheral clock. The DIVIDER (SPIx_CLKDIV[8:0]) setting determines the divisor of the clock rate calculation.



Note: x = 0, 1
SPI0_S = CLK_DIVCTL2[11:10]
SPI1_S = CLK_DIVCTL2[13:12]
SPI0CKEN = CLK_PCLKEN1[5]
SPI1CKEN = CLK_PCLKEN1[6]

Figure 6.17-2 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.

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.

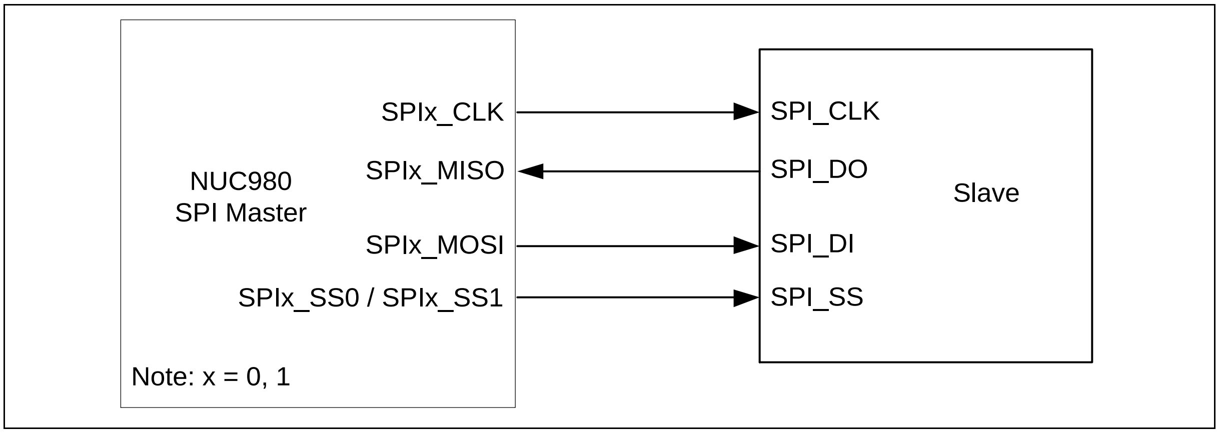


Figure 6.17-3 SPI Full-Duplex Master Mode Application Block Diagram

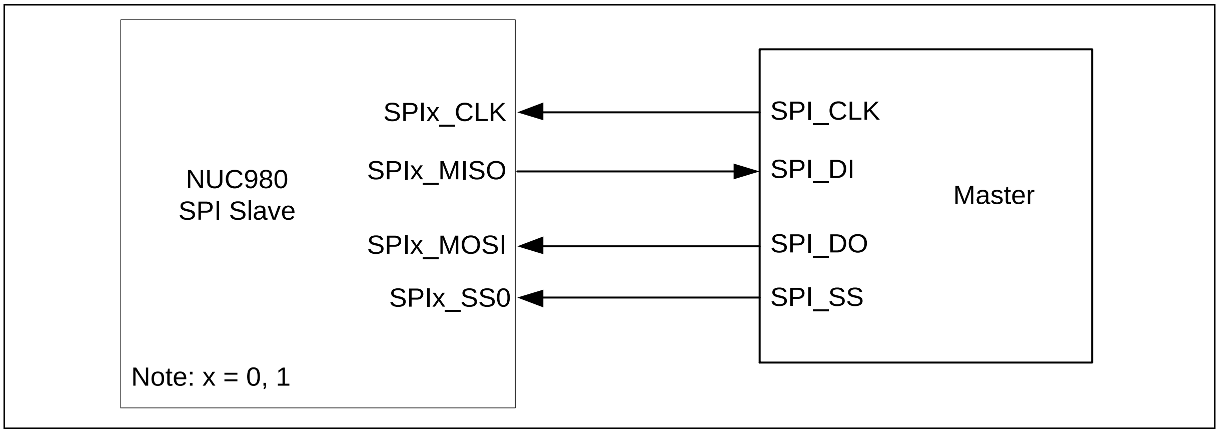


Figure 6.17-4 SPI Full-Duplex Slave Mode Application Block Diagram

Slave Selection

In Master mode, the SPI controller can drive off-chip slave device through the slave select output pin SPIx_SS0 or SPIx_SS1. In Slave mode, the off-chip master device drives the slave selection signal from the SPIx_SS0 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 will be set to 1.

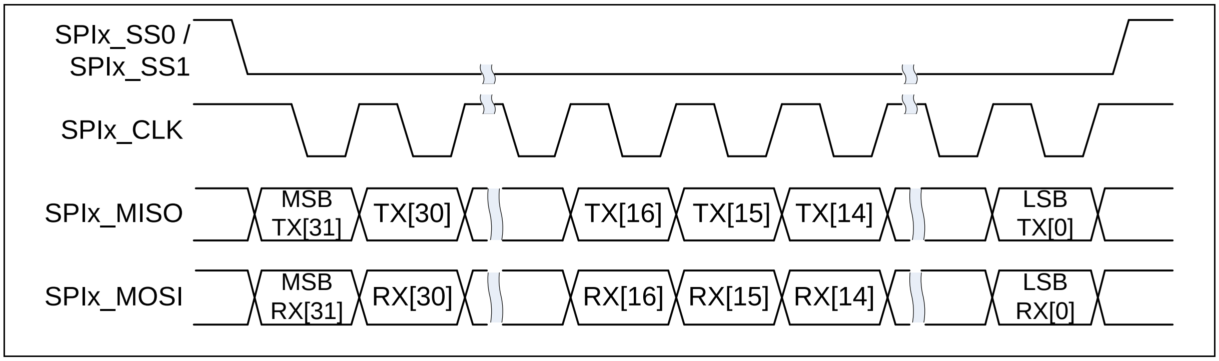


Figure 6.17-532-bit in One Transaction

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).

6.17.5.2 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_SS0 or SPIx_SS1 pin according to whether SS0 (SPIx_SSCTL[0]) or SS1 (SPIx_SSCTL[1]) is enabled or not. The slave selection signal will be set to active state by the SPI controller when the SPI data transfer is started by writing to FIFO. It will be set to inactive state when SPI bus is idle. If SPI bus is not idle, i.e. TX FIFO, TX shift register or TX skew buffer is not empty, the slave selection signal will be set to inactive state between transactions if the value of SUSPITV (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 SS0/SS1 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.

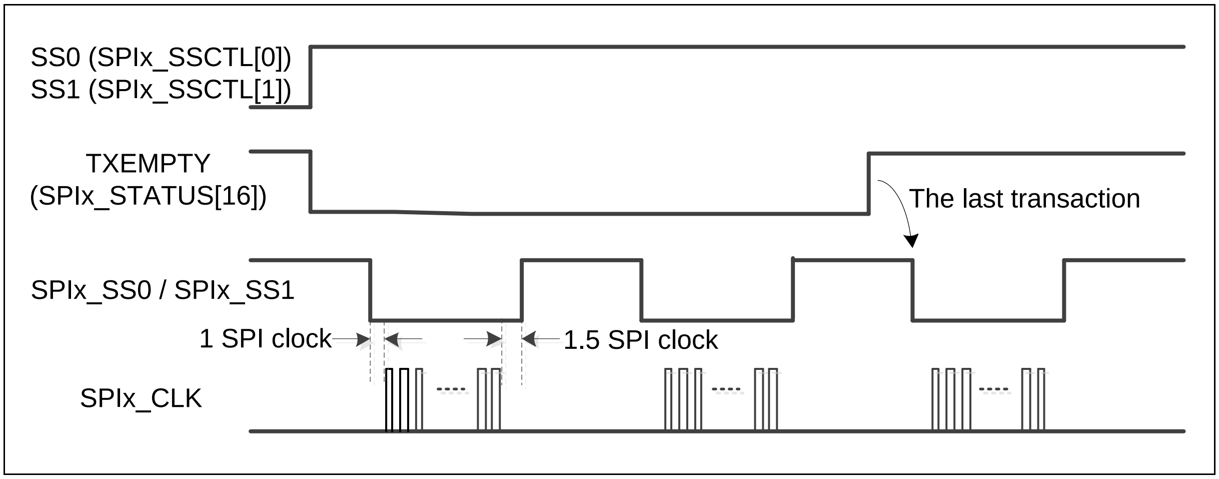


Figure 6.17-6 Automatic Slave Selection (SSACTPOL = 0, SUSPITV > 0x2)

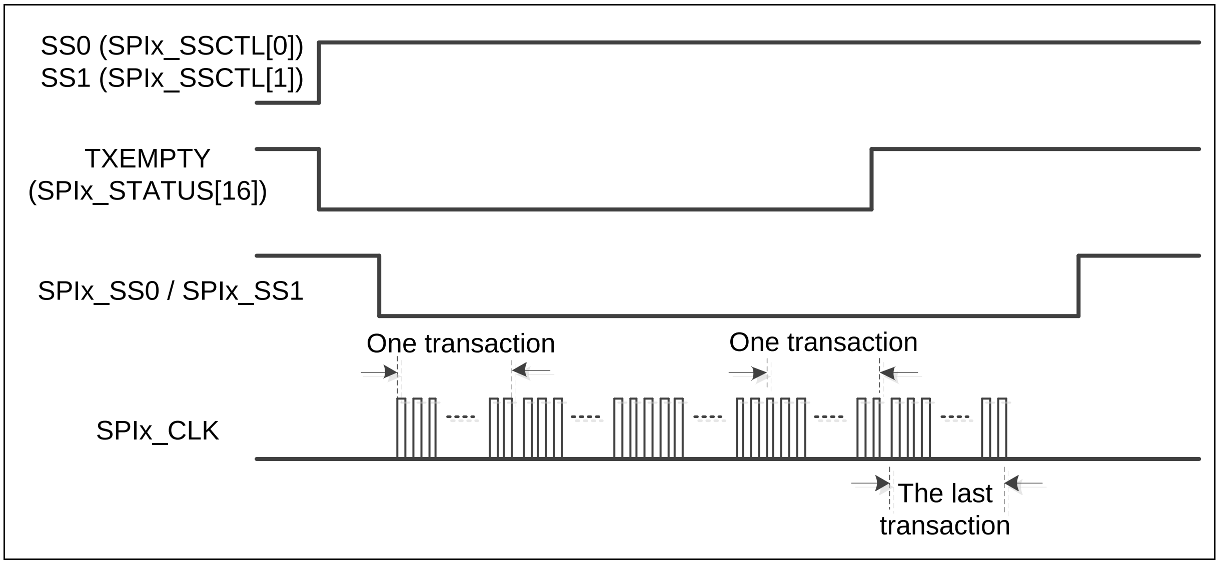


Figure 6.17-7 Automatic Slave Selection (SSACTPOL = 0, SUSPITV < 0x3)

6.17.5.3 Byte Reorder and Suspend Function

When the transfer is set as MSB first (LSB = 0) and the REORDER (SPIx_CTL[19]) is set to 1, the data stored in the TX buffer and RX buffer will be rearranged in the order as [Byte0, Byte1, Byte2, Byte3] in 32-bit transfer (DWIDTH = 0). The sequence of transmitted/received data will be Byte0, Byte1, Byte2, and then Byte3. If the DWIDTH is set as 24-bit transfer mode, the data in TX buffer and RX buffer will be rearranged as [unknown byte, Byte0, Byte1, Byte2]. The SPI controller will transmit/receive data with the sequence of Byte0, Byte1 and then Byte2. Each byte will be transmitted/received with MSB first. The rule of 16-bit mode is the same as above. Byte Reorder function is only available when DWIDTH is configured as 16, 24, and 32 bits.

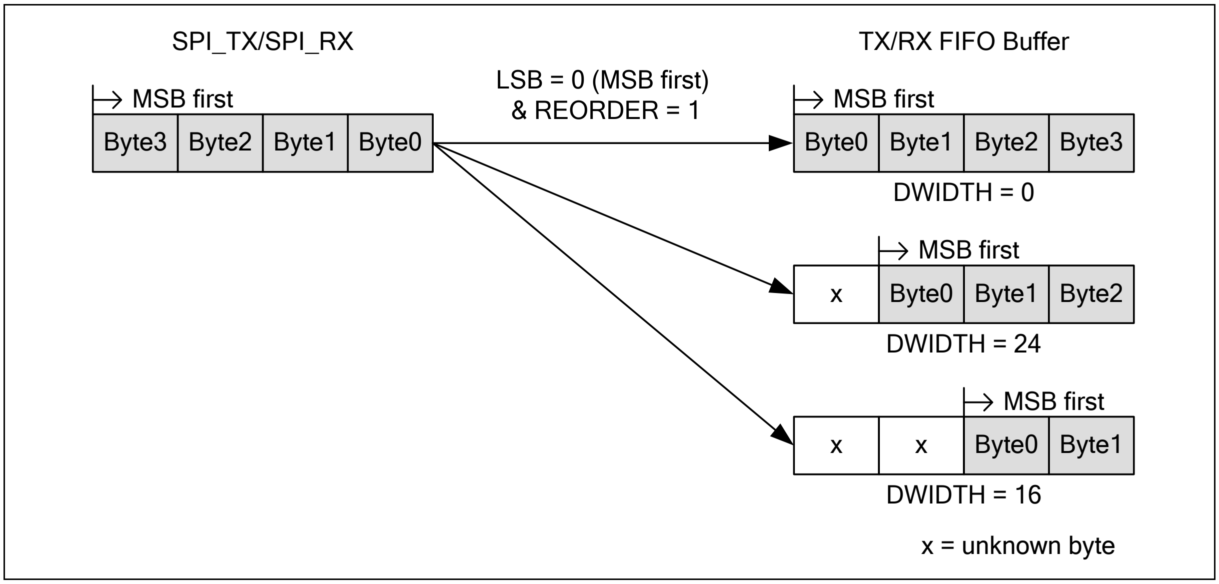


Figure 6.17-8 Byte Reorder Function

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]).

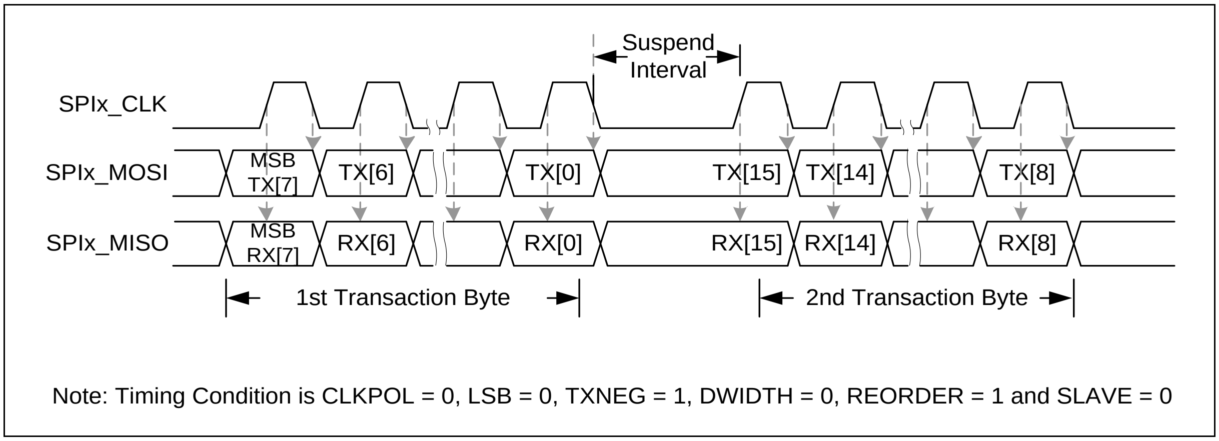


Figure 6.17-9 Timing Waveform for Byte Suspend

6.17.5.4 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.

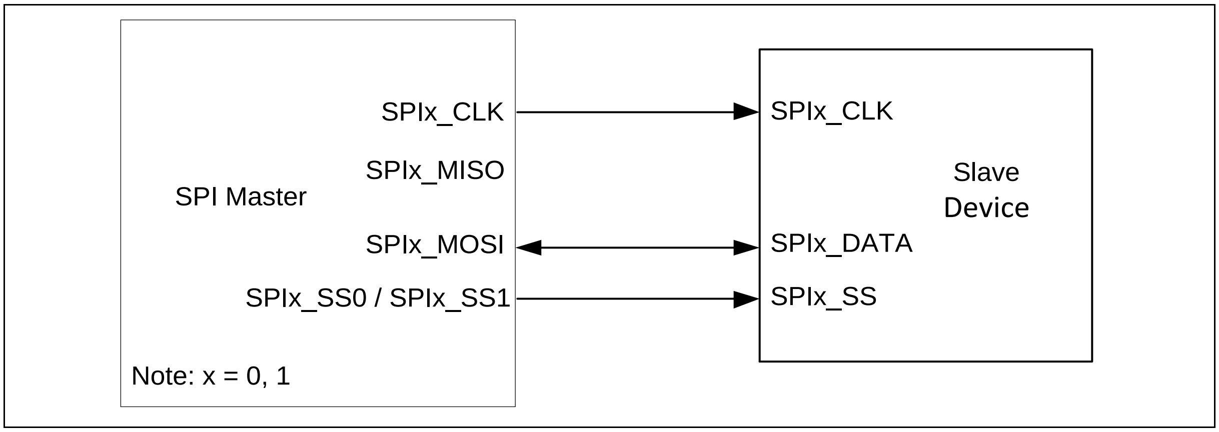


Figure 6.17-10 SPI Half-Duplex Master Mode Application Block Diagram

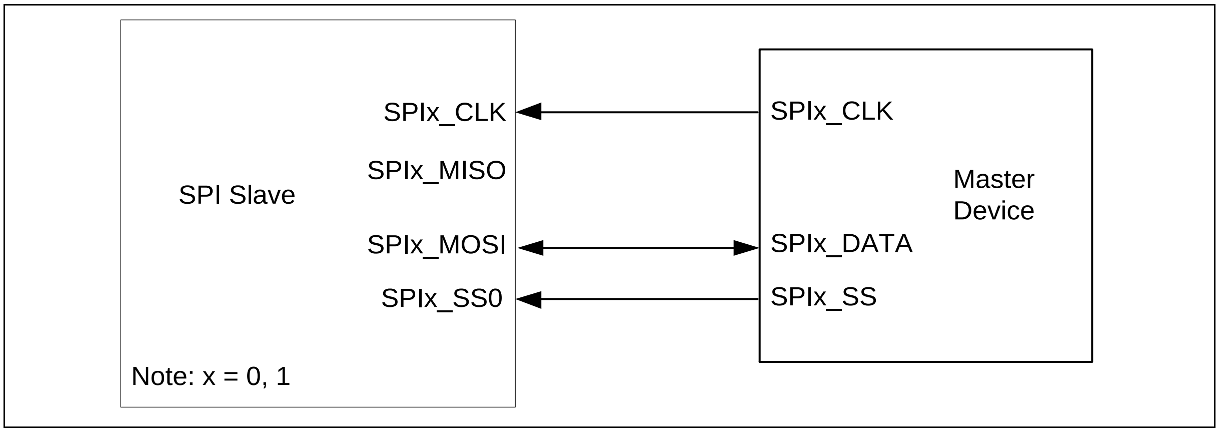


Figure 6.17-11 SPI Half-Duplex Slave Mode Application Block Diagram

6.17.5.5 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. After enabling this mode, the output SPI bus clock will be sent out in 6 peripheral clock cycles. In this mode, the data which has been written into transmit FIFO will be loaded into transmit shift register and sent out.

6.17.5.6 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.

Note: SPI supports single request PDMA (Read/Write) only, burst request PDMA is not supported.

6.17.5.7 FIFO Buffer Operation

The SPI controllers include 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.

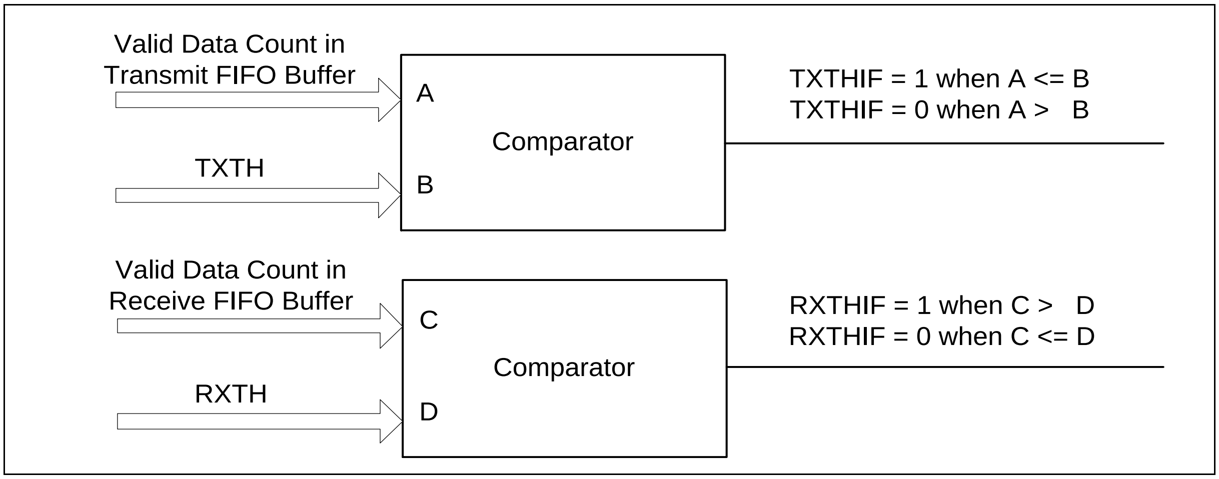


Figure 6.17-12 FIFO Threshold Comparator

In Master mode, when the first datum is written to the SPIx_TX register, the TXEMPTY flag (SPIx_STATUS[16]) will be cleared to 0. The transmission will start after 1 APB 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.

In the example 1 of Figure 6.17-13, it indicates the updated condition of TXEMPTY (SPIx_STATUS[16]) and the relationship among the FIFO buffer, shift register and the skew buffer. 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.

In the Example 2, it indicates the updated condition of TXFULL (SPIx_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.

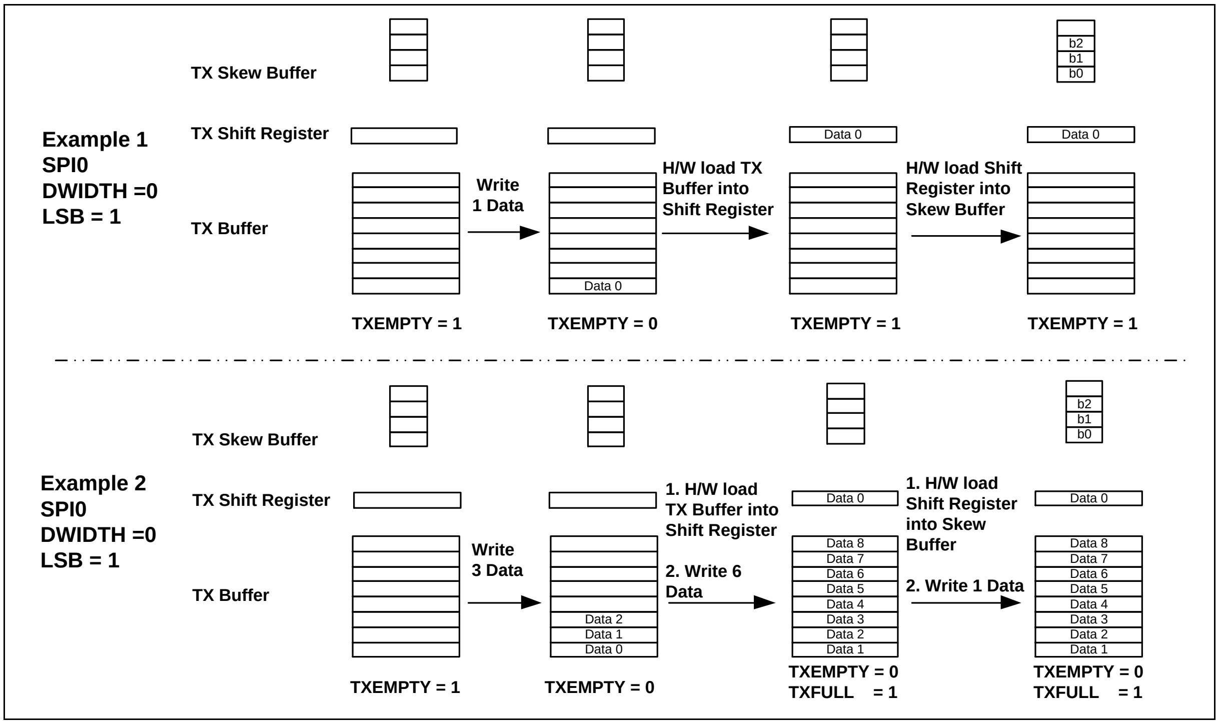


Figure 6.17-13 Transmit FIFO Buffer Example

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). 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).

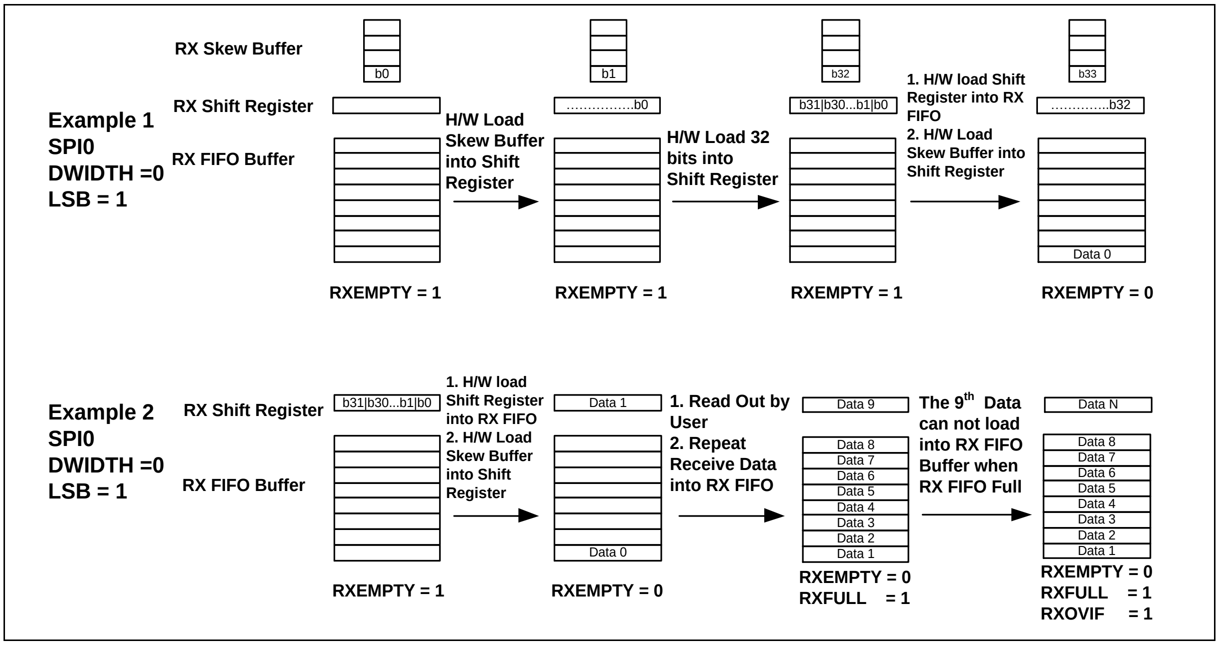


Figure 6.17-14 Receive FIFO Buffer Example

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_SS0 goes to inactive state.

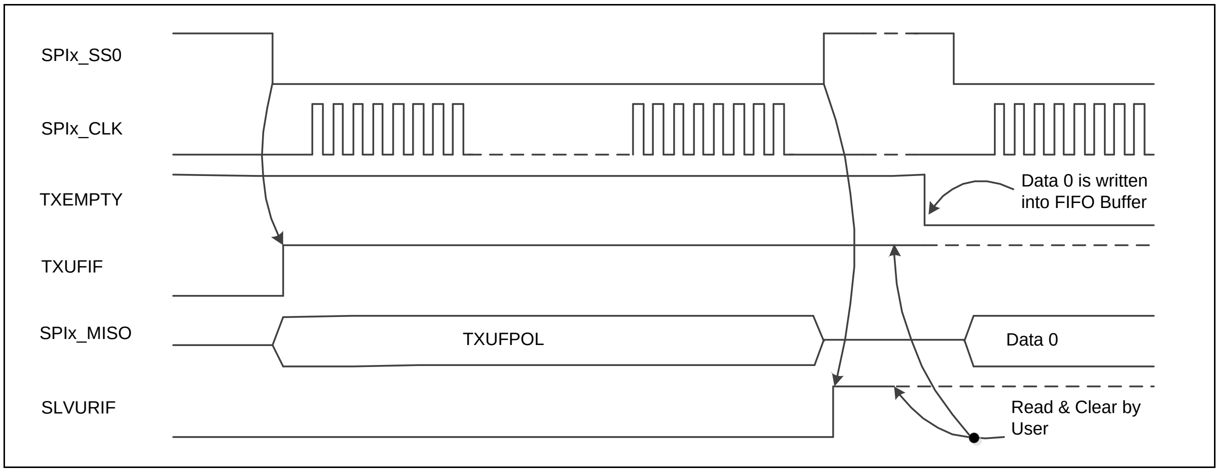


Figure 6.17-15 TX Underflow Event and Slave Under Run Event

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.

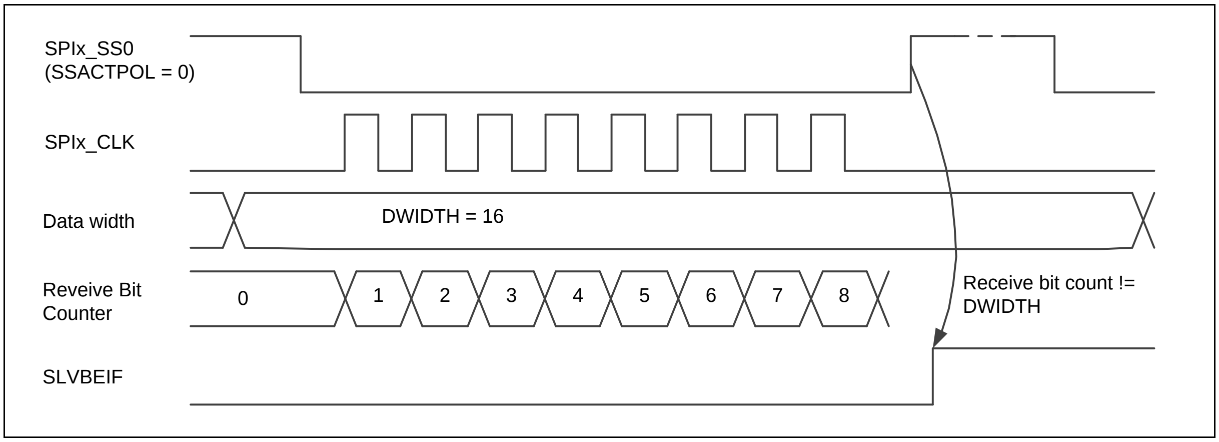


Figure 6.17-16 Slave Mode Bit Count Error

A receive time-out function is built-in in this controller. When the receive FIFO is not empty and no read operation in receive FIFO over 64 SPI 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.

6.17.5.8 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_SS0 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 is 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

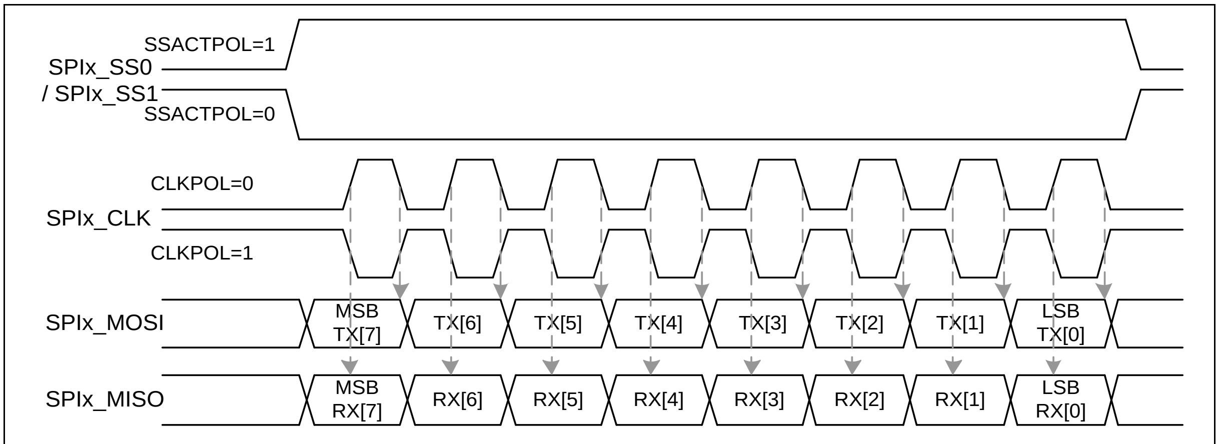
  In FIFO mode, if the valid data count of the transmit FIFO buffer is less than or equal to the setting value of TXTH (SPIx_FIFOCTL[30:28]), the transmit FIFO interrupt flag TXTHIF (SPIx_STATUS[18]) will be set to 1. The SPI controller will generate a transmit FIFO interrupt to the system if the transmit 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.

6.17.6 Timing Diagram

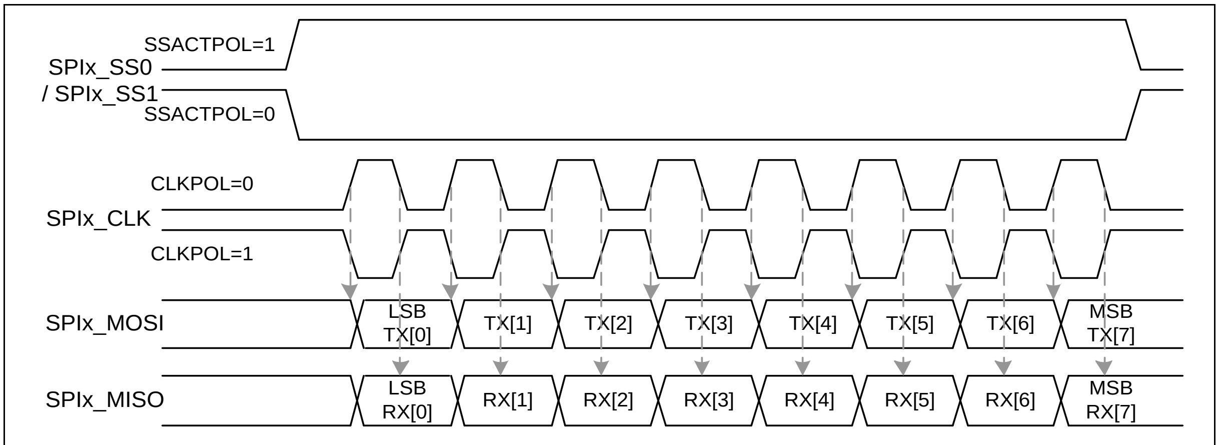
The active state of slave selection signal can be defined by setting the SSACTPOL (SPIx_SSCTL[2]). The SPI clock which is in idle state can be configured as high or low state by setting the CLKPOL (SPIx_CTL[3]). It also provides the bit length of a transaction word in DWIDTH (SPIx_CTL[12:8]), and transmitting/receiving 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.



Master Mode: SLAVE=0, LSB=0, DWIDTH=0x08

1. CLKPOL=0, TXNEG=1, RXNEG=0 or
2. CLKPOL=1, TXNEG=0, RXNEG=1

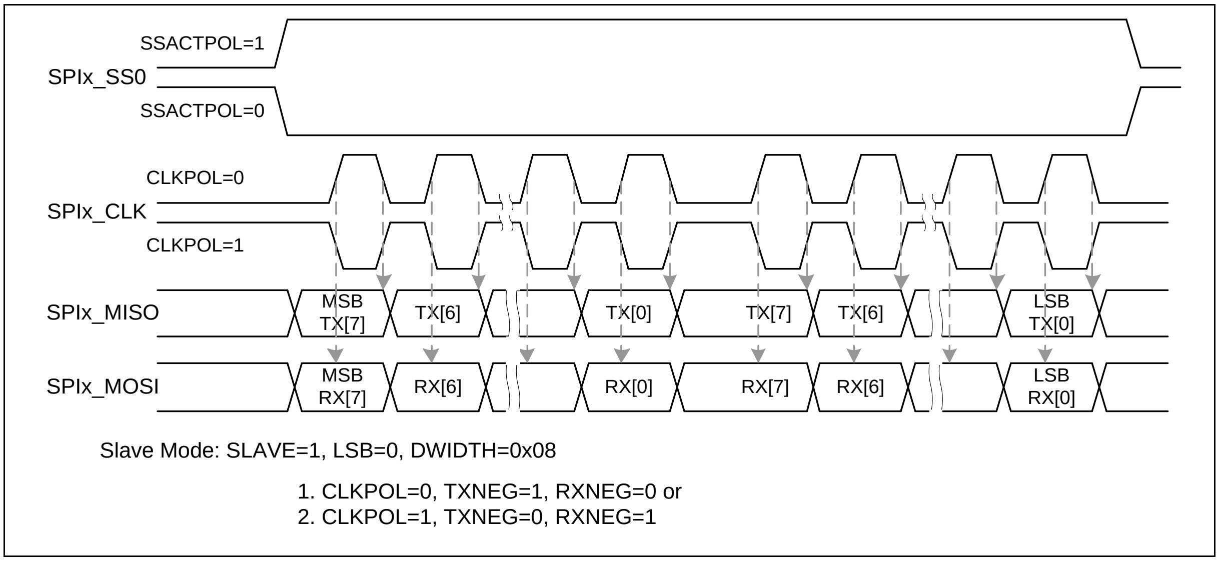
Figure 6.17-17 SPI Timing in Master Mode



Master Mode: SLAVE=0, LSB=1, DWIDTH=0x08

1. CLKPOL=0, TXNEG=0, RXNEG=1 or
2. CLKPOL=1, TXNEG=1, RXNEG=0

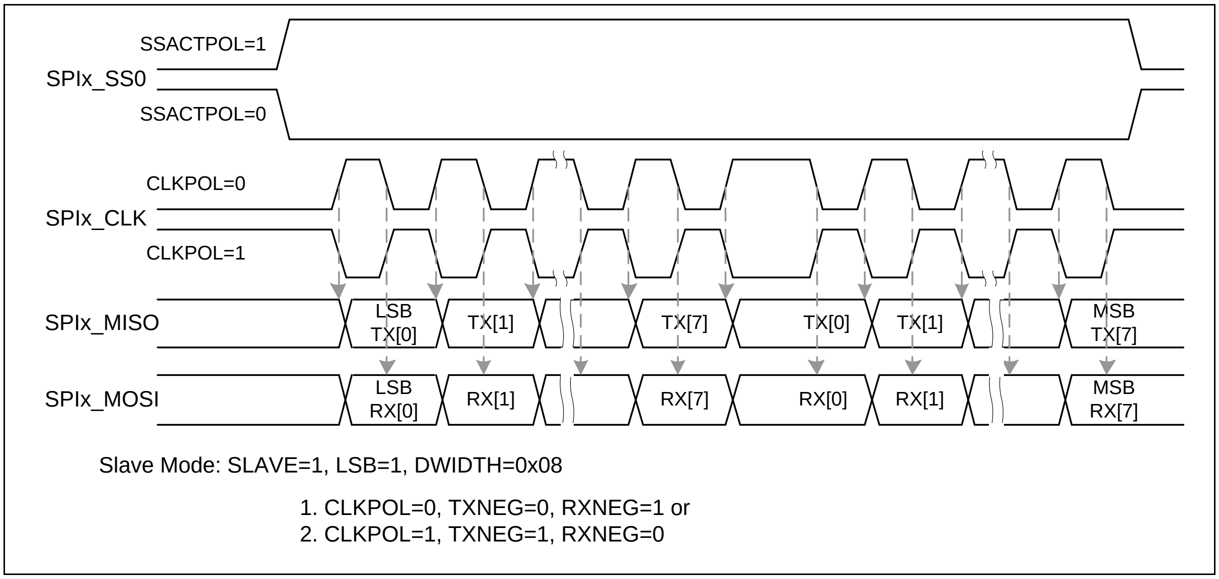
Figure 6.17-18 SPI Timing in Master Mode (Alternate Phase of SPIx_CLK)



Slave Mode: SLAVE=1, LSB=0, DWIDTH=0x08

1. CLKPOL=0, TXNEG=1, RXNEG=0 or
2. CLKPOL=1, TXNEG=0, RXNEG=1

Figure 6.17-19 SPI Timing in Slave Mode



Slave Mode: SLAVE=1, LSB=1, DWIDTH=0x08

1. CLKPOL=0, TXNEG=0, RXNEG=1 or
2. CLKPOL=1, TXNEG=1, RXNEG=0

Figure 6.17-20 SPI Timing in Slave Mode (Alternate Phase of SPIx_CLK)

6.17.7 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 SS0 (SPIx_SSCTL[0]) or SS1 (SPIx_SSCTL[1]) 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 SS0 (SPIx_SSCTL[0]) or SS1 (SPIx_SSCTL[1]) to 0 to inactivate the off-chip slave device.

Example 2:

The SPI controller is set as a full-duplex slave device and connects with an off-chip master device. The off-chip master device communicates with the on-chip SPI slave controller through the SPI interface with the following specifications:

- Data bit is latched on positive edge of SPI bus clock.
- Data bit is driven on negative edge of SPI bus clock.
- Data is transferred from LSB first.
- SPI bus clock is idle at high state.
- Only one byte of data to be transmitted/received in a transaction.
- Slave selection signal is active high.

The operation flow is as follows:

1. Write the SPIx_SSCTL register a proper value for the related settings of Slave mode.

2. Select high level for the input of slave selection signal by setting SSACTPOL (SPIx_SSCTL[2]) to 1.
3. Write the related settings into the SPIx_CTL register to control this SPI slave actions
   1) Set the SPI controller as slave device by setting SLAVE (SPIx_CTL[18]) to 1.
   2) Select the SPI clock idle state at high by setting CLKPOL (SPIx_CTL[3]) to 1.
   3) Select data transmitted on negative edge of SPI bus clock by setting TXNEG (SPIx_CTL[2]) to 1.
   4) Select data latched on positive edge of SPI bus clock by clearing RXNEG (SPIx_CTL[1]) to 0.
   5) Set the bit length of a transaction as 8-bit in DWIDTH bit field (SPIx_CTL[12:8] = 0x08).
4. Set LSB transfer first by setting LSB (SPIx_CTL[13]) to 1.
5. Set the SPIEN (SPIx_CTL[0]) to 1. Wait for the slave select trigger input and SPI clock input from the off-chip master device to start the data transfer.
6. If this SPI slave attempts to transmit (be read) one byte data to the off-chip master device, write the byte data that will be transmitted into the SPIx_TX register.
7. If this SPI slave just only attempts to receive (be written) one byte data from the off-chip master device and does not care what data will be transmitted, the SPIx_TX register does not need to be updated by software.
8. Waiting for SPI interrupt if the UNITIEN (SPIx_CTL[17]) is set to 1, or just polling the unit transfer interrupt flag UNITIF (SPIx_STATUS[1]).
9. Read out the received one byte data from SPIx_RX register.
10. Go to 7 to continue another data transfer or stop data transfer.

6.17.8 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 = 0xB006_1000 + (0x0000_1000 * x)** **x = 0, 1**				
SPIx_CTL	SPIx_BA+0x00	R/W	SPI Control Register	0x0000_0034
SPIx_CLKDIV	SPIx_BA+0x04	R/W	SPI Clock Divider Register	0x0000_0000
SPIx_SSCTL	SPIx_BA+0x08	R/W	SPI Slave Select Control Register	0x0000_0000
SPIx_PDMACTL	SPIx_BA+0x0C	R/W	SPI PDMA Control Register	0x0000_0000
SPIx_FIFOCTL	SPIx_BA+0x10	R/W	SPI FIFO Control Register	0x2200_0000
SPIx_STATUS	SPIx_BA+0x14	R/W	SPI Status Register	0x0005_0110
SPIx_TX	SPIx_BA+0x20	W	SPI Data Transmit Register	0x0000_0000
SPIx_RX	SPIx_BA+0x30	R	SPI Data Receive Register	0x0000_0000

6.17.9 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

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 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_DIVCTL2. **Note:** The time interval must be larger than or equal 5 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

31	30	29	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**	**SS1**	**SS0**

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 SS0 (SPIx_SSCTL[0]) or SS1 (SPIx_SSCTL[1]). 1 = Automatic slave selection function Enabled.
[2]	**SSACTPOL**	**Slave Selection Active Polarity** This bit defines the active polarity of slave selection signal (SPIx_SS0 and SPIx_SS1). 0 = The slave selection signal SPIx_SS0/ SPIx_SS1 is active low. 1 = The slave selection signal SPIx_SS0/ SPIx_SS1 is active high.
[1]	**SS1**	**Slave Selection 1 Control (Master Only)**

NUC980 SERIES TECHNICAL REFERENCE MANUAL

		If AUTOSS bit is cleared to 0, 0 = set the SPIx_SS1 line to inactive state. 1 = set the SPIx_SS1 line to active state. If the AUTOSS bit is set to 1, 0 = Keep the SPIx_SS1 line at inactive state. 1 = SPIx_SS1 line will be automatically driven to active state for the duration of data transfer, and will be driven to inactive state for the rest of the time. The active state of SPIx_SS1 is specified in SSACTPOL (SPIx_SSCTL[2]).
[0]	**SS0**	**Slave Selection 0 Control (Master Only)** If AUTOSS bit is cleared to 0, 0 = set the SPIx_SS0 line to inactive state. 1 = set the SPIx_SS0 line to active state. If the AUTOSS bit is set to 1, 0 = Keep the SPIx_SS0 line at inactive state. 1 = SPIx_SS0 line will be automatically driven to active state for the duration of data transfer, and will be driven to inactive state for the rest of the time. The active state of SPIx_SS0 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. 2. 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

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. SPIx_CTL[0] = 1 and TXEMPTY = 0. b. For SPI Master mode, SPIx_CTL[0] = 1 and TXEMPTY = 1 but the current transaction is not finished yet. c. For SPI Master mode, SPIx_CTL[0] = 1 and RXONLY = 1. d. For SPI Slave mode, the 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, the 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. 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. **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.

6.18 Quad Serial Peripheral Interface (QSPI)

6.18.1 Overview

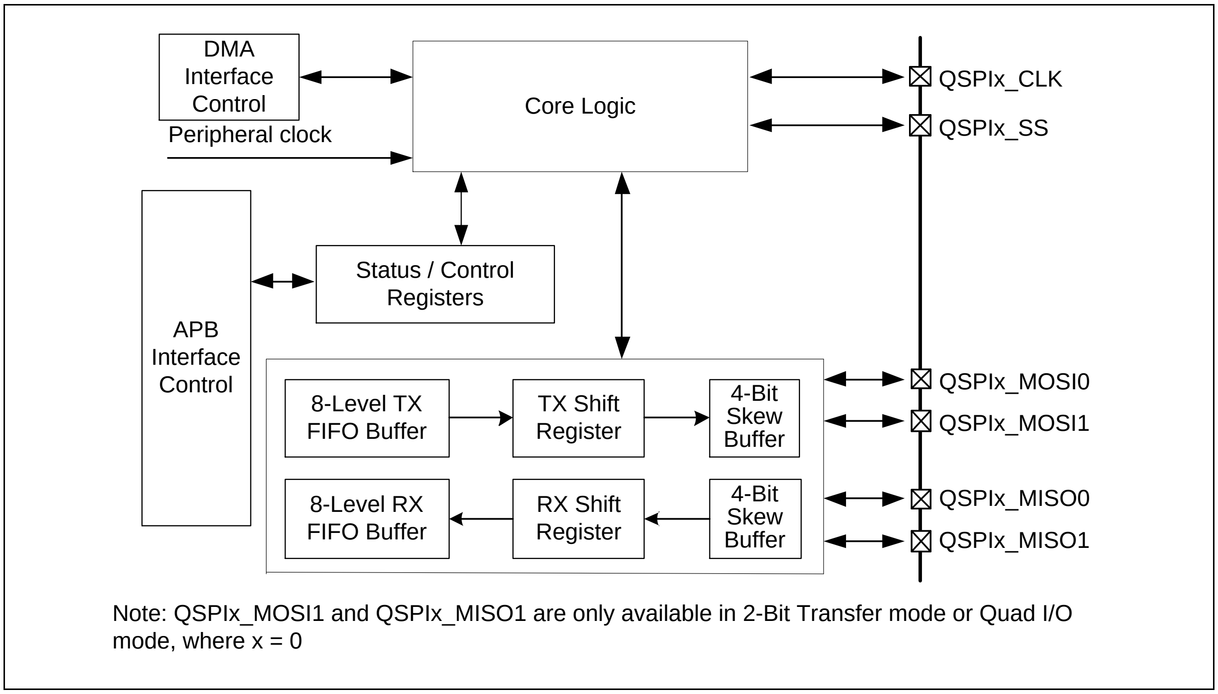
The Quad Serial Peripheral Interface (QSPI) applies to synchronous serial data communication and allows full duplex transfer. Devices communicate in Master/Slave mode with the 4-wire bi-direction interface. The chip contains one QSPI controller performing a serial-to-parallel conversion on data received 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.

6.18.2 Features

- Supports Master or Slave mode operation
- Master mode up to 100 MHz and Slave mode up to 30 MHz (when chip works at V_{DD} = 2.7~3.6V)
- 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

6.18.3 Block Diagram



Note: QSPIx_MOSI1 and QSPIx_MISO1 are only available in 2-Bit Transfer mode or Quad I/O mode, where x = 0

Figure 6.18-1 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.

6.18.4 Basic Configuration

6.18.4.1 QSPI0 Basic Configuration

- Clock source Configuration

  - Select the source of QSPI0 peripheral clock on QSPI0_S (CLK_ DIVCTL2 [9:8]).
  - Enable QSPI0 peripheral clock in QSPI0CKEN (CLK_ PCLKEN1[4]).
- Reset Configuration
  - Reset QSPI0 controller in QSPI0RST (SYS_ APBIPRST1[4]).
- Pin Configuration

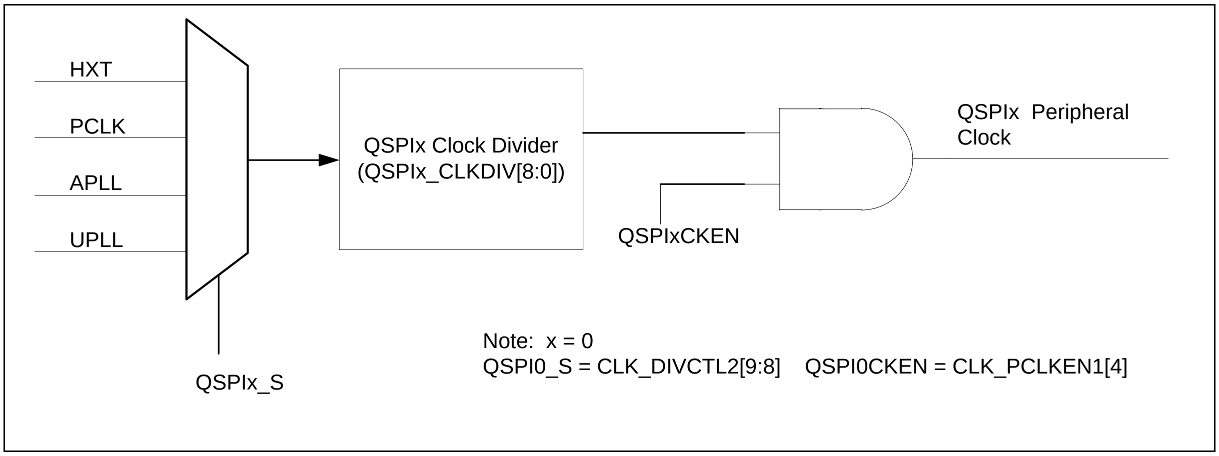
Group	Pin Name	GPIO	MFP
QSPI0	QSPI0_CLK	PD.3	MFP1
	QSPI0_MISO0	PD.5	MFP1
	QSPI0_MISO1	PD.7	MFP1
	QSPI0_MOSI0	PD.4	MFP1
	QSPI0_MOSI1	PD.6	MFP1
	QSPI0_SS0	PD.2	MFP1
	QSPI0_SS1	PA.0, PD.0	MFP1
		PB.3	MFP6

6.18.5 Functional Description

6.18.5.1 Terminology

<u>QSPI Peripheral Clock and QSPI Bus Clock</u>

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 can be HXT, PCLK, APLL, or UPLL. QSPIx_S of CLK_DIVCTL2 register determines the clock source of the peripheral clock. The DIVIDER (QSPIx_CLKDIV[8:0]) setting determines the divisor of the clock rate calculation.



Note: x = 0
QSPI0_S = CLK_DIVCTL2[9:8] QSPI0CKEN = CLK_PCLKEN1[4]

Figure 6.18-2 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. If the clock source of peripheral clock is not system clock, the frequency of QSPI peripheral clock shall be slower than the system clock frequency 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.

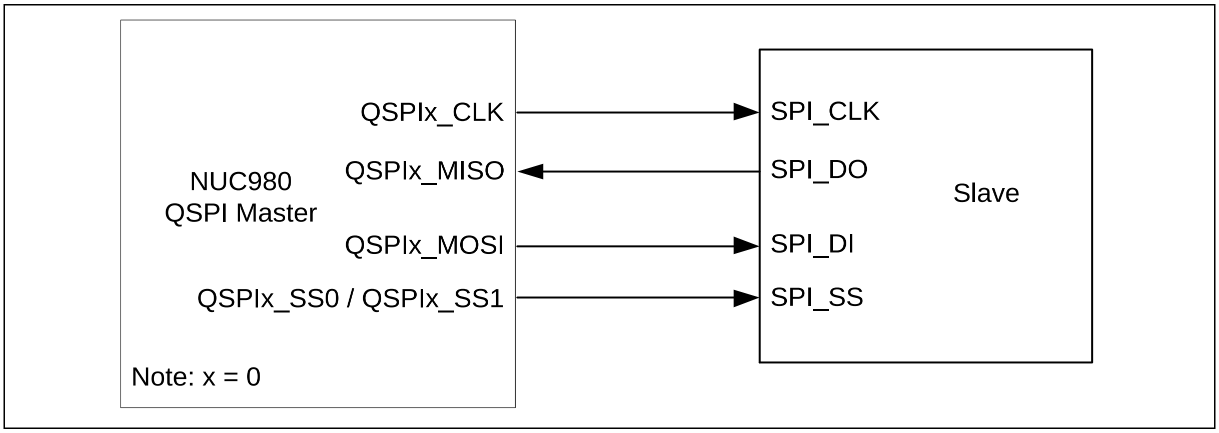


Figure 6.18-3 QSPI Full-Duplex Master Mode Application Block Diagram

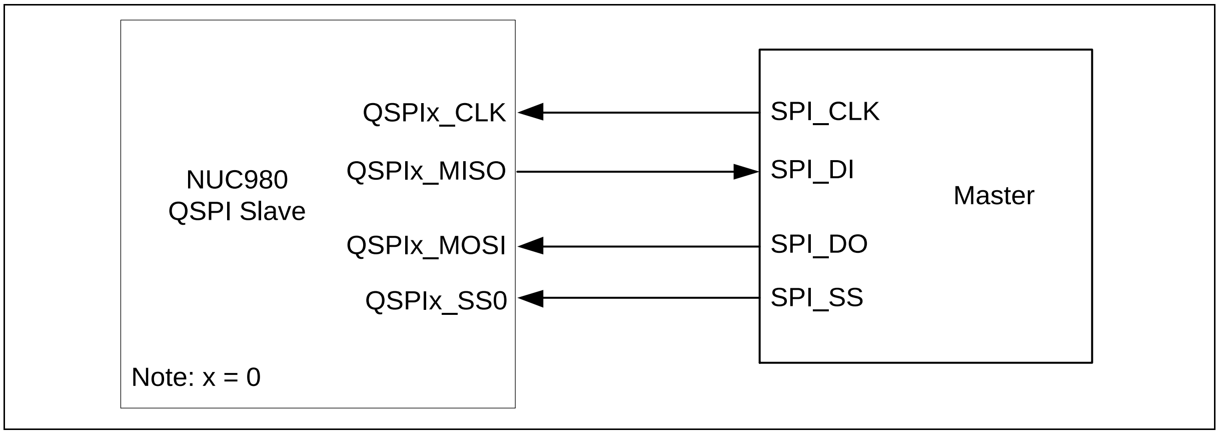


Figure 6.18-4 QSPI Full-Duplex Slave Mode Application Block Diagram

Slave Selection

In Master mode, the QSPI controller can drive off-chip slave device through the slave select output pin QSPIx_SS0 or QSPIx_SS1. In Slave mode, the off-chip master device only drives the slave selection signal from the QSPIx_SS0 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 will be set to 1.

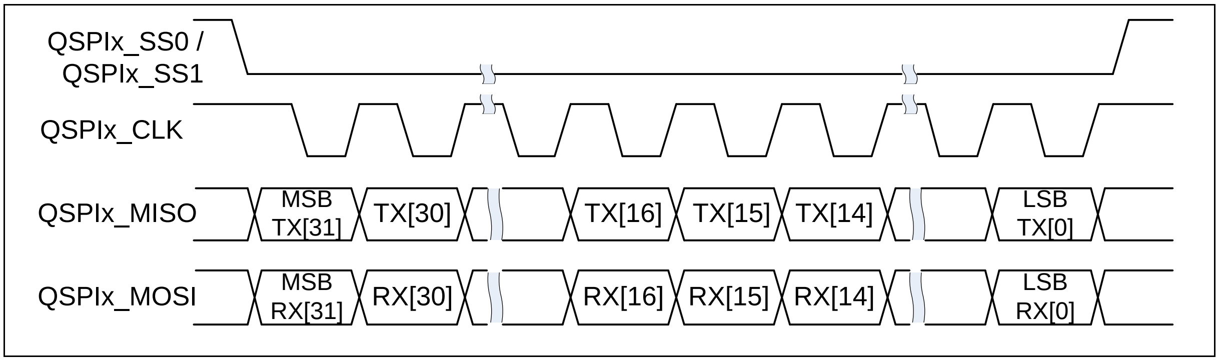


Figure 6.18-5 32-bit in One Transaction

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).

6.18.5.2 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_SS0 or QSPIx_SS1 pin according to whether SS0 (QSPIx_SSCTL[0]) or SS1 (QSPIx_SSCTL[1]) is enabled or not. The slave selection signal will be set to active state by the SPI 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 SS0 or SS1 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.

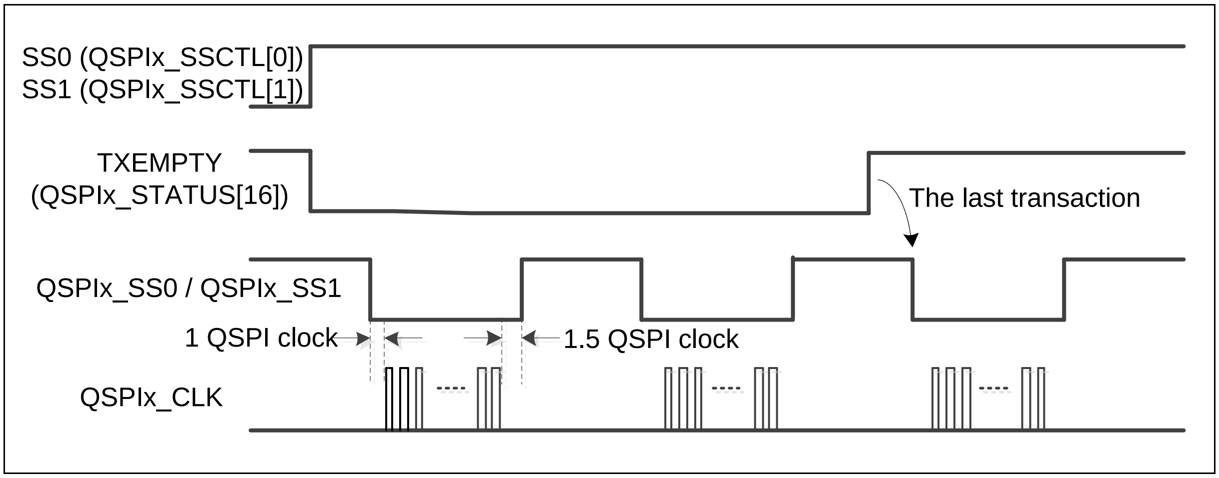


Figure 6.18-6 Automatic Slave Selection (SSACTPOL = 0, SUSPITV > 0x2)

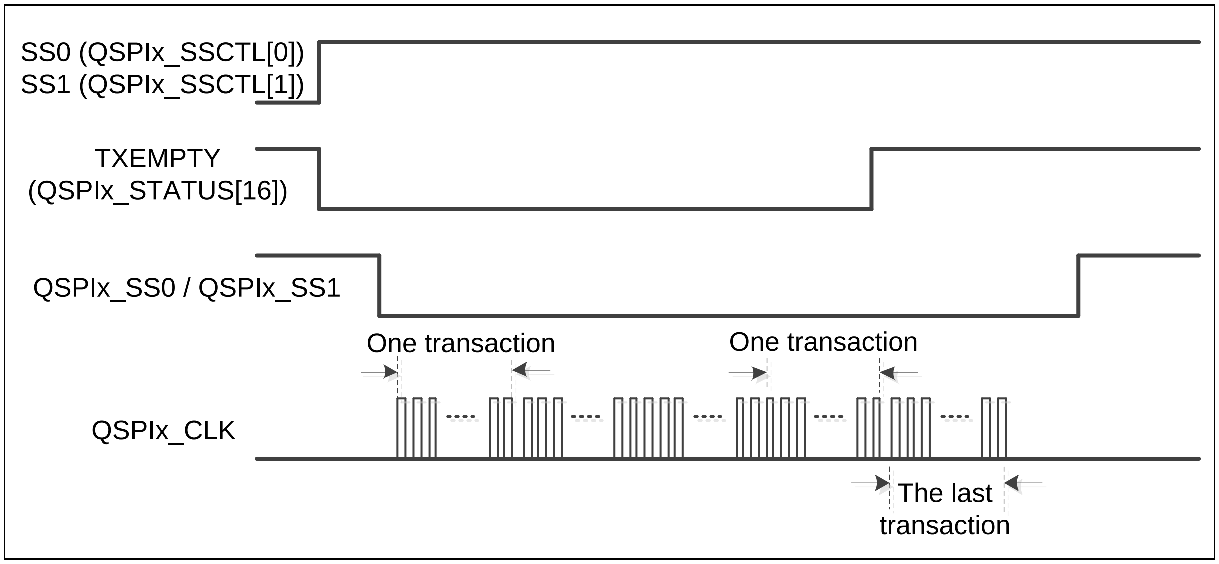


Figure 6.18-7 Automatic Slave Selection (SSACTPOL = 0, SUSPITV < 0x3)

6.18.5.3 Byte Reorder and Suspend Function

When the transfer is set as MSB first (LSB = 0) and the REORDER (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.

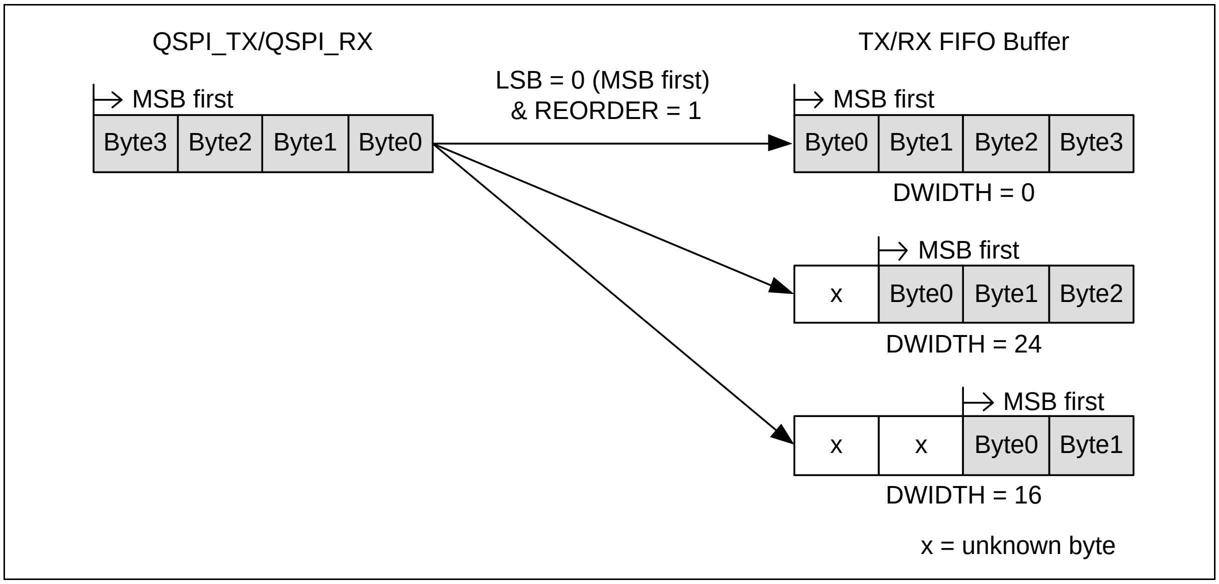


Figure 6.18-8 Byte Reorder Function

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]).

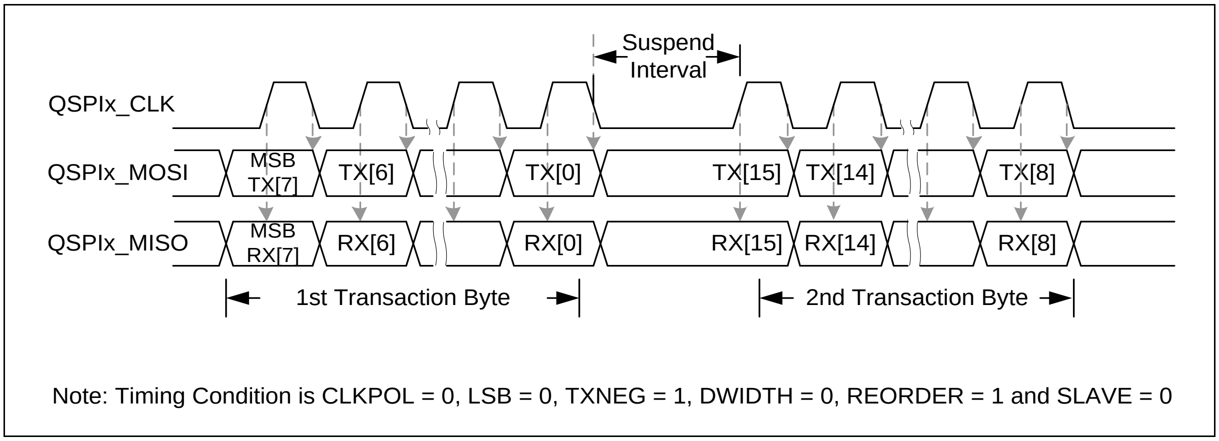


Figure 6.18-9 Timing Waveform for Byte Suspend

6.18.5.4 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.

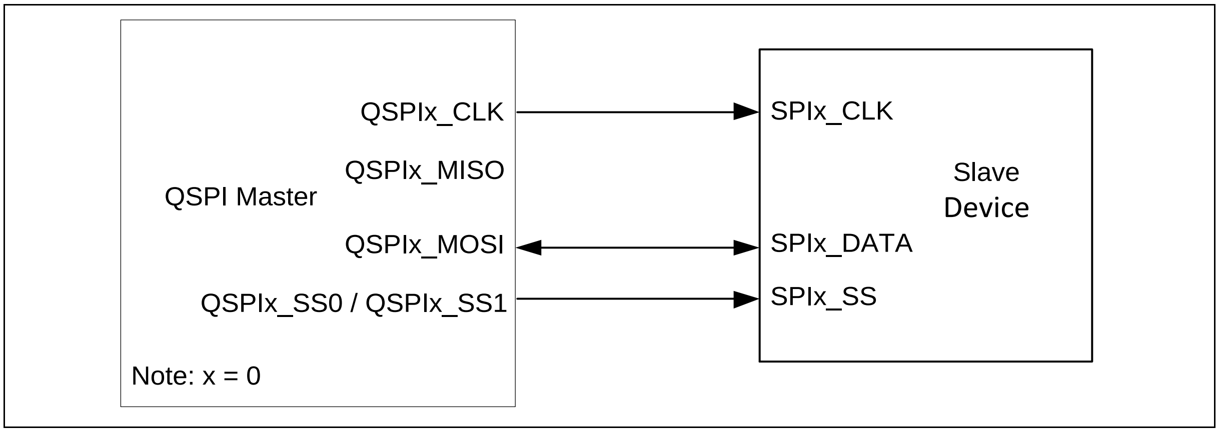


Figure 6.18-10 QSPI Half-Duplex Master Mode Application Block Diagram

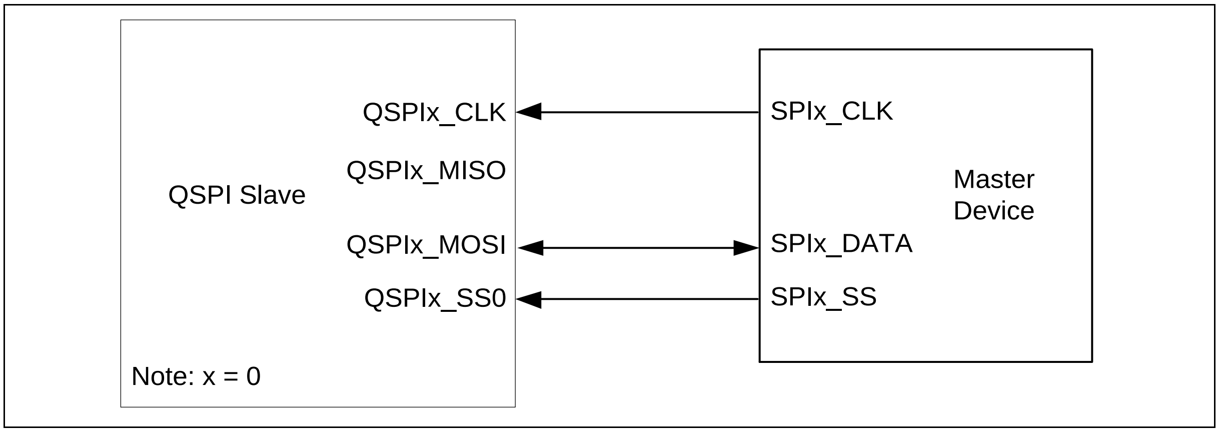


Figure 6.18-11 QSPI Half-Duplex Slave Mode Application Block Diagram

6.18.5.5 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. After enabling this mode, the output QSPI bus clock will be sent out in 6 peripheral clock cycles. In this mode, the data which has been written into transmit FIFO will be loaded into transmit shift register and sent out.

6.18.5.6 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_SS0 and QSPIx_SS1 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.

6.18.5.7 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.

Note: QSPI supports single request PDMA (Read/Write) only, burst request PDMA is not supported.

6.18.5.8 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 QSPIx_TX register will be transmitted through the QSPIx_MOSI0 pin and the odd data (TX Data (n+1)) stored in the QSPIx_TX register 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 QSPIx_TX register will be transmitted through the QSPIx_MISO0 pin and QSPIx_MISO1 pin respectively. In the meanwhile, the QSPIx_RX register 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.

Note: QSPI_SS0/QSPI_SS1 signals for Slave 0 and Slave 1 are in active state for two-bit transfer mode simultaneously when data are transmitting and receiving.

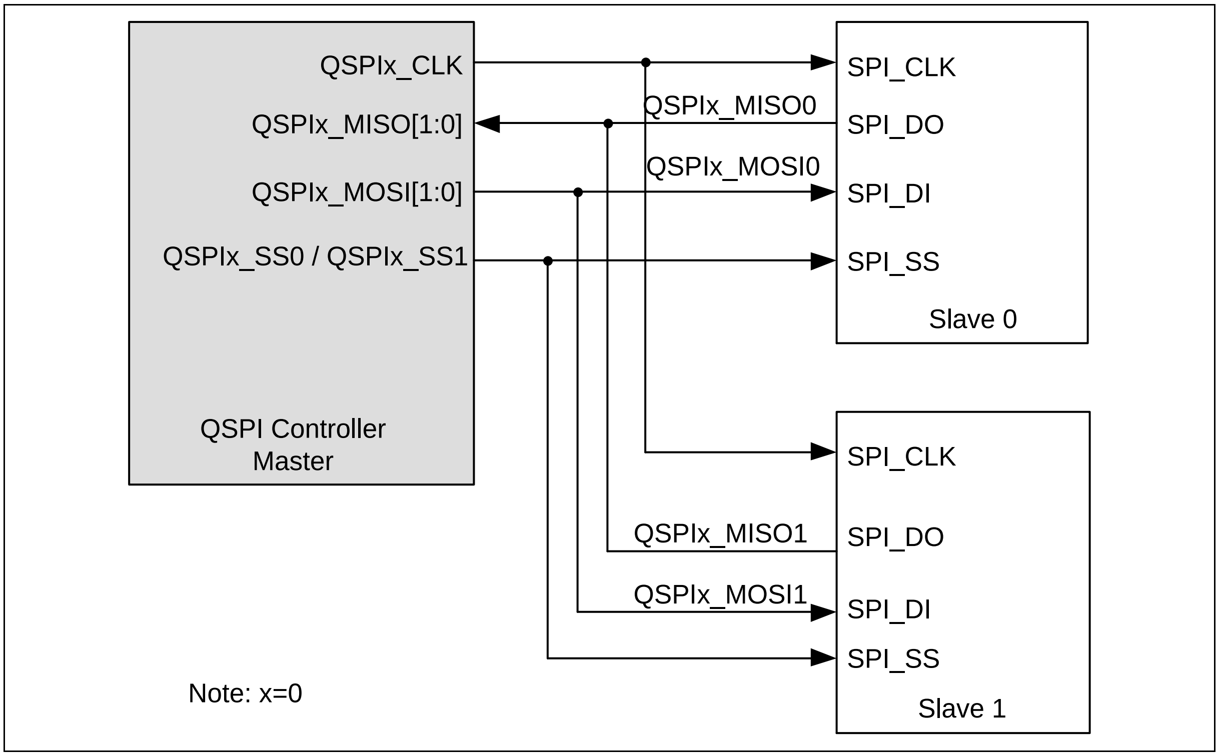


Figure 6.18-12 Two-bit Transfer Mode System Architecture

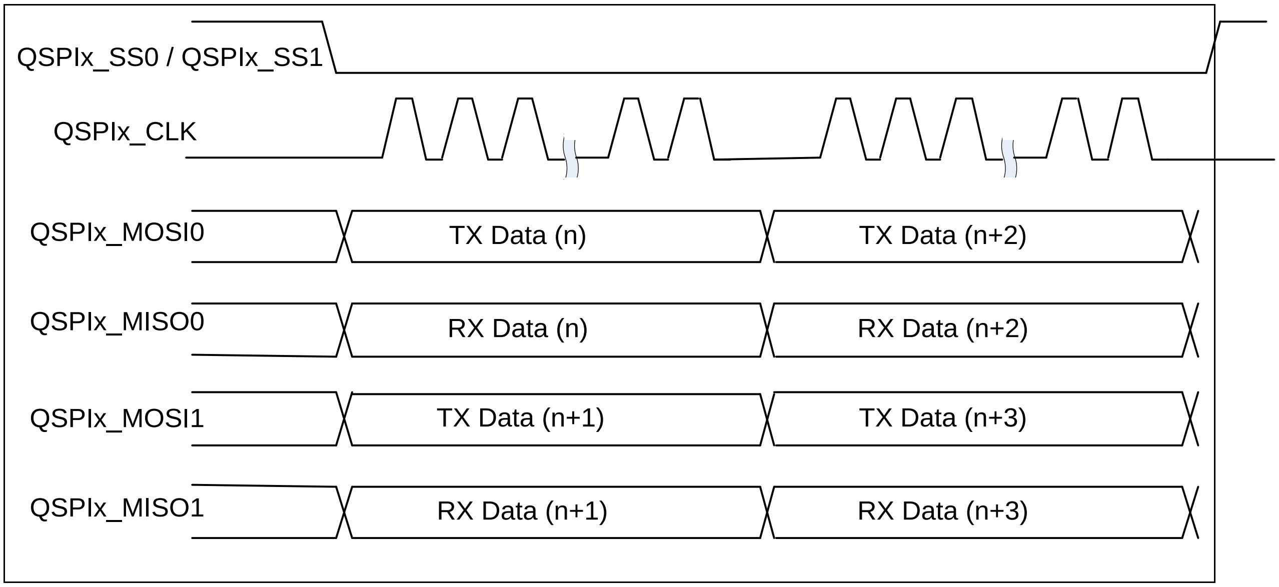


Figure 6.18-13 Two-bit Transfer Mode Timing (Master Mode)

6.18.5.9 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.

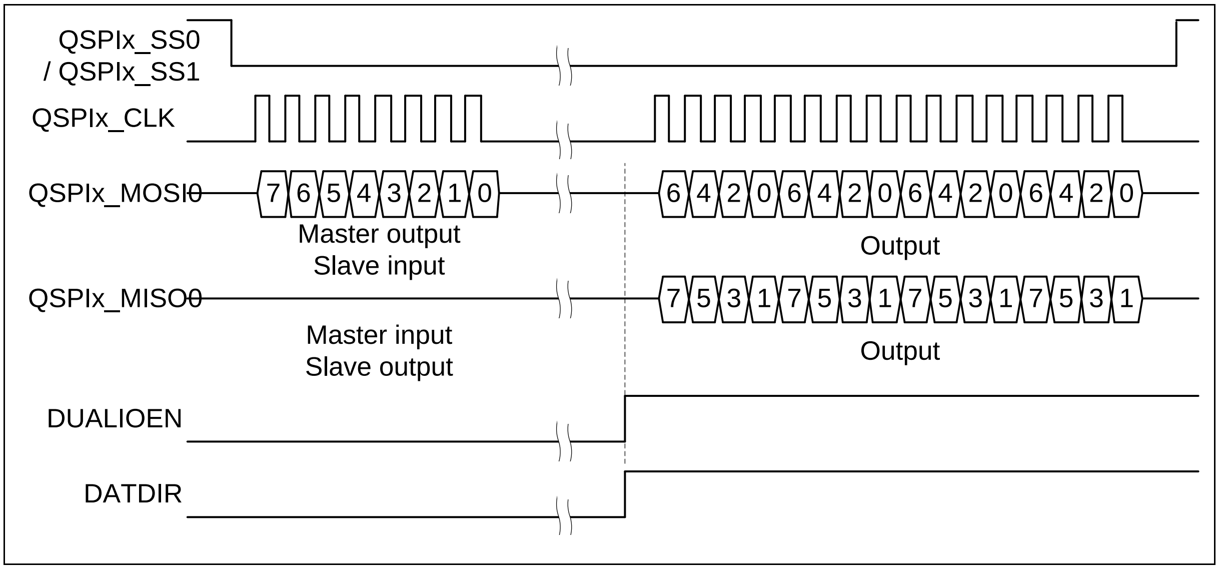


Figure 6.18-14 Bit Sequence of Dual Output Mode

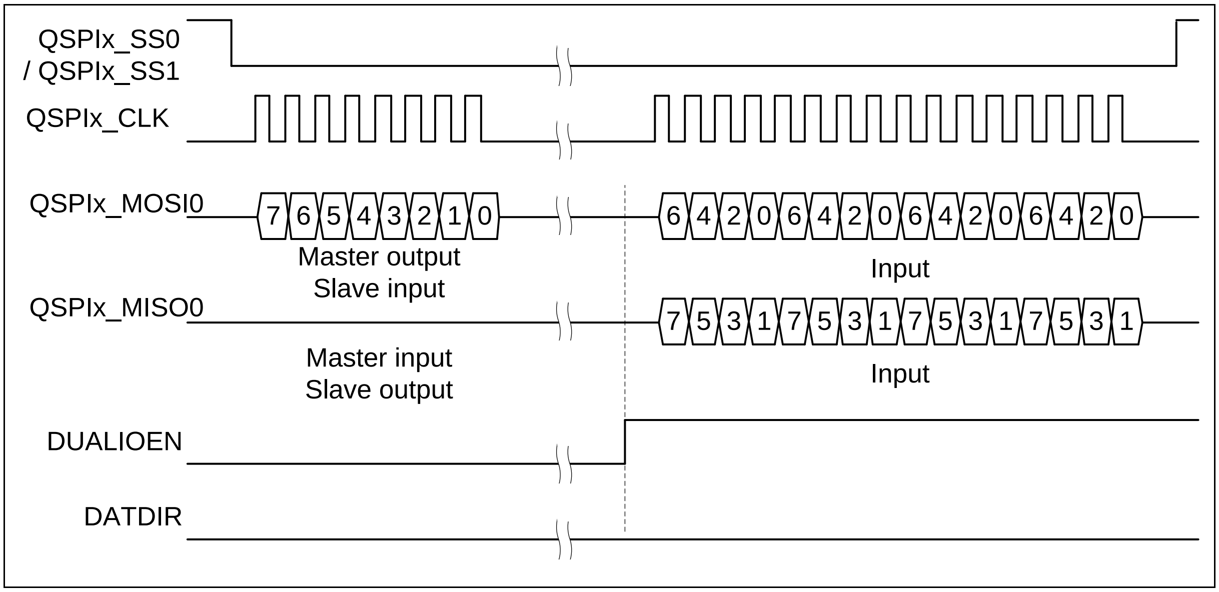


Figure 6.18-15 Bit Sequence of Dual Input Mode

6.18.5.10 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.

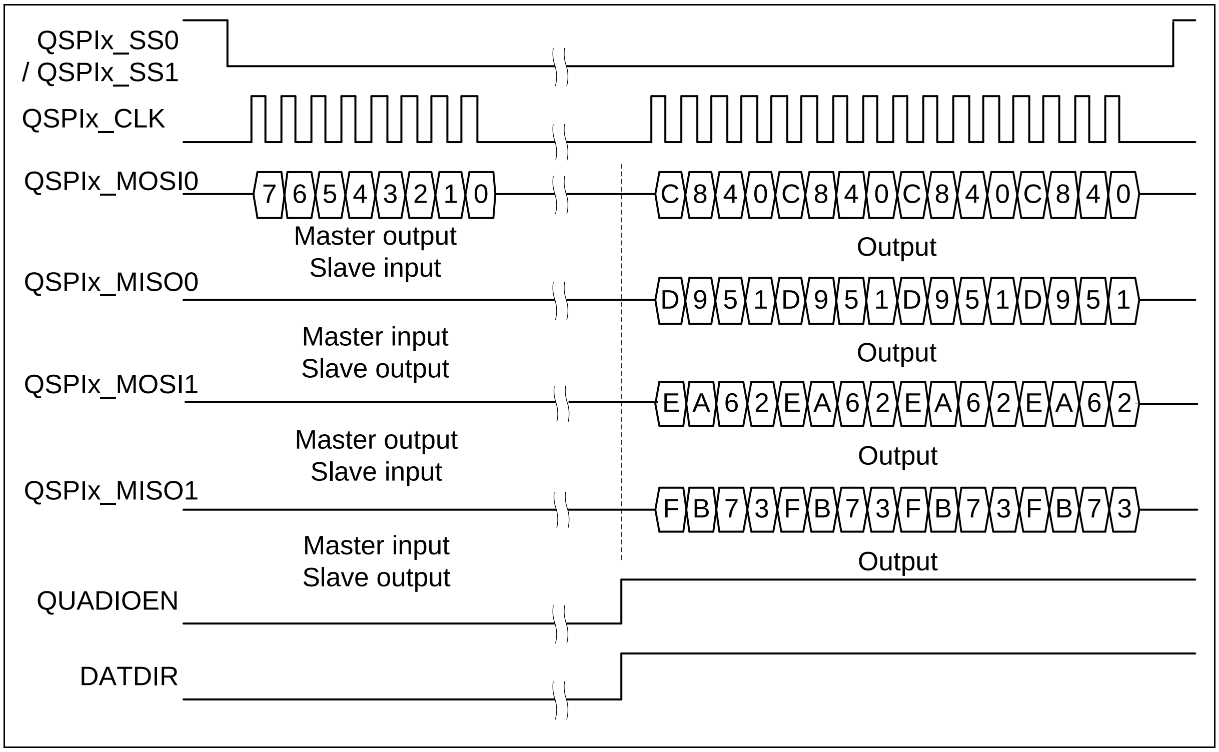


Figure 6.18-16 Bit Sequence of Quad Output Mode

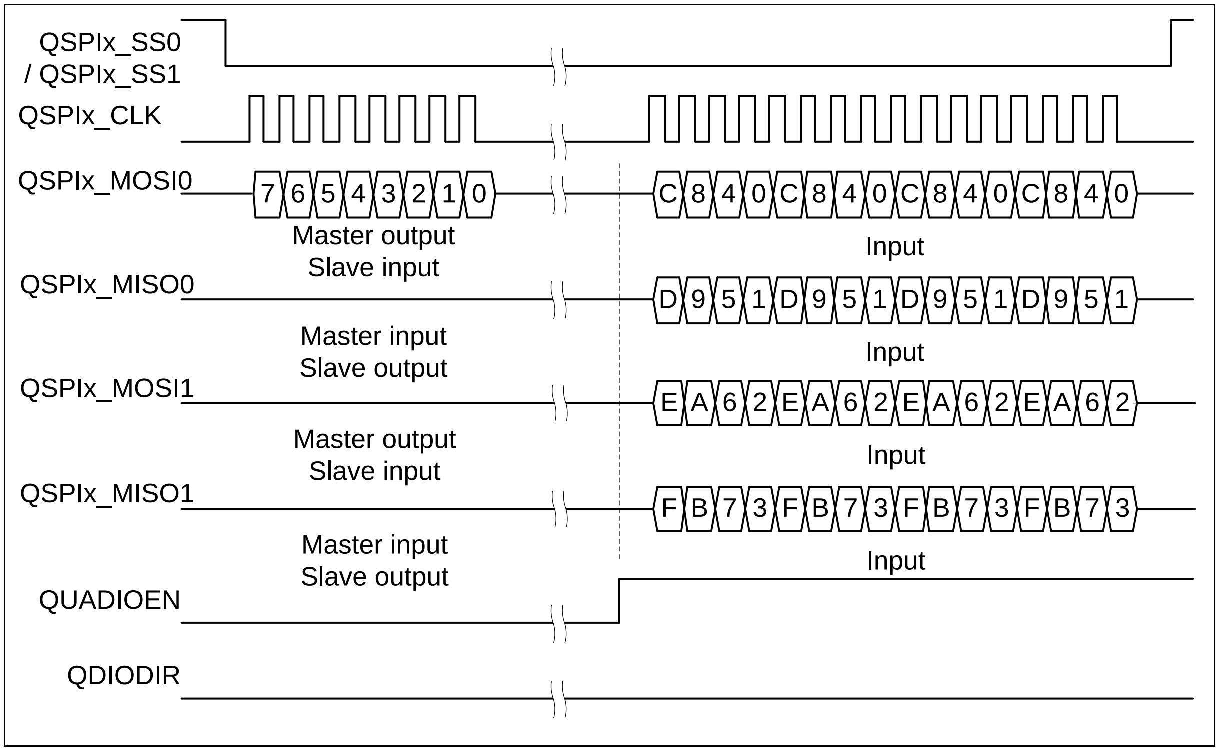


Figure 6.18-17 Bit Sequence of Quad Input Mode

6.18.5.11 FIFO Buffer Operation

The QSPI controllers include 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 (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 SPI 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.

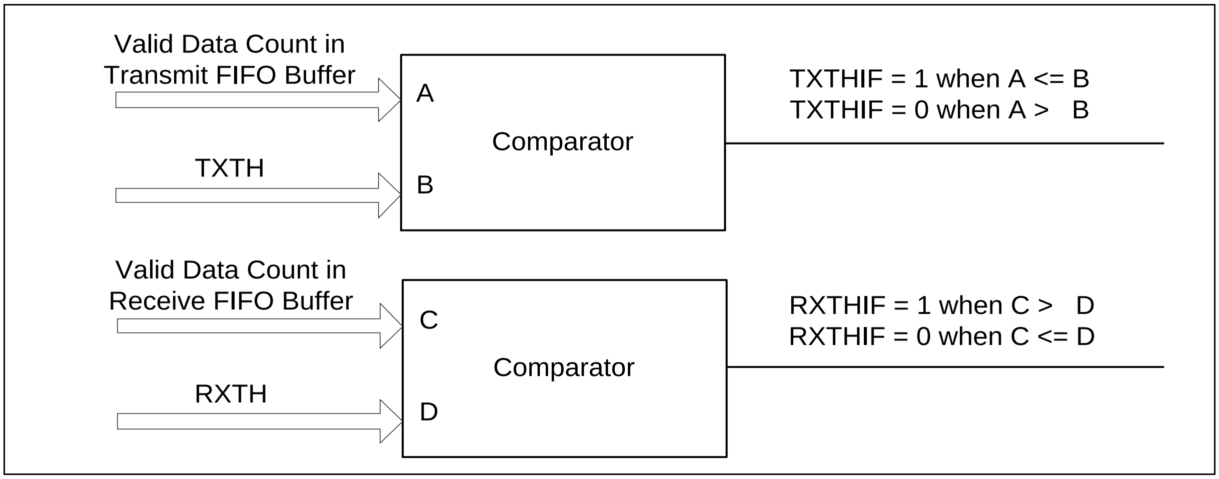


Figure 6.18-18 FIFO Threshold Comparator

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 APB 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 6.17-13, 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.

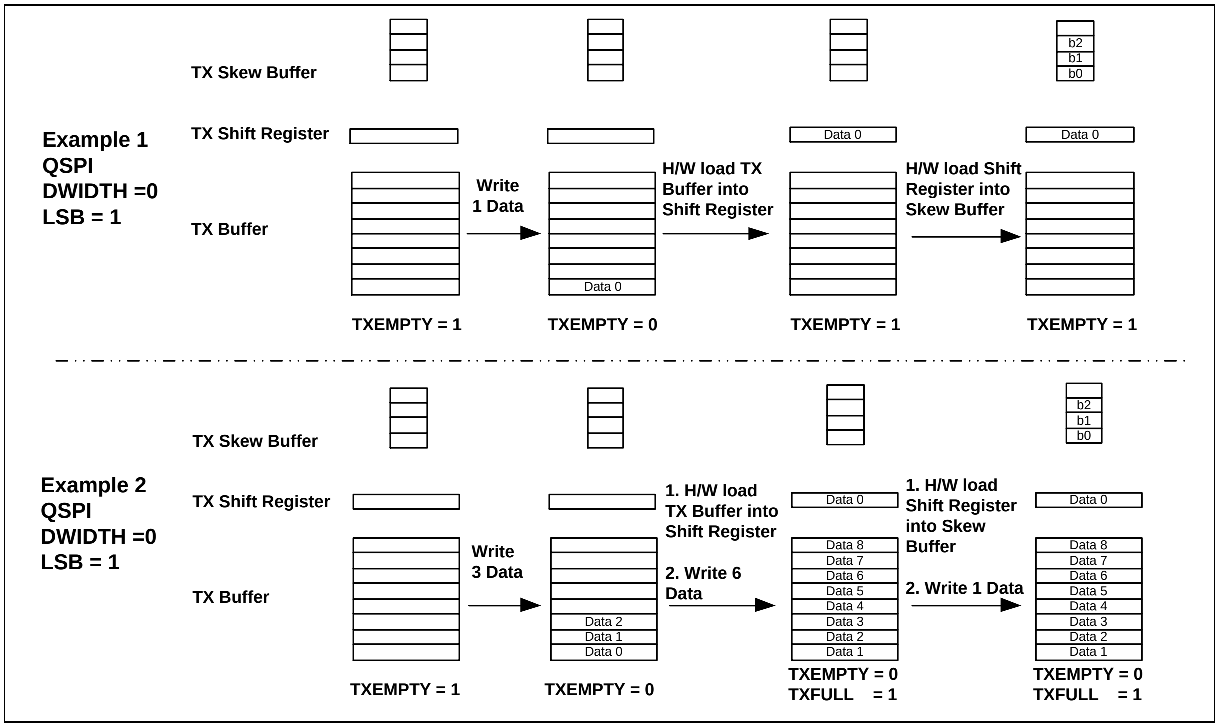


Figure 6.18-19 Transmit FIFO Buffer Example

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).

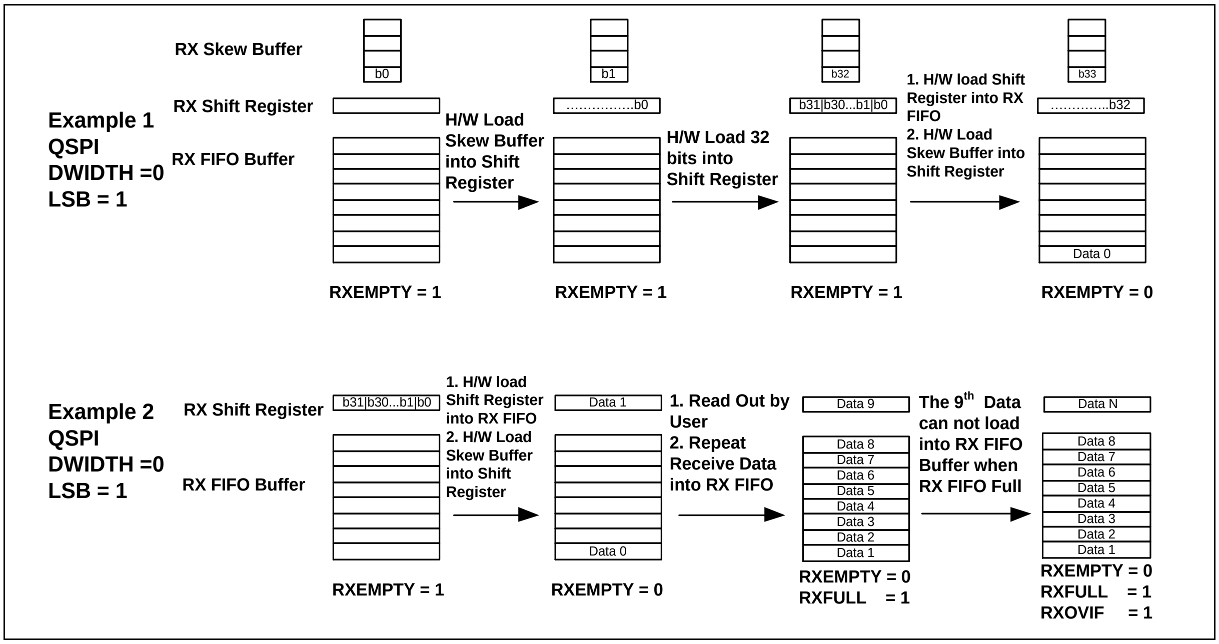


Figure 6.18-20 Receive FIFO Buffer Example

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_SS0 goes to inactive state.

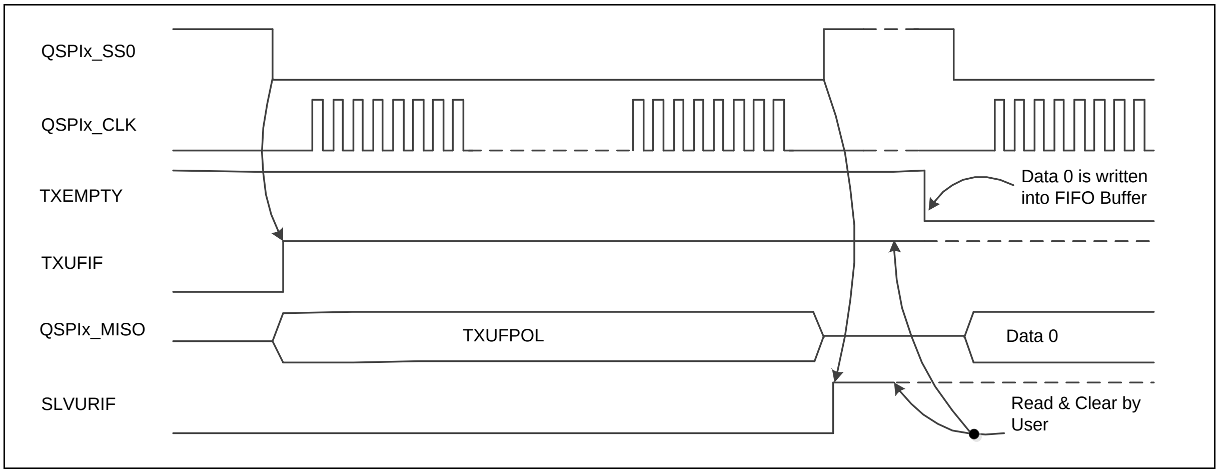


Figure 6.18-21 TX Underflow Event and Slave Under Run Event

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.

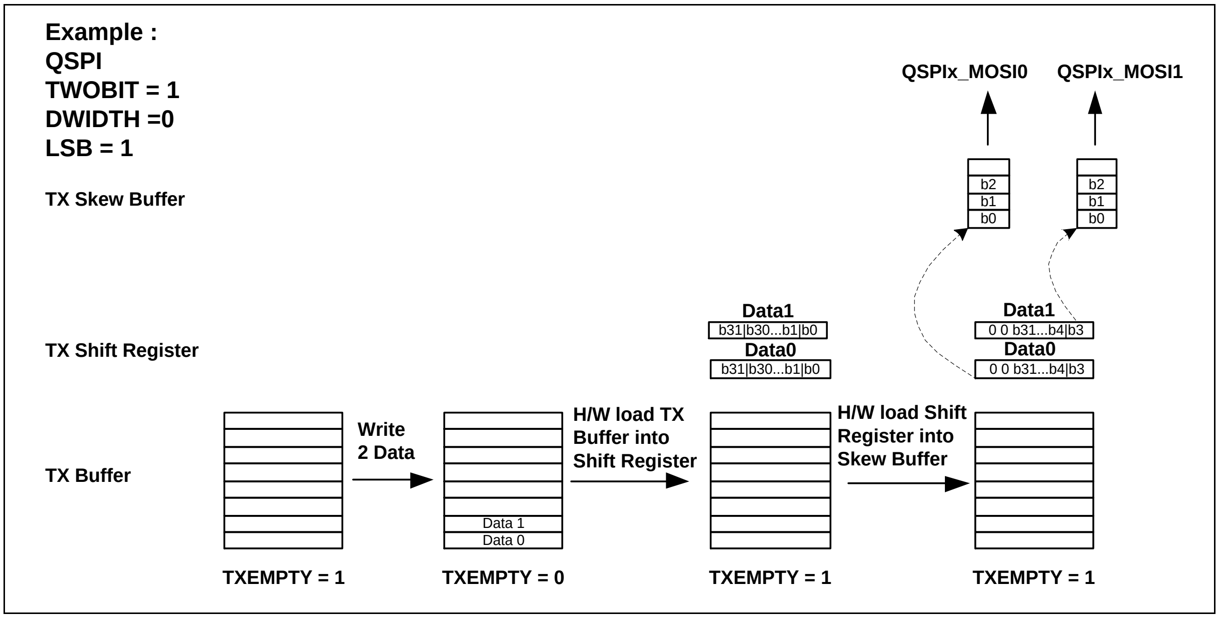


Figure 6.18-22 Two-bit Transfer Mode FIFO Buffer Example

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.

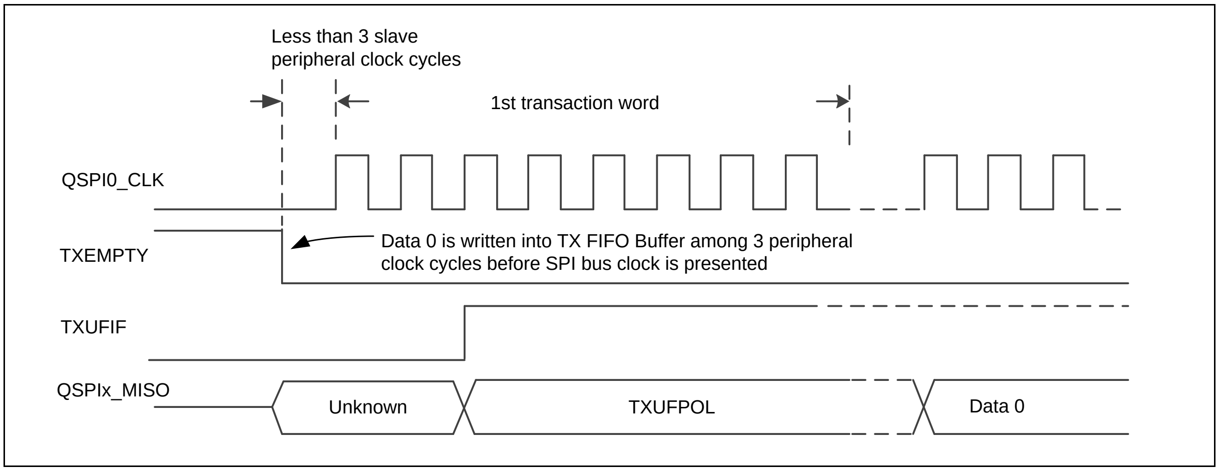


Figure 6.18-23 TX Underflow Event (QSPI0 Slave 3-Wire Mode Enabled)

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.

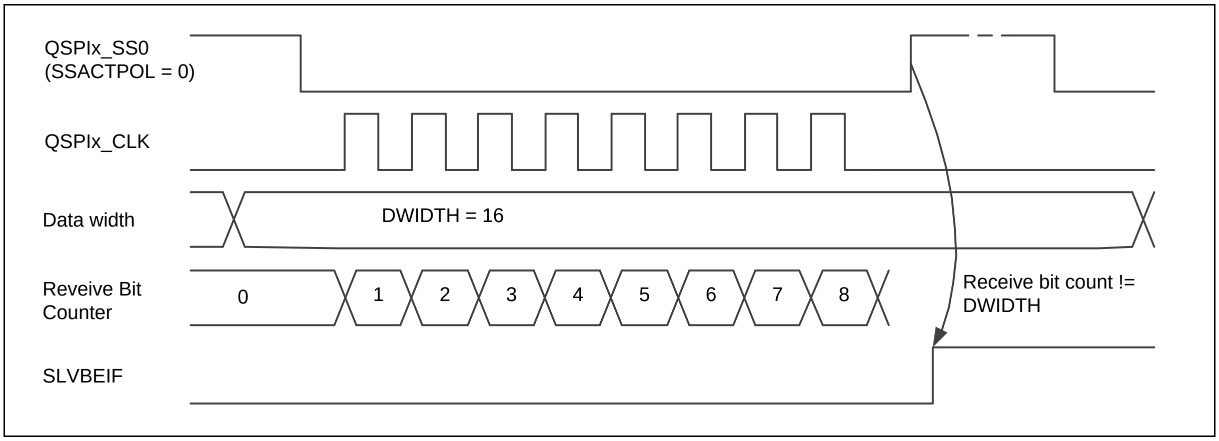


Figure 6.18-24 Slave Mode Bit Count Error

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.

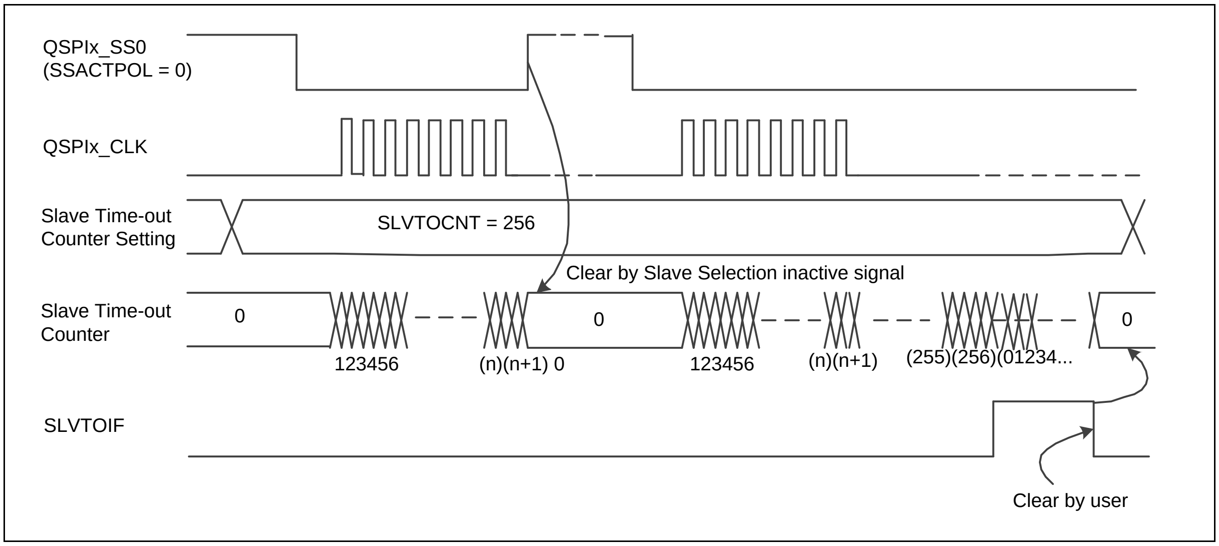


Figure 6.18-25 Slave Time-out Event

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 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.

6.18.5.12 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_SS0 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 is 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.

6.18.6 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.

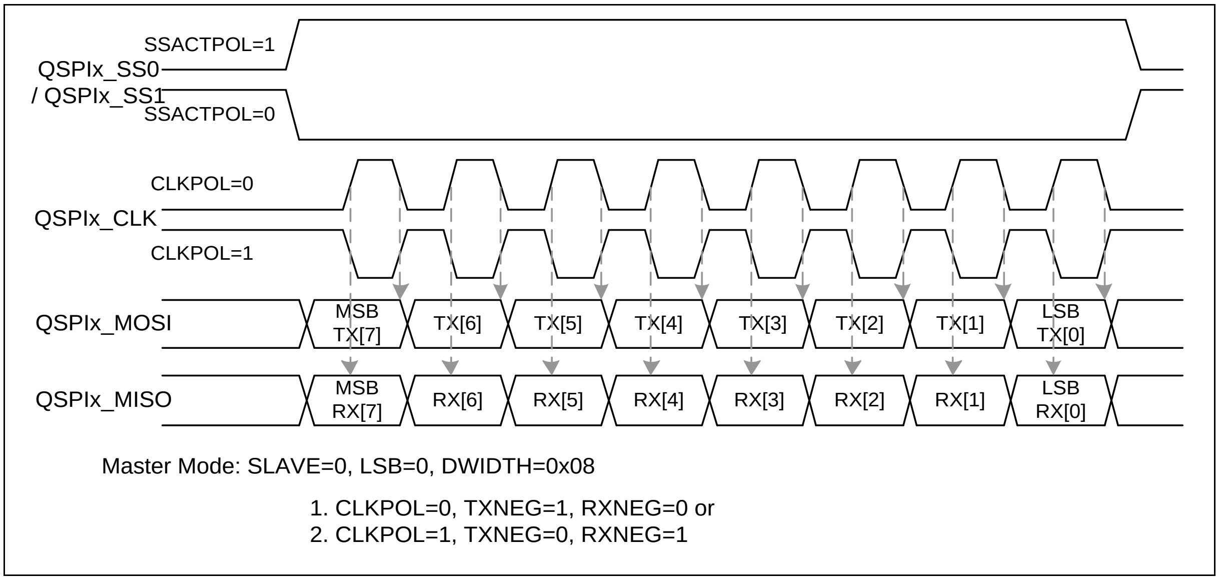


Figure 6.18-26 QSPI Timing in Master Mode

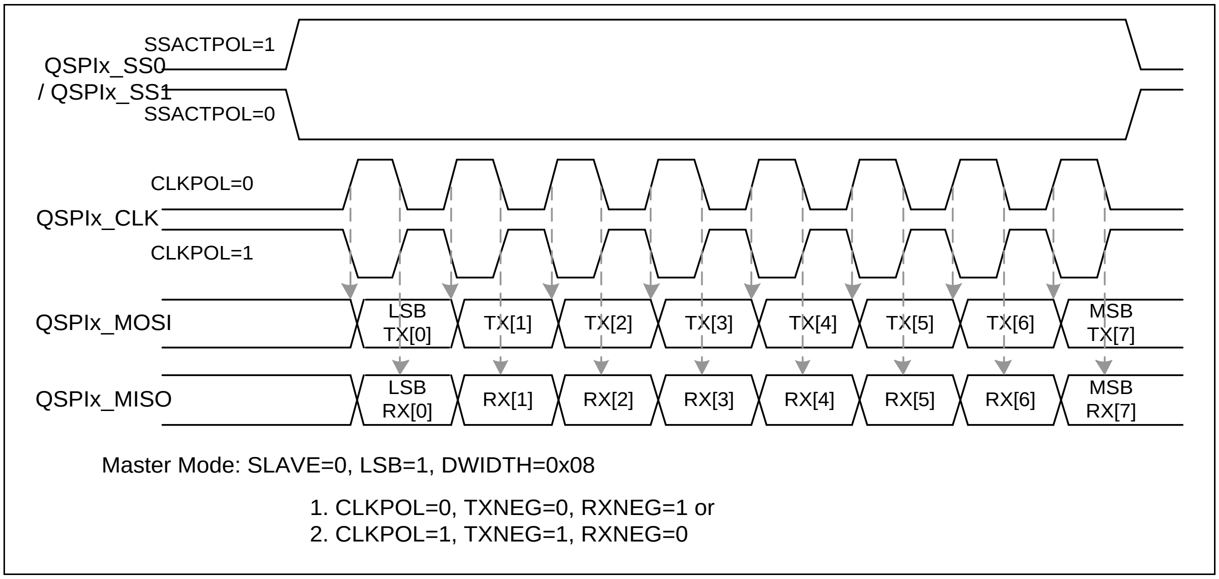


Figure 6.18-27 QSPI Timing in Master Mode (Alternate Phase of QSPIx_CLK)

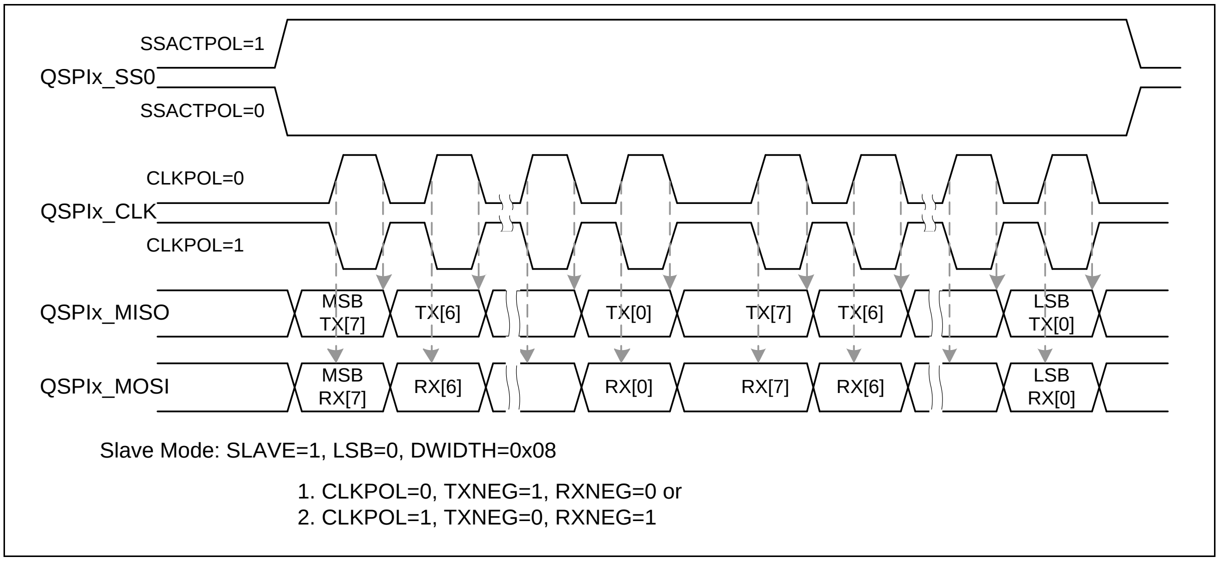


Figure 6.18-28 QSPI Timing in Slave Mode

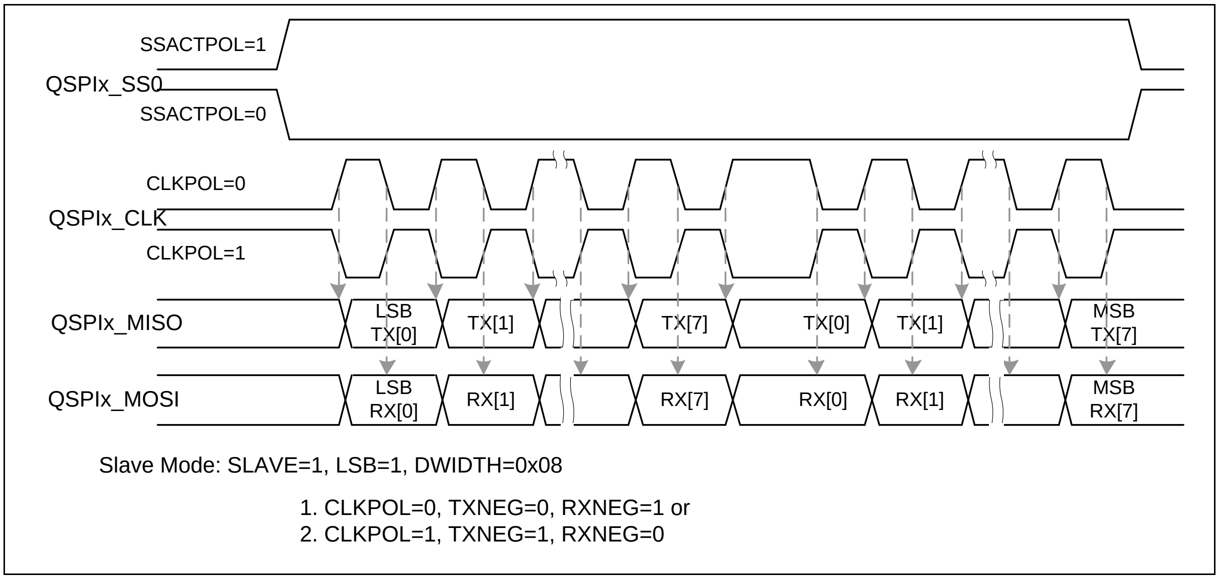


Figure 6.18-29 QSPI Timing in Slave Mode (Alternate Phase of QSPIx_CLK)

6.18.7 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 SS0 (QSPIx_SSCTL[0]) or SS1 (QSPIx_SSCTL[1]) 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 SS0 (QSPIx_SSCTL[0]) or SS1 (QSPIx_SSCTL[1]) 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.

6.18.8 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 = 0xB006_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

6.18.9 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 SPI 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[3:0] + 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_{spi_eclk}, and the QSPI bus clock of QSPI 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_DIVCTL2. **Note:** The time interval must be larger than or equal 5 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.

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	SS1	SS0

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 SPIx_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 SS0 (QSPIx_SSCTL[0]) or SS1 (QSPIx_SSCTL[1]). 1 = Automatic slave selection function Enabled.
[2]	**SSACTPOL**	**Slave Selection Active Polarity** This bit defines the active polarity of slave selection signal (QSPIx_SS0 and QSPIx_SS1). 0 = The slave selection signal QSPIx_SS0/QSPIx_SS1 is active low. 1 = The slave selection signal QSPIx_SS0/QSPIx_SS1 is active high.
[1]	**SS1**	**Slave Selection 1 Control (Master Only)** If AUTOSS bit is cleared to 0, 0 = set the QSPIx_SS1 line to inactive state. 1 = set the QSPIx_SS1 line to active state. If the AUTOSS bit is set to 1, 0 = Keep the QSPIx_SS1 line at inactive state. 1 = QSPIx_SS1 line will be automatically driven to active state for the duration of data transfer, and will be driven to inactive state for the rest of the time. The active state of QSPIx_SS1 is specified in SSACTPOL (QSPIx_SSCTL[2]).
[0]	**SS0**	**Slave Selection 0 Control (Master Only)** If AUTOSS bit is cleared to 0, 0 = set the QSPIx_SS0 line to inactive state. 1 = set the QSPIx_SS0 line to active state. If the AUTOSS bit is set to 1, 0 = Keep the QSPIx_SS0 line at inactive state. 1 = QSPIx_SS0 line will be automatically driven to active state for the duration of data transfer, and will be driven to inactive state for the rest of the time. The active state of QSPIx_SS0 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. 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**	**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 (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 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 (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, as 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: f. QSPIx_CTL[0] = 1 and TXEMPTY = 0. g. For QSPI Master mode, QSPIx_CTL[0] = 1 and TXEMPTY = 1 but the current transaction is not finished yet. h. For QSPI Master mode, QSPIx_CTL[0] = 1 and RXONLY = 1. i. For QSPI Slave mode, the QSPIx_CTL[0] = 1 and there is serial clock input into the QSPI core logic when slave select is active. **j.** For QSPI Slave mode, the 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]) in QSPI mode. In QSPI mode, 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.

6.19 I^{2}S Controller (I^{2}S)

6.19.1 Overview

The I^{2}S controller consists of I^{2}S and PCM protocols to interface with external audio CODEC. The I^{2}S and PCM interface supports 8, 16, 18, 20 and 24-bit left/right precision in record and playback. When operating in 18/20/24-bit precision, each left/right-channel sample is stored in a 32-bit word. Each left/right-channel sample has 24/20/18 MSB bits of valid data and other LSB bits are the padding zeros. When operating in 16-bit precision, right-channel sample is stored in MSB of a 32-bit word and left-channel sample is stored in LSB of a 32-bit word.

The following are the property of the DMA.

- When 16-bit precision, the DMA always 8-beat incrementing burst (FIFO_TH = 0) or 4-beat incrementing burst (FIFO_TH = 1).
- When 24/20/18-bit precision, the DMA always 16-beat incrementing burst (FIFO_TH = 0) or 8-beat incrementing burst (FIFO_TH = 1).
- Always bus lock when 4-beat or 8-beat or 16-beat incrementing burst.
- When reach eighth, quarter, middle and end address of destination address, a DMA_IRQ is triggered to CPU automatically.

An AHB master port and an AHB slave port are offered in I^{2}S controller.

6.19.2 Features

- Supports I^{2}S interface record and playback
  - Left/right channel
  - 8, 16, 20, 24-bit data precision
  - Mater and slave mode
- Supports PCM interface record and playback
  - Two slots
  - 8, 16, 20, 24-bit data precision
  - Master mode
- Use DMA to playback and record data, with interrupt
- Supports two addresses for left/right channel data and different slots

6.19.3 Block Diagram

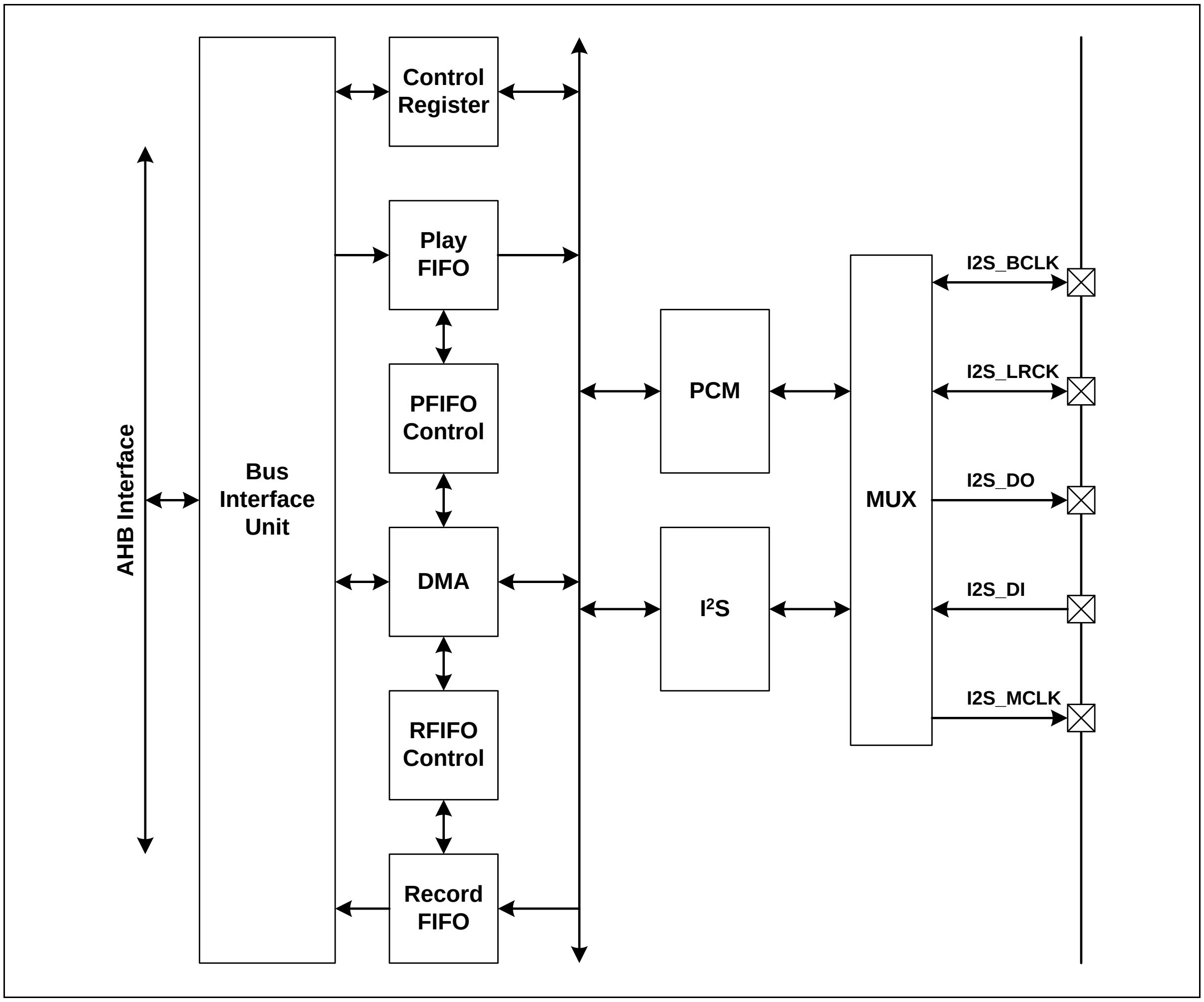


Figure 6.19-1 I^{2}S Controller Block Diagram

6.19.4 Basic Configuration

6.19.4.1 I^{2}S Basic Configuration

- Clock source Configuration
  - Enable I^{2}S peripheral clock in CLK_HCLKEN[24].
- Reset Configuration
  - Reset I^{2}S controller in SYS_AHBIPRST[8].
- Pin Configuration

Group	Pin Name	GPIO	MFP
I^{2}S	I2S_BCLK	PA.3	MFP2
		PB.4	MFP3
		PG.10	MFP8

	I2S_LRCK	PA.2	MFP2
		PB.6	MFP3
		PA.15	MFP8
	I2S_DO	PA.5	MFP2
		PB.5	MFP3
	I2S_DI	PA.4	MFP2
		PB.7	MFP3
	I2S_MCLK	PA.6	MFP2
		PB.1	MFP3

6.19.5 Functional Description

6.19.5.1 I^{2}S Interface

The I^{2}S interface signals are shown as Figure 6.19-2 and Figure 6.19-3.

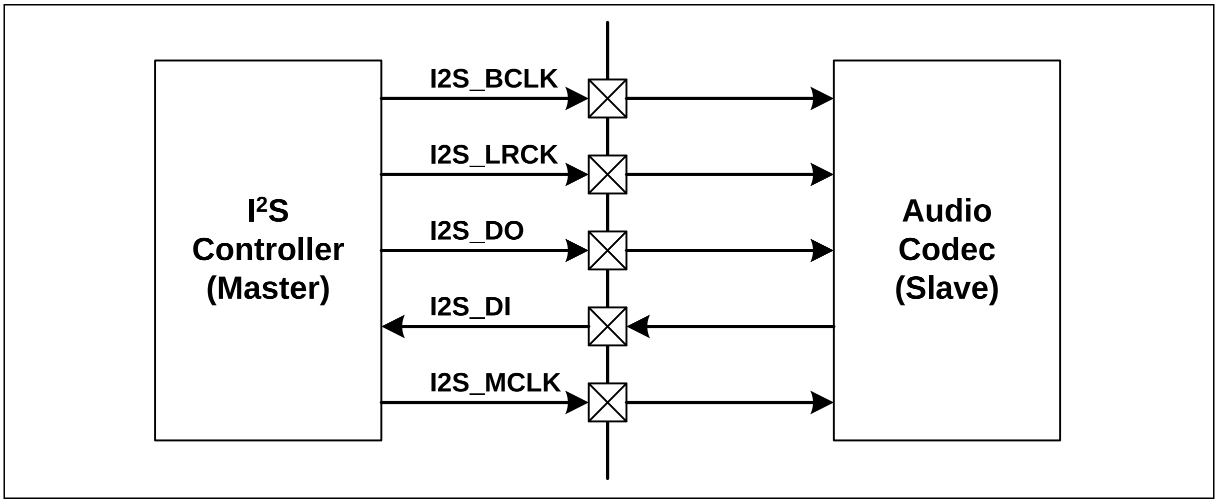


Figure 6.19-2 I^{2}S Interface Signal of Master Mode

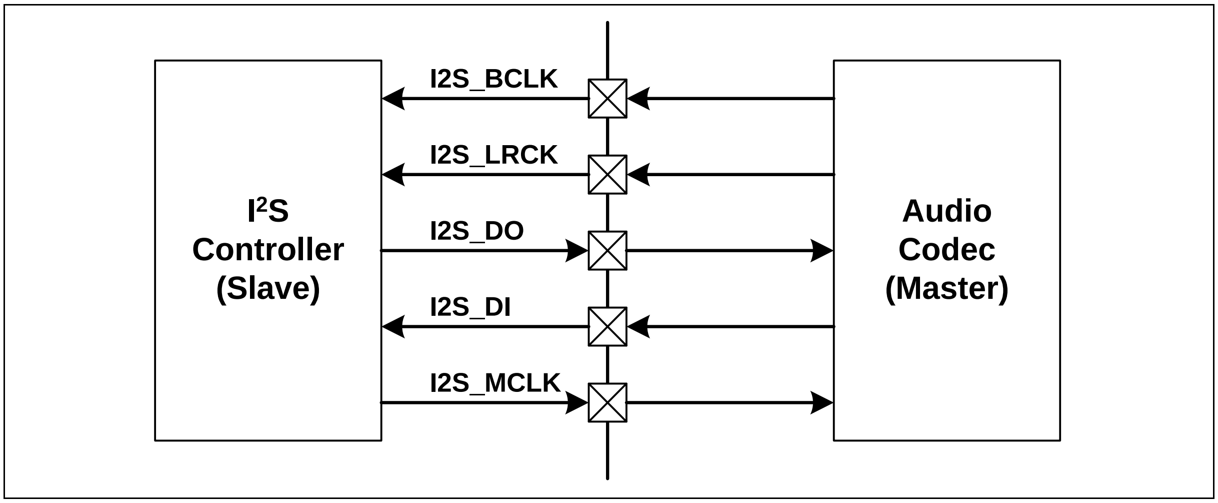


Figure 6.19-3 I^{2}S Interface Signal of Slave Mode

The I^{2}S and MSB-justified format are supported; the timing diagram is shown as Figure 6.19-4.

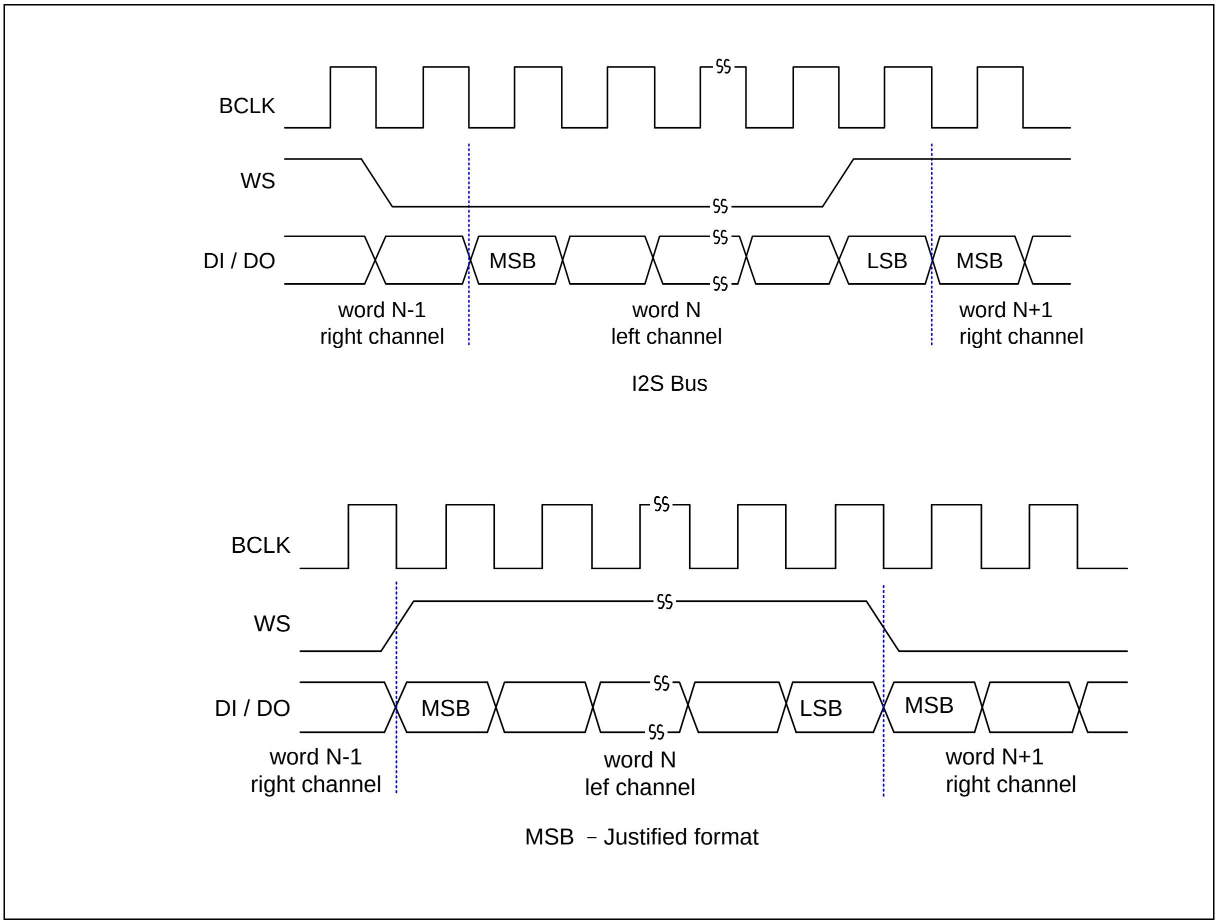


Figure 6.19-4 I^{2}S MSB-Justified Format

The sampling rate, bit shift clock frequency could be set by the control register I2S_CON.

6.19.5.2 PCM Interface

The PCM interface signals are shown as Figure 6.19-5 and Figure 6.19-6.

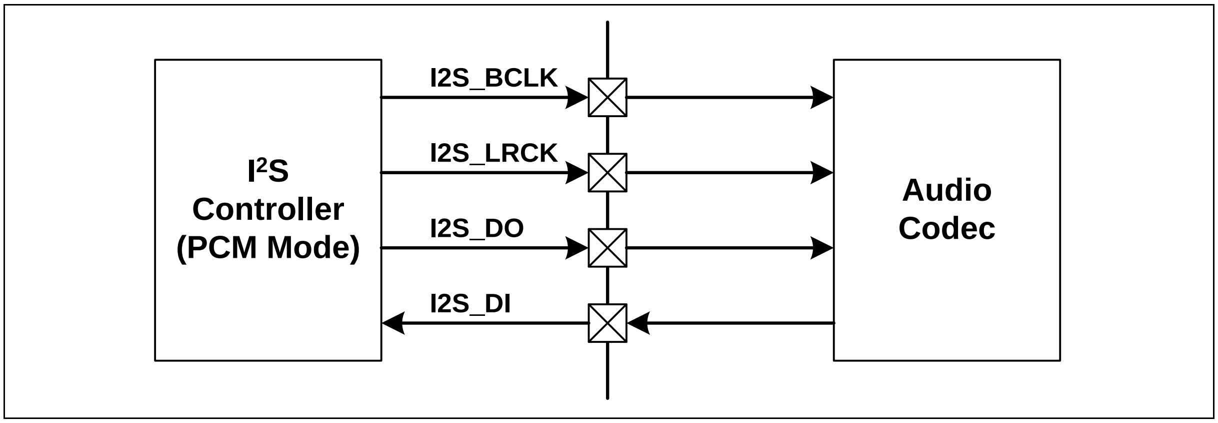


Figure 6.19-5 PCM Mode Signal Interface

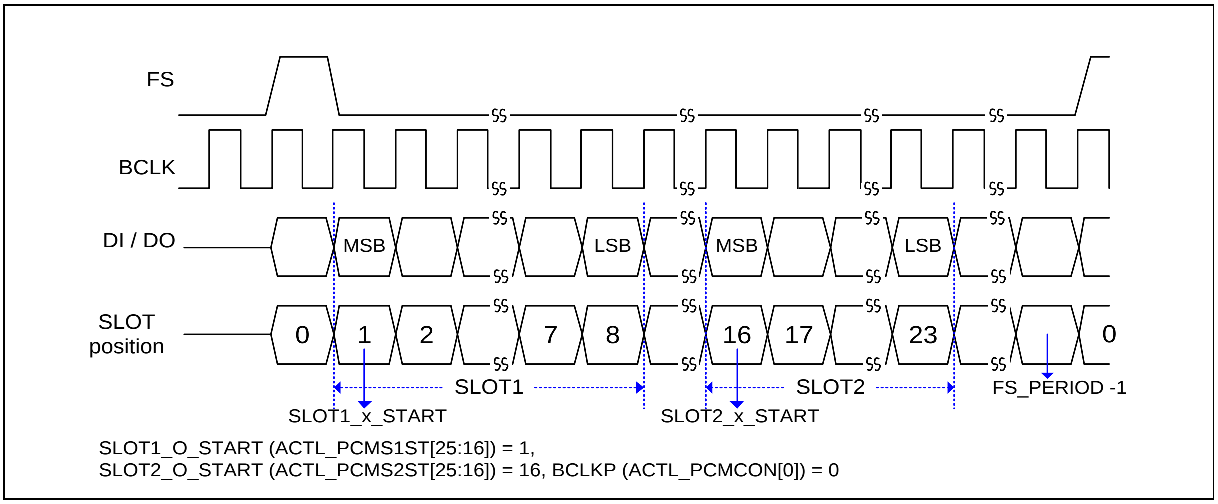

FS
BCLK
DI / DO
MSB
LSB
MSB
LSB
SLOT
position
0
1
2
7
8
16
17
23
0
SLOT1
SLOT2
FS_PERIOD -1
SLOT1_x_START
SLOT2_x_START
SLOT1_O_START (ACTL_PCMS1ST[25:16]) = 1,
SLOT2_O_START (ACTL_PCMS2ST[25:16]) = 16, BCLKP (ACTL_PCMCON[0]) = 0

Figure 6.19-6 PCM Mode Interface Waveform

6.19.5.3 Split Left/Right and Slot1/Slot2 Data

For SPLIT=0, 8bit-data, L=8bit left/slot0 data, R=8bit right/slot1 data, and address at I2S_RDESB / I2S_PDESB.

0xC	0x8	0x4	0x0
R,L,R,L	R,L,R,L	R,L,R,L	R,L,R,L

For SPLIT=0, 16bit-data, L=16bit left/slot0 data, R=16bit right/slot1 data, and address at I2S_RDESB / I2S_PDESB.

0xC	0x8	0x4	0x0
R,L	R,L	R,L	R,L

For SPLIT=0, 24bit-data, L=24bit left/slot0 data, R=24bit right/slot1 data, and address at I2S_RDESB / I2S_PDESB.

0xC	0x8	0x4	0x0
R	L	R	L

For SPLIT=1, 8bit-data, L=8bit left/slot0 data, R=8bit right/slot1 data, and address at I2S_RDESB / I2S_PDESB.

0xC	0x8	0x4	0x0
L,L,L,L	L,L,L,L	L,L,L,L	L,L,L,L

For SPLIT=1, 8bit-data, L=8bit left/slot0 data, R=8bit right/slot1 data, and address at I2S_RDESB2 / I2S_PDESB2.

0xC	0x8	0x4	0x0
R,R,R,R	R,R,R,R	R,R,R,R	R,R,R,R

For SPLIT=1, 16bit-data, L=16bit left/slot0 data, R=16bit right/slot1 data, and address at I2S_RDESB / I2S_PDESB.

0xC	0x8	0x4	0x0
L,L	L,L	L,L	L,L

For SPLIT=1, 16bit-data, L=16bit left/slot0 data, R=16bit right/slot1 data, and address at I2S_RDESB2 / I2S_PDESB2.

0xC	0x8	0x4	0x0
R,R	R,R	R,R	R,R

For SPLIT=1, 24bit-data, L=24bit left/slot0 data, R=24bit right/slot1 data, and address at I2S_RDESB / I2S_PDESB.

0xC	**0x8**	**0x4**	**0x0**
L	L	L	L

For SPLIT=1, 24bit-data, L=24bit left/slot0 data, R=24bit right/slot1 data, and address at I2S_RDESB2 / I2S_PDESB2.

0xC	**0x8**	**0x4**	**0x0**
R	R	R	R

6.19.6 Register Map

R: read only, **W**: write only, **R/W**: both read and write, **C**: Only value 1 can be written

Register	Offset	R/W	Description	Reset Value
I²S Base Address: **I2S_BA = 0xB002_0000**				
I2S_GLBCON	I2S_BA+0x000	R/W	I²S Global Control Register	0x0000_0000
I2S_RESET	I2S_BA+0x004	R/W	I²S Sub Block Reset Control Register	0x0000_0000
I2S_RDESB	I2S_BA+0x008	R/W	I²S Record DMA Destination Base Address Register	0x0000_0000
I2S_RDES_LENGTH	I2S_BA+0x00C	R/W	I²S Record DMA Destination Length Register	0x0000_0000
I2S_RDESC	I2S_BA+0x010	R	I²S Record DMA Destination Current Address Register	0x0000_0000
I2S_PDESB	I2S_BA+0x014	R/W	I²S Play DMA Destination Base Address Register	0x0000_0000
I2S_PDES_LENGTH	I2S_BA+0x018	R/W	I²S Play DMA Destination Length Register	0x0000_0000
I2S_PDESC	I2S_BA+0x01C	R	I²S Play DMA Destination Current Address Register	0x0000_0000
I2S_RSR	I2S_BA+0x020	R/W	I²S Record Status Register	0x0000_0000
I2S_PSR	I2S_BA+0x024	R/W	I²S Play Status Register	0x0000_0000
I2S_CON	I2S_BA+0x028	R/W	I²S Control Register	0x0000_0000
I2S_COUNTER	I2S_BA+0x02C	R/W	I²S Play DMA Down Counter Register	0xFFFF_FFFF
I2S_PCMCON	I2S_BA+0x030	R/W	I²S PCM Mode Control Register	0x0000_0000
I2S_PCMS1ST	I2S_BA+0x034	R/W	I²S PCM Mode Slot 1 Start Register	0x0000_0000
I2S_PCMS2ST	I2S_BA+0x038	R/W	I²S PCM Mode Slot 2 Start Register	0x0000_0000
I2S_RDESB2	I2S_BA+0x040	R/W	I²S Record DMA Destination Base Address 2 Register	0x0000_0000
I2S_PDESB2	I2S_BA+0x044	R/W	I²S Play DMA Destination Base Address 2 Register	0x0000_0000

6.19.7 Register Description

I^{2}S Global Control Register (I2S_GLBCON)

Register	Offset	R/W	Description	Reset Value
I2S_GLBCON	I2S_BA+0x000	R/W	I^{2}S Global Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved		R_DMA_IRQ_EN	P_DMA_IRQ_EN	R_FIFO_FULL_IRQ_EN	R_FIFO_EMPTY_IRQ_EN	P_FIFO_FULL_IRQ_EN	P_FIFO_EMPTY_IRQ_EN
15	14	13	12	11	10	9	8
R_DMA_IRQ_SEL		P_DMA_IRQ_SEL		R_DMA_IRQ	P_DMA_IRQ	BITS_SELECT	
7	6	5	4	3	2	1	0
FIFO_TH	Reserved		IRQ_DMA_CNTER_EN	IRQ_DMA_DATA_ZERO_EN	Reserved	BLOCK_EN	

Bits	Description	
[31:20]	**Reserved**	Reserved.
[21]	**R_DMA_IRQ_EN**	**Record DMA Interrupt Request Enable Bit** 0 = R_DMA_IRQ Disabled. 1 = R_DMA_IRQ Enabled.
[20]	**P_DMA_IRQ_EN**	**Playback DMA Interrupt Request Enable Bit** 0 = P_DMA_IRQ Disabled. 1 = P_DMA_IRQ Enabled.
[19]	**R_FIFO_FULL_IRQ_EN**	**Record FIFO Full Interrupt Request Enable Bit** 0 = R_FIFO_FULL_IRQ Disabled. 1 = R_FIFO_FULL_IRQ Enabled.
[18]	**R_FIFO_EMPTY_IRQ_EN**	**Record FIFO Empty Interrupt Request Enable Bit** 0 = R_FIFO_EMPTY_IRQ Disabled. 1 = R_FIFO_EMPTY_IRQ Enabled.
[17]	**P_FIFO_FULL_IRQ_EN**	**Playback FIFO Full Interrupt Request Enable Bit** 0 = P_FIFO_FULL_IRQ Disabled. 1 = P_FIFO_FULL_IRQ Enabled.
[16]	**P_FIFO_EMPTY_IRQ_EN**	**Playback FIFO Empty Interrupt Request Enable Bit** 0 = P_FIFO_EMPTY_IRQ Disabled. 1 = P_FIFO_EMPTY_IRQ Enabled.

[15:14]	**R_DMA_IRQ_SEL**	**Record DMA Interrupt Request Selection Bits** 00 = When record DMA address reach DMA record destination **end** address, the R_DMA_RIA_IRQ will be issued. 01 = When record DMA address reach each **half** of DMA record destination end address, the R_DMA_RIA_IRQ will be issued. 10 = When record DMA address reach each **quarter** of DMA record destination end address, the R_DMA_RIA_IRQ will be issued. 11 = When record DMA address reach each **eighth** of DMA record destination end address, the R_DMA_RIA_IRQ will be issued.
[13:12]	**P_DMA_IRQ_SEL**	**Play DMA Interrupt Request Selection Bits** 00 = When play DMA address reach DMA play destination **end** address, the P_DMA_RIA_IRQ will be issued. 01 = When play DMA address reach each **half** of DMA play destination **end** address, the P_DMA_RIA_IRQ will be issued. 10 = When play DMA address reach each **quarter** of DMA play destination end address, the P_DMA_RIA_IRQ will be issued. 11 = When play DMA address reach each **eighth** of DMA play destination end address, the P_DMA_RIA_IRQ will be issued.
[11]	**R_DMA_IRQ**	**Record DMA Interrupt Request Bit** When R_DMA_RIA_IRQ or R_FIFO_FULL or R_FIFO_EMPTY is set to "1" in record and these corresponding interrupt enable bits are set to "1", the R_DMA_IRQ bit will be set to 1 automatically, and this bit could be cleared to 0 by CPU writing "1". The bit is hardwired to A™ rm96 as interrupt request signal with an inverter.
[10]	**P_DMA_IRQ**	**Playback DMA Interrupt Request Bit** When P_DMA_RIA_IRQ (I2S_PSR[0]) or DMA_DATA_ZERO_IRQ (I2S_PSR[3]) or DMA_CNTER_IRQ (I2S_PSR[4]) or P_FIFO_FULL (I2S_PSR[2]) or P_FIFO_EMPTY (I2S_PSR[1]) is set to 1 in playback and these corresponding interrupt enable bits are set to "1", the bit P_DMA_IRQ will be set to 1, and this bit could be clear to 0 by CPU writing "1". And the bit is hardwired to Arm9™ 6 as interrupt request signal with an inverter.
[9:8]	**BITS_SELECT**	**BITS Selection** 00 = Data format is 8-bits of a channel. 01 = Data format is 16-bits of a channel. 10 = Data format is 24-bits of a channel. 11 = Reserve.
[7]	**FIFO_TH**	**FIFO Threshold Control Bit** 0 = The FIFO threshold is 8 levels. 1 = The FIFO threshold is 4 levels.
[6:5]	**Reserved**	Reserved.
[4]	**IRQ_DMA_CNTER_EN**	**IRQ_DMA Counter Function Enable Bit** 0 = Not allowed to set P_DMA_IRQ (I2S_GLBCON[10]) if I2S_PSR[4] is set to 1. 1 = Allowed to set P_DMA_IRQ (I2S_GLBCON[10]) if (I2S_PSR[4]) is set to 1.
[3]	**IRQ_DMA_DATA_ZERO_EN**	**IRQ_DMA_DATA Zero and Sign Detect Enable Bit** 0 = Not allowed to set P_DMA_IRQ (I2S_GLBCON[10]) if I2S_PSR[3] is set to 1. 1 = Allowed to set P_DMA_IRQ (I2S_GLBCON[10]) if I2S_PSR[3] is set to 1.
[2]	**Reserved**	Reserved.

[1:0]	**BLOCK_EN**	**Block Function Enable Register** 00 = The I^{2}S & PCM interface Disabled. 01 = The I^{2}S interface Enabled. 10 = The PCM interface Enabled. 11 = Reserved.

I^{2}S Sub Block Reset Control Register (I2S_RESET)

Register	Offset	R/W	Description	Reset Value
I2S_RESET	I2S_BA+0x004	R/W	I^{2}S Sub Block Reset Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved			**SPLIT_DATA**	**Reserved**			**RESET**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RECORD_SINGLE		**PLAY_SINGLE**		**Reserved**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**RECORD**	**PLAY**	**DMA_CNTER EN**	**DMA_DATA_ZERO_EN**	**Reserved**		**BLOCK_RESET**

Bits	Description	
[31:17]	**Reserved**	Reserved.
[20]	**SPLIT_DATA**	**SPLIT Left/Right and Slot1/Slot2 Data** 0 = Left/Right channel data place at I2S_RDESB / I2S_PDESB address. 1 = Left channel data at I2S_RDESB / I2S_PDESB address, Right channel data at I2S_RDESB2 / I2S_PDESB2. For details about the split data, refer to section 6.19.5.3. **Note:** This bit works in stereo/dual slot mode only.
[19:17]	**Reserved**	Reserved.
[16]	**RESET**	**Audio Controller Reset Control Bit** 0 = The audio controller is normal operation. 1 = The whole audio controller is reset.
[15:14]	**RECORD_SINGLE**	**Record Single/Dual Channel Select Bits** 00 = Reserved. 01 = The record only selects I²S left channel/pcm slot0. 10 = The record only selects I²S right channel/pcm slot1. 11 = The record is dual channel.
[13:12]	**PLAY_SINGLE**	**Playback Single/Dual Channel Select Bits** **In I^{2}S mode,** 00 = Reserved. 01 = Reserved. 10 = The playback is **mono** mode. 11 = The playback is **stereo** mode. **In PCM mode,** 00 = The playback is dual slot with slot1 data. 01 = The playback is dual slot with slot0 data.

		10 = The playback is mono data. slot0 only. 11 = The playback is dual slot.
[11:7]	**Reserved**	Reserved.
[6]	**RECORD**	**I^{2}S/PCM Record Enable Bit** 0 = The record path of I^{2}S/PCM Disabled. 1 = The record path of I^{2}S/PCM Enabled.
[5]	**PLAY**	**I^{2}S/PCM Playback Enable Bit** 0 = The playback path of I^{2}S/PCM Disabled. 1 = The playback path of I^{2}S/PCM Enabled.
[4]	**DMA_CNTER_EN**	**DMA Counter Function Enable Bit** This function is supported to count playback data for software monitoring. When one playback data is transferred to codec, the DMA counter subtracts 1. When the I2S_COUNTER [31:0] register is Zero that set DMA_CNTER_IRQ bit =1. 0 = The DMA counter function Disabled. 1 = The DMA counter function Enabled.
[3]	**DMA_DATA_ZERO_EN**	**DMA_DATA Zero and Sign Detect Enable Bit** 0 = The DMA_DATA zero and sign detect function Disabled. 1 = The DMA_DATA zero and sign detect function Enabled.
[2:1]	**Reserved**	Reserved.
[0]	**BLOCK_RESET**	**I^{2}S/PCM RESET Control Bit** 0 = Release the I^{2}S/PCM function block from reset mode. 1 = Force the I^{2}S/PCM function block to reset mode.

I²S Record DMA Destination Base Address Register (I2S_RDESB)

Register	Offset	R/W	Description	Reset Value
I2S_RDESB	I2S_BA+0x008	R/W	I²S Record DMA Destination Base Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**AUDIO_RDESB**	**32-bit Record Destination Base Address** This bit field indicates the record destination base address of DMA.

I²S Record DMA Destination Length Register (I2S_RDES_LENGTH)

Register	Offset	R/W	Description	Reset Value
I2S_RDES_LENGTH	I2S_BA+0x00C	R/W	I²S Record DMA Destination Length Register	0x0000_0000

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

Bits	Description	
[31:0]	**AUDIO_RDES_L**	**32-bit Record Destination Address Length** The AUDIO_RDES_L [31:0] bits are read/write. The minimum values for 8-bits, 16-bits, 18-bits, 20-bits, and 24-bits modes are 0x40.

I^{2}S Record DMA Destination Current Address Register (I2S_RDESC)

Register	Offset	R/W	Description	Reset Value
I2S_RDESC	I2S_BA+0x010	R	I^{2}S Record DMA Destination Current Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**AUDIO_RDESC**	**32-bit Record Destination Current Address (Read Only)** This bit field indicates the current address of DMA record destination.

I^{2}S Play DMA Destination Base Address Register (I2S_PDESB)

Register	Offset	R/W	Description	Reset Value
I2S_PDESB	I2S_BA+0x014	R/W	I^{2}S Play DMA Destination Base Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**AUDIO_PDESB**	**32-bit Play Destination Base Address** This bit field indicates the play destination base address of DMA.

I^{2}S Play DMA Destination Length Register (I2S_PDES_LENGTH)

Register	Offset	R/W	Description	Reset Value
I2S_PDES_LENGTH	I2S_BA+0x018	R/W	I^{2}S Play DMA Destination Length Register	0x0000_0000

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

Bits	Description	
[31:0]	**AUDIO_PDES_L**	**32-bit Play Destination Address Length** The minimum values for 8-bits, 16-bits, 18-bits, 20-bits, and 24-bits modes are 0x40.

I^{2}S Play DMA Destination Current Address Register (I2S_PDESC)

Register	Offset	R/W	Description	Reset Value
I2S_PDESC	I2S_BA+0x01C	R	I^{2}S Play DMA Destination Current Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**AUDIO_PDESC**	**32-bit Play Destination Current Address (Read Only)** This bit field indicates the current address of DMA play destination.

I^{2}S Record Status Register (I2S_RSR)

Register	Offset	R/W	Description	Reset Value
I2S_RSR	I2S_BA+0x020	R/W	I^{2}S Record 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**
R_DMA_RIA_SN			**Reserved**		**R_FIFO_FULL**	**R_FIFO_EMPTY**	**R_DMA_RIA_IRQ**

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:5]	**R_DMA_RIA_SN**	**Record DMA Reach Indicative Address Section Number Bit (Read Only)** R_DMA_IRQ_SEL (I2S_GLBCON[15:14]) = 01, **R_DMA_RIA_SN[2:0]= 1, 0.** R_DMA_IRQ_SEL (I2S_GLBCON[15:14]) = 10, **R_DMA_RIA_SN[2:0]= 1, 2, 3, 0.** R_DMA_IRQ_SEL (I2S_GLBCON[15:14]) = 11, **R_DMA_RIA_SN[2:0]= 1, 2, 3, 4, 5, 6, 7, 0.**
[4:3]	**Reserved**	Reserved.
[2]	**R_FIFO_FULL**	**Record FIFO Full Indicator Bit** When record FIFO is full and the record data is written into record FIFO, the R_FIFO_FULL bit is set to 1. This bit indicates the full error of record FIFO has happened. 0 = The full error of record FIFO not happened. 1 = The full error of record FIFO happened. **Note:** This bit is readable, and can only be cleared by writing “1” to it.
[1]	**R_FIFO_EMPTY**	**Record FIFO EMPTY Indicator Bit** When record FIFO is empty and the record data is read from record FIFO, the R_FIFO_EMPTY bit is set to 1. This bit indicates the empty error of record FIFO happened. 0 = The empty error of record FIFO not happened. 1 = The empty error of record FIFO happened. **Note:** This bit is readable, and can only be cleared by writing “1” to it.
[0]	**R_DMA_RIA_IRQ**	**Record DMA Reach Indicative Address Interrupt Request Bit** 0 = Record DMA address does not reach the indicative address by R_DMA_IRQ_SEL (I2S_GLBCON[15:14]). 1 = Record the DMA address reach the indicative address by R_DMA_IRQ_SEL (I2S_GLBCON[15:14]). **Note:** This bit is readable, and can only be cleared by writing “1” to it.

I^2S Play Status Register (I2S_PSR)

Register	Offset	R/W	Description	Reset Value
I2S_PSR	I2S_BA+0x024	R/W	I^2S Play 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**
P_DMA_RIA_SN			**DMA_CNTER_IRQ**	**DMA_DATA_ZERO_IRQ**	**P_FIFO_FULL**	**P_FIFO_EMPTY**	**P_DMA_RIA_IRQ**

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:5]	**P_DMA_RIA_SN**	**Play DMA Reach Indicative Address Section Number Bit (Read Only)** P_DMA_IRQ_SEL (I2S_GLBCON[13:12]) = 01, **P_DMA_RIA_SN[2:0]= 1, 0.** P_DMA_IRQ_SEL (I2S_GLBCON[13:12]) = 10, **P_DMA_RIA_SN[2:0]= 1, 2, 3, 0.** P_DMA_IRQ_SEL (I2S_GLBCON[13:12]) = 11, **P_DMA_RIA_SN[2:0]= 1, 2, 3, 4, 5, 6, 7, 0.**
[4]	**DMA_CNTER_IRQ**	**DMA Counter IRQ** When one playback data is transferred to codec, the DMA counter subtracts 1. The counting of playback data number is used for software monitoring. If the DMA counter I2S_COUNTER [31:0] is Zero, this bit DMA_CNTER_IRQ will be set to 1. 0 = DMA counter I2S_COUNTER [31:0] has not counted to zero. 1 = DMA counter I2S_COUNTER [31:0] has counted down to zero. **Note:** This bit is readable, and can only be cleared by writing “1” to it.
[3]	**DMA_DATA_ZERO_IRQ**	**DMA_DATA Zero IRQ** 0 = Not found the all data bit of playback DMA is zero or its sign bit does not change (two channels). 1 = Found all the data bit of playback DMA is zero or its sign bit change (two channels). **Note:** This bit is readable, and can only be cleared by writing “1” to it.
[2]	**P_FIFO_FULL**	**Playback FIFO Full Indicator Bit** When playback FIFO is full and the playback data is written into playback FIFO, the P_FIFO_FULL bit is set to 1. This bit indicates the full error of playback FIFO has happened. 0 = The full error of playback FIFO not happened. 1 = The full error of playback FIFO happened. **Note:** This bit is readable, and can only be cleared by writing “1” to it.

[1]	**P_FIFO_EMPTY**	**Playback FIFO EMPTY Indicator Bit** When playback FIFO is empty and the playback data is read from playback FIFO, the P_FIFO_EMPTY bit is set to 1. This bit indicates the empty error of playback FIFO has happened. 0 = The empty error of playback FIFO not happened. 1 = The empty error of playback FIFO happened. **Note:** This bit is readable, and can only be cleared by writing "1" to it.
[0]	**P_DMA_RIA_IRQ**	**Playback DMA Reach Indicative Address Interrupt Request Bit** 0 = Playback DMA address does not reach the specific address by P_DMA_IRQ_SEL (I2S_GLBCON[13:12]) bits. 1 = Playback DMA address reach the indicative address by P_DMA_IRQ_SEL (I2S_GLBCON[13:12]) bits. **Note:** This bit is readable, and can only be cleared by writing "1" to it.

I^{2}S Control Register (I2S_CON)

Register	Offset	R/W	Description	Reset Value
I2S_CON	I2S_BA+0x028	R/W	I^{2}S Control Register	0x0000_0000

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

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20]	**SLAVE**	**I^{2}S Slave Mode Selection Bit** 0 = I^{2}S Master mode. 1 = I^{2}S Slave mode.
[19:16]	**PRS**	**I^{2}S Frequency PRE_SCALER Selection Bits (FPLL Is the Input PLL Frequency, MCLK Is the Output Main Clock)** 0000 = MCLK=FPLL/1. 0001 = MCLK=FPLL/2. 0010 = MCLK=FPLL/3. 0011 = MCLK=FPLL/4. 0100 = MCLK=FPLL/5. 0101 = MCLK=FPLL/6. 0110 = MCLK=FPLL/7. 0111 = MCLK=FPLL/8. 1000 = RESERVED. 1001 = MCLK=FPLL/10. 1010 = RESERVED. 1011 = MCLK=FPLL/12. 1100 = RESERVED. 1101 = MCLK=FPLL/14. 1110 = RESERVED. 1111 = MCLK=FPLL/16. **Note:** When the division factor is 3/5/7, the duty cycle of MCLK is not 50%, the high duration is 0.5*FPLL.
[15:8]	**Reserved**	Reserved.
[7:5]	**BCLK_DIV**	**I^{2}S Serial Data Clock Frequency Selection Bit**

		This bit field is used to decide the relationship of frequency between PLL and I^{2}S serial data clock. The frequency of I^{2}S serial data clock follows the formula below: $frequency(\mathrm{BCLK}) = \frac{frequency(\mathrm{PLL})}{(1+\mathrm{PRS}[3:0])} \times \frac{1}{(\mathrm{BCLK\_DIV}+1)\times 2}$
[4]	**MCLK_SEL**	**MCLK Clock Selection Bit** 0 = I^{2}S MCLK output will follow the PRS [3:0] setting. 1 = I^{2}S MCLK output will be the same with the input frequency of PLL.
[3]	**FORMAT**	**I^{2}S Format Selection Bit** 0 = I^{2}S compatible format is selected. 1 = MSB-justified format is selected.
[2:0]	**Reserved**	Reserved.

I^{2}S Play DMA Down Counter Register (I2S_COUNTER)

Register	Offset	R/W	Description	Reset Value
I2S_COUNTER	I2S_BA+0x02C	R/W	I^{2}S Play DMA Down Counter Register	0xFFFF_FFFF

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

Bits	Description	
[31:0]	**COUNTER**	**Play DMA Down Counter** This bit field is used to count playback data number for software monitoring. When one playback data is transferred to codec, the DMA counter subtracts 1. If the I2S_COUNTER [31:0] register is Zero, the DMA_CNTER_IRQ (I2S_PSR[4]) will be set to 1.

I^{2}S PCM Mode Control Register (I2S_PCMCON)

Register	Offset	R/W	Description	Reset Value
I2S_PCMCON	I2S_BA+0x030	R/W	I^{2}S PCM Mode Control Register	0x0000_0000

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

Bits	Description	
[31:28]	**PCM_MCLK_PRS**	**PCM MCLK Frequency PRE_SCALER Selection Bits (FPLL Is the Input PLL Frequency, MCLK Is the Output Main Clock)** 0000 = PCM_MCLK=FPLL/1. 0001 = PCM_MCLK=FPLL/2. 0010 = PCM_MCLK=FPLL/3. 0011 = PCM_MCLK=FPLL/4. 0100 = PCM_MCLK=FPLL/5. 0101 = PCM_MCLK=FPLL/6. 0110 = PCM_MCLK=FPLL/7. 0111 = PCM_MCLK=FPLL/8. 1000 = Reserved. 1001 = PCM_MCLK=FPLL/10. 1010 = Reserved. 1011 = PCM_MCLK=FPLL/12. 1100 = Reserved. 1101 = PCM_MCLK=FPLL/14. 1110 = Reserved. 1111 = PCM_MCLK=FPLL/16. **Note:** When the division factor is 3/5/7, the duty cycle of MCLK is not 50%, the high duration is 0.5*FPLL.
[27:26]	**Reserved**	Reserved.
[25:16]	**FS_PERIOD**	**FS Pulse Period** BCLK counts between two FS pulse. Use this bit to set sample rate.
[15:8]	**PCM_PRS**	**PCM_BCLK Frequency PRE_SCALER Selection Bits** For BCLK of PCM mode, this control register setting and PCM BCLK frequency

		are described as follows. **(1) Case1 : When PCM_MCLK_PRS (I2S_PCMCON[31:28]) = 0.** BCLK frequency = PCM_MCLK frequency / (2*(PCM_PRS + 1)). Where PCM_PRS = 0,1,2,3,4,…. **(2) Case2 : When PCM_MCLK_PRS (I2S_PCMCON[31:28])≠0** BCLK frequency = PCM_MCLK frequency / (2*n), where n=1,2,3,4,…. PCM_PRS = (PCM_MCLK_PRS + 1)*n – 1, where the value of n equals to the value of n, which is shown in BCLK frequency formula above.
[7:1]	**Reserved**	Reserved.
[0]	**BCLKP**	**BCLK Polarity** 0 = Send data at rising edge, latch data at falling edge. 1 = Send data at falling edge, latch data at rising edge.

I^{2}S PCM Mode Slot 1 Start Register (I2S_PCMS1ST)

Register	Offset	R/W	Description	Reset Value
I2S_PCMS1ST	I2S_BA+0x034	R/W	I^{2}S PCM Mode Slot 1 Start Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						**SLOT1_O_START**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SLOT1_O_START							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**SLOT1_I_START**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SLOT1_I_START							

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25:16]	**SLOT1_O_START**	**Slot 1 Data Out Start Position** This bit field is used to set the start position of slot1 output data. Example: For Short Frame Sync, set SLOT1_O_START to 1. For Long Frame Sync, set SLOT1_O_START to 0.
[15:10]	**Reserved**	Reserved.
[9:0]	**SLOT1_I_START**	**Slot 1 Data in Start Position** This bit field is used to set the start position of slot1 input data. Example: For Short Frame Sync, set SLOT1_I_START to 1. For Long Frame Sync, set SLOT1_I_START to 0.

I^{2}S PCM Mode Slot 2 Start Register (I2S_PCMS2ST)

Register	Offset	R/W	Description	Reset Value
I2S_PCMS2ST	I2S_BA+0x038	R/W	I^{2}S PCM Mode Slot 2 Start Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						SLOT2_O_START	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SLOT2_O_START							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						SLOT2_I_START	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SLOT2_I_START							

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25:16]	**SLOT2_O_START**	**Slot 2 Data Out Start Position** This bit field is used to set the start position of slot2 output data.
[15:10]	**Reserved**	Reserved.
[9:0]	**SLOT2_I_START**	**Slot 2 Data in Start Position** This bit field is used to set the start position of slot2 input data.

I²S Record DMA Destination Base Address 2 Register (I2S_RDESB2)

Register	Offset	R/W	Description	Reset Value
I2S_RDESB2	I2S_BA+0x040	R/W	I²S Record DMA Destination Base Address 2 Register	0x0000_0000

31	30	29	28	27	26	25	24
AUDIO_RDESB2							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
AUDIO_RDESB2							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
AUDIO_RDESB2							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
AUDIO_RDESB2							

Bits	Description	
[31:0]	**AUDIO_RDESB2**	**32-bit Record Destination Base Address for Right Channel** This bit field indicates the record destination base address of DMA.

I^{2}S Play DMA Destination Base Address 2 Register (I2S_PDESB2)

Register	Offset	R/W	Description	Reset Value
I2S_PDESB2	I2S_BA+0x044	R/W	I^{2}S Play DMA Destination Base Address 2 Register	0x0000_0000

31	30	29	28	27	26	25	24
AUDIO_PDESB2							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
AUDIO_PDESB2							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
AUDIO_PDESB2							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
AUDIO_PDESB2							

Bits	Description	
[31:0]	**AUDIO_PDESB2**	**32-bit Play Destination Base Address for Right Channel** This bit field indicates the play destination base address of DMA.

6.20 Ethernet MAC Controller (EMAC)

6.20.1 Overview

This chip provides 2 Ethernet MAC Controller (EMAC) for Network application.

The Ethernet MAC controller consists of IEEE 802.3/Ethernet protocol engine with internal CAM function for recognizing Ethernet MAC addresses; Transmit-FIFO, Receive-FIFO, TX/RX state machine controller, time stamping engine for IEEE 1588, Magic Packet parsing engine and status controller.

The EMAC supports RMII (Reduced MII) interface to connect with external Ethernet PHY.

6.20.2 Features

- Supports IEEE Std. 802.3 CSMA/CD protocol
- Supports Ethernet frame time stamping for IEEE Std. 1588 – 2002 protocol
- Supports both half and full duplex for 10 Mbps or 100 Mbps operation
- Supports RMII interface
- Supports MII Management function to control external Ethernet PHY
- Supports pause and remote pause function for flow control
- Supports long frame (more than 1518 bytes) and short frame (less than 64 bytes) reception
- Supports 16 entries CAM function for Ethernet MAC address recognition
- Supports Magic Packet recognition to wake system up from power-down mode
- Supports 256 bytes transmit FIFO and 256 bytes receive FIFO
- Supports DMA function

6.20.3 Block Diagram

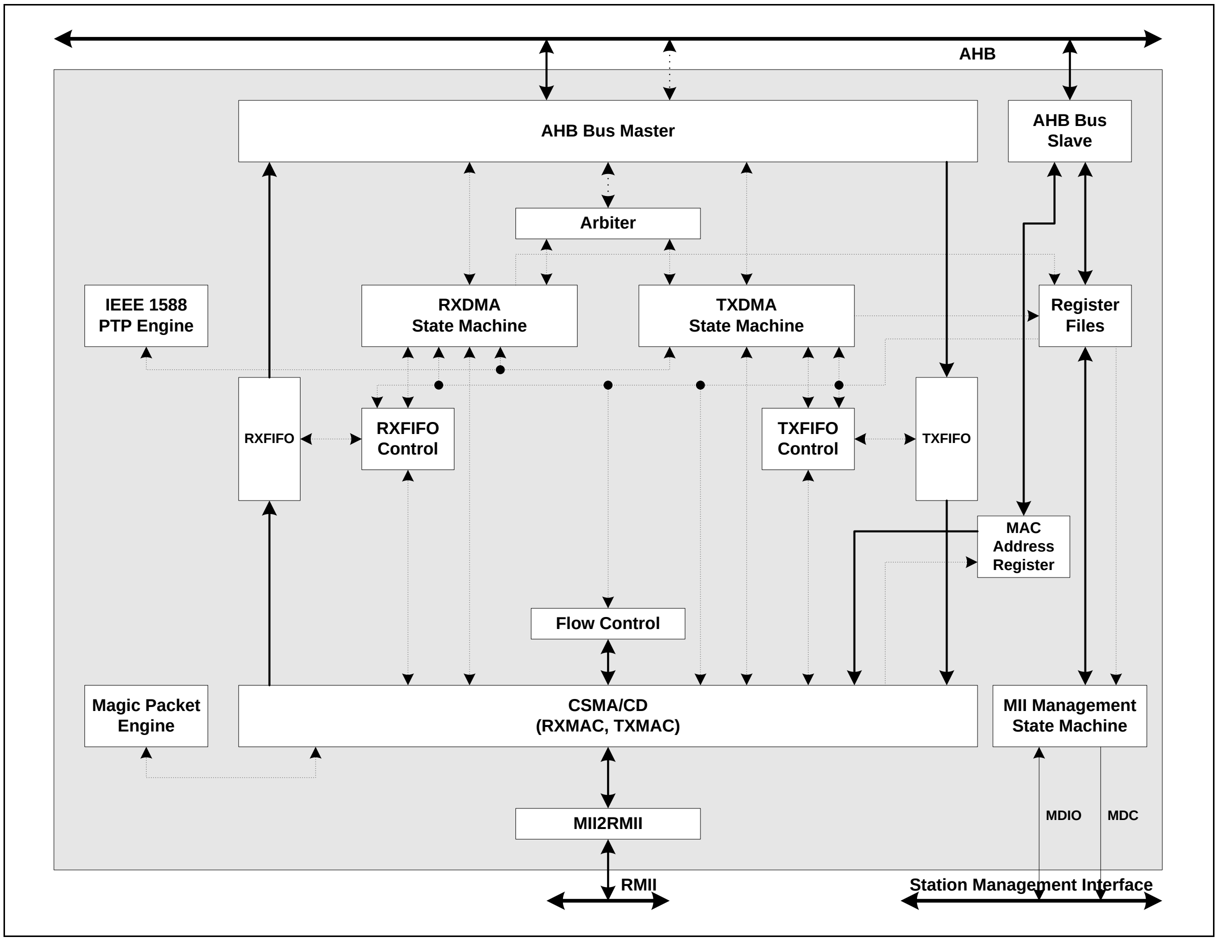


Figure 6.20-1 Ethernet MAC Controller Block Diagram

6.20.4 Basic Configuration

6.20.4.1 EMAC0 Basic Configuration

- Clock source Configuration
  - Enable EMAC0 clock in EMAC0 (CLK_HCLKEN[16]).
- Reset Configuration
  - Reset EMAC0 controller in EMAC0RST (SYS_AHBIPRST[16]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
RMII0	RMII0_RXERR	PE.0	MFP1
	RMII0_CRSDV	PE.1	MFP1
	RMII0_RXD1	PE.2	MFP1
	RMII0_RXD0	PE.3	MFP1

	RMII0_REFCLK	PE.4	MFP1
	RMII0_TXEN	PE.5	MFP1
	RMII0_TXD1	PE.6	MFP1
	RMII0_TXD0	PE.7	MFP1
	RMII0_MDIO	PE.8	MFP1
	RMII0_MDC	PE.9	MFP1

Table 6.20-1 EMAC0 Pin Configuration

6.20.4.2 EMAC1 Basic Configuration

- Clock source Configuration
  - Enable EMAC1 clock in EMAC1 (CLK_HCLKEN[17]).
- Reset Configuration
  - Reset EMAC1 controller in EMAC1RST (SYS_AHBIPRST[17]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
RMII1	RMII1_RXERR	PF.0	MFP1
	RMII1_CRSDV	PF.1	MFP1
	RMII1_RXD1	PF.2	MFP1
	RMII1_RXD0	PF.3	MFP1
	RMII1_REFCLK	PF.4	MFP1
	RMII1_TXEN	PF.5	MFP1
	RMII1_TXD1	PF.6	MFP1
	RMII1_TXD0	PF.7	MFP1
	RMII1_MDIO	PF.8	MFP1
	RMII1_MDC	PF.9	MFP1

Table 6.20-2 EMAC1 Pin Configuration

6.20.5 Functional Description

6.20.5.1 Arbiter

In the EMAC, there are two different bus requests, RXREQ and TXREQ respectively. Arbiter does the arbitration between the RXREQ and TXREQ, and then decides which one can request the AHB bus. The arbitration results are shown as Table 6.20-3:

RXREQ	TXREQ	Granted
0	0	Neither TXDMA nor RXDMA granted.
0	1	TXDMA granted.
1	0	RXDMA granted.

1	1	If TXFIFO valid data byte count is less than RXFIFO free space byte count, TXDMA granted.
1	1	If RXFIFO free space byte count is less than or equal to TXFIFO valid space byte count, RXDAM granted.

Table 6.20-3 Arbiter Arbitration Results

6.20.5.2 TXDMA State Machine

The TXDMA state machine transfers data from the system memory to the internal 256 bytes transmit FIFO through the AHB master. Then, the TXDMA state machine will request the transmit MAC to send the data out. During the transmission process, the TXDMA will fetch the transmit descriptor first. Through the buffer address field of the transmit descriptor, the TXDMA fetch the frame data from the system and store it into the internal 256 bytes transmit FIFO. Then, the transmit MAC will read frame data from the transmit FIFO and send the frame out. After the finish of the frame transmission, the TXDMA updates the transmit status of current frame and write the transmit descriptor back to the system memory to indicate the frame transmission has finished.

6.20.5.3 RXDMA State Machine

The RXDMA state machine transfers data from the internal 256 bytes receiving FIFO to the system memory through AHB master. During the receiving process, the RXDMA will fetch the received descriptor first. Through the buffer address field of the received descriptor, the RXDMA will know memory space which is allocated to store the incoming frame. After the received MAC indicates there is a new incoming frame, the RXDMA starts to transfer the frame data from the internal received FIFO to the system memory. After the receiving process has finished, the RXDMA will update the receiving status of current frame and write the received descriptor back to system memory to indicate a new incoming frame is in the system memory.

6.20.5.4 Flow Control

This block implements the flow control function while EMAC operates in the full duplex mode. The flow control function is defined in the IEEE 802.3 Std. chapter 31. The type of flow control frame defined in the IEEE 802.3 Std. is only the PAUSE frame at the moment. The control frame transmission and reception is programmable through the control registers.

To receive a control frame, software must set the bit ACP (Accept Control Packet) of register EMACn_MCMDR (MAC Command Register). While a PAUSE frame is received, the flow control function will pause the transmission process after the current transmitting frame has been transmitted out.

To transmit a control frame out, software must program the destination MAC address of control frame into the register pair {EMACn_CAM13M, EMACn_CAM13L}, source MAC address into the register pair {EMACn_CAM14M, EMACn_CAM14L}, and configure length/type, op-code and operand of control frame into the register pair {EMACn_CAM15M, EMACn_CAM15L}, and then set the bit SDPZ (EMACn_MCMDR[16]). The bit SDPZ will be cleared while the control frame has been transmitted out.

6.20.5.5 MII Management State Machine

The MII management function of EMAC is compliant to IEEE 802.3 Std. Through the MII management interface, software can access the control and status registers of the external PHY chip. Two programmable registers EMACn_MIID (MAC MII Management Data Register) and EMACn_MIIDA (MAC MII Management Data Control and Address Register) are for MII management function. Set the bit BUSY (EMACn_MIIDA[17]) will trigger the MII management state machine. After the MII management cycle is finished, the BUSY bit will be cleared automatically.

6.20.5.6 Media Access Control (MAC)

The function of Ethernet MAC fully meets the requirements defined by the IEEE802.3u specification. Figure 6.20-2 describes the frame structure and the operation of the transmission and receiving.

The transmission data frame sent from the transmit DMA will be encapsulated by the MAC before transmitting onto the MII bus. The sent data will be assembled with the preamble, the start frame delimiter (SFD), the frame check sequence and the padding for enforcing those less than 64 bytes to meet the minimum size frame and CRC sequence. The outgoing frame format will be as Figure 6.20-2.

110101010 --- 10101010	10101011	d0	d1	d2	--	dn	Padding	CRC31	CRC30	---	CRC0

Figure 6.20-2 Ethernet Frame Format

As mentioned by the above format, the preamble is a consecutive 7-byte long with the pattern “10101010” and the SFD is a one byte 10101011 data. The padding data will be all 0 value if the sent data frame is less than 64 bytes. The padding disable function specified in the bit P of the transmit descriptor is used to control if the MAC needs to pad data at the end of frame data or not when the transmitted data frame is less than 64 bytes. The padding data will not be appended if the padding disable bit is set to be high. The bits CRC0 ... CRC31 are the 32 bits cyclic redundancy check (CRC) sequence. The CRC encoding is defined by the following polynomial specified by the IEEE802.3. This 32 bits CRC appending function will be disabled if the Inhibit CRC of the transmission descriptor is set to high.

The MAC also performs many other transmission functions specified by the IEEE802.3, including the inter-frame spacing function, collision detection, collision enforcement, collision back off and retransmission. The collision back-off timer is a function of the integer slot time, 512 bit times. The number of slot times to delay between the current transmission attempt to the next attempt is determined by a uniformly distributed random integer algorithm specified by the IEEE802.3. The MAC performs the receive functions specified by the IEEE 802.3 including the address recognition function, the frame check sequence validation, the frame disassembly, framing and collision filtering.

6.20.5.7 Time Stamping Engine for IEEE 1588

The EMAC supports a time stamping engine for IEEE Std. 1588. In this time stamping engine, a 64-bit counter implemented to generate the reference timing, the registers EMACn_TSSEC and ETSLSR.

In frame transmission, if TSEN (EMACn_TSCTL[0]) and TTSEN of TXDES 0 (TXDMA Descriptor Word 0) are both high, EMAC would store the 64-bit reference timing value to TXDES 1 (TXDMA Descriptor Word 1) and TXDES 2 (TXDMA Descriptor Word 2) when frame transmission completed.

In frame reception, if TSEN (EMACn_TSCTL[0]) is high, EMAC would store the 64-bit reference timing value to RXDES 1 (RXDMA Descriptor Word 1) and RXDES 3 (RXDMA Descriptor Word 3) when the frame reception finished.

Figure 6.20-3 describes how the 64-bit counter works to generate the reference timing.

The 64-bit counter formed by two 32-bit counters, the EMACn_TSSEC and EMACn_TSSUBSEC, a updated using the EMAC’s input reference clock, the HCLK. Two difference methods, controlled by TSMODE (EMACn_TSCTL[3]), implemented to increase 32-bit EMACn_TSSUBSEC counter by value configured in register EMACn_TSINC. When TSMODE (EMACn_TSCTL[3]) is low, TSLSR counter increased in every clock. When TSMODE (EMACn_TSCTL[3]) is high, TSLSR counter increased only when accumulator is overflow.

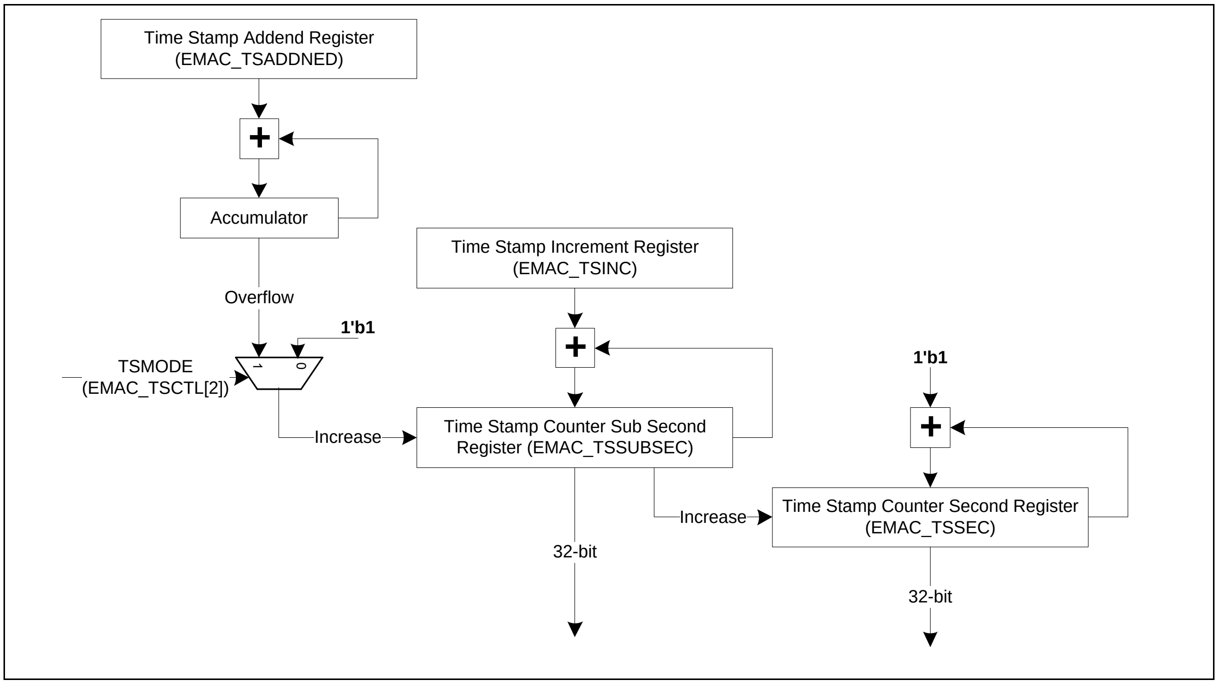


Figure 6.20-364-bit Reference Timing Counter

6.20.5.8 Magic Packet Parsing Engine

The EMAC supports a Magic packet parsing engine for recognizing Magic packet. The Magic packet is a broadcast frame which payload includes 6 bytes of 0xFF, followed by 16 repetitions of 48-bit MAC address defined by registers EMACn_CAM0M and EMACn_CAM0L.

The MGP_WAKE (EMACn_MCMDR[6]) controls if the Magic packet parsing engine enabled. If MGP_WAKE (EMACn_MCMDR[6]) is high, EMAC will set bit MGPR (EMACn_MISTA[15]) high to indicate Magic packet received. At the same time, EMAC generates an event to wake system up from power-down mode. If WOLIEN (EMACn_MIEN[15]) is high, EMAC generates an RX interrupt to CPU simultaneously.

6.20.6 DMA Descriptors Data Structure

A link-list data structure named as descriptor is used to keep the control, status and data information of each frame. Through the descriptor, CPU and EMAC exchange the information for frame reception and transmission.

Two different descriptors defined in EMAC. One named as RXDMA descriptor for frame reception and the other named as TXDMA descriptor for frame transmission. Each RXDMA or TXDMA descriptor consists of four words. The descriptor keeps the much control, status information and the details of descriptor are described below.

6.20.6.1 RXDMA Descriptor Data Structure

The RXDMA descriptor consists of four 32-bit words. The data structure of RXDMA descriptor shown in Figure 6.20-4.

	31		15	0
RXDES 0	OWN	Receive Frame Status	Receive Frame Byte Count	
RXDES 1	Receive Frame Buffer Starting Address / Time Stamp Least Significant 32-Bit			
RXDES 2	Reserved			
RXDES 3	Next RxDMA Descriptor Starting Address / Time Stamp Most Significant 32-Bit			

Figure 6.20-4 RXDMA Descriptor Data Structure

RXDES 0: RXDMA Descriptor Word 0

The RXDMA descriptor word 0 contains a descriptor ownership indicator, receive frame status, and receive frame byte count. The detailed description of RXDES 0 is shown below.

31	30	29	28	27	26	25	24
Owner	Reserved						
23	22	21	20	19	18	17	16
RTSAS	RP	ALIE	RXGD	PTLE	Reserved	CRCE	RXINTR
15	14	13	12	11	10	9	8
RBC							
7	6	5	4	3	2	1	0
RBC							

Bits	Description	
[31]	**Owner**	**Ownership** The ownership field defines which one, the CPU or EMAC, is the owner of each RX descriptor. Only the owner has right to modify the RX descriptor and the others can read the RX descriptor only. If the O=1'b1 indicates the EMAC RXDMA is the owner of RX descriptor and the RX descriptor is available for frame reception. After the frame reception completed, EMAC RXDMA modified ownership field to 1'b0. If the O=1'b0 indicates the CPU is the owner of RX descriptor. After the CPU completed the frame processing, it modified the ownership field to 1'b1 and released the RX descriptor to EMAC RXDMA. 0 = The owner is CPU. 1 = The owner is EMAC.
[30:24]	**Reserved**	Reserved.
[23]	**RTSAS**	**RX Time Stamp Active Status** This bit is to indicate the time stamping circuit stamped this incoming frame successfully. When this bit set high, RX Descriptor Word 1 and RX Descriptor Word 3 keep the time stamp value recorded when this incoming frame is received completely. 0 = RX Descriptor Word 1 and RX Descriptor Word 3 does not keep the time stamp value. 1 = RX Descriptor Word 1 and RX Descriptor Word 3 keep the time stamp value.
[22]	**RP**	**Runt Packet** The RP indicates the frame stored in the data buffer pointed by RX descriptor is a short frame (frame length is less than 64 bytes). 0 = The frame is not a short frame. 1 = The frame is a short frame.
[21]	**ALIE**	**Alignment Error** The ALIE indicates the frame stored in the data buffer pointed by RX descriptor is not a multiple of byte. 0 = The frame is a multiple of byte. 1 = The frame is not a multiple of byte.

[20]	**RXGD**	**Frame Reception Complete** The RXGD indicates the frame reception has completed and stored in the data buffer pointed by RX descriptor. 0 = The frame reception does not complete yet. 1 = The frame reception completed.
[19]	**PTLE**	**Packet Too Long** The PTLE indicates the frame stored in the data buffer pointed by RX descriptor is a long frame (frame length is greater than 1518 bytes). 0 = The frame is not a long frame. 1 = The frame is a long frame.
[18]	**Reserved**	Reserved.
[17]	**CRCE**	**CRC Error** The CRCE indicates the frame stored in the data buffer pointed by RX descriptor incurred CRC error. 0 = The frame does not incur CRC error. 1 = The frame incurred CRC error.
[16]	**RXINTR**	**Receive Interrupt** The RXINTR indicates the frame stored in the data buffer pointed by RX descriptor caused an interrupt condition. 0 = The frame does not cause an interrupt. 1 = The frame caused an interrupt.
[15:0]	**RBC**	**Receive Byte Count** The RBC indicates the byte count of the frame stored in the data buffer pointed by RX descriptor. The four bytes CRC field is also included in the receive byte count. But if the SPCRC (EMACn_MCMDR[5]) is enabled, the four bytes CRC field will be excluded from the receive byte count.

RXDES 1: RXDMA Descriptor Word 1

The RXDMA descriptor word 1 contains the received frame buffer starting address or time stamp least significant 32-bit value. The detailed description of RXDES 1 is shown below.

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

Bits	Description	
[31:0]	**RXBSA**	**Receive Buffer Starting Address** The RXBSA is the buffer starting address to store the received packet.

Bits	Description	
[31:0]	**TSLSB**	**Time Stamp Least Significant 32-bit** If TSEN (EMACn_TSCR[0]) enabled, Ethernet MAC controller would store time stamp least signification 32-bit value, register EMACn_TSLSR, into this field when it writes back RX Descriptor to system memory.

RXDES 2: RXDMA Descriptor Word 2

The RXDMA descriptor word 2 is reserved.

31	30	29	28	27	26	25	24
Reserved							
23	22	21	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	
[31:0]	**Reserved**	Reserved.

RXDES 3: RXDMA Descriptor Word 3

The RXDMA descriptor word 3 contains the next RXDMA descriptor starting address or time stamp most significant 32-bit value. The detailed description of RXDES 3 is shown below.

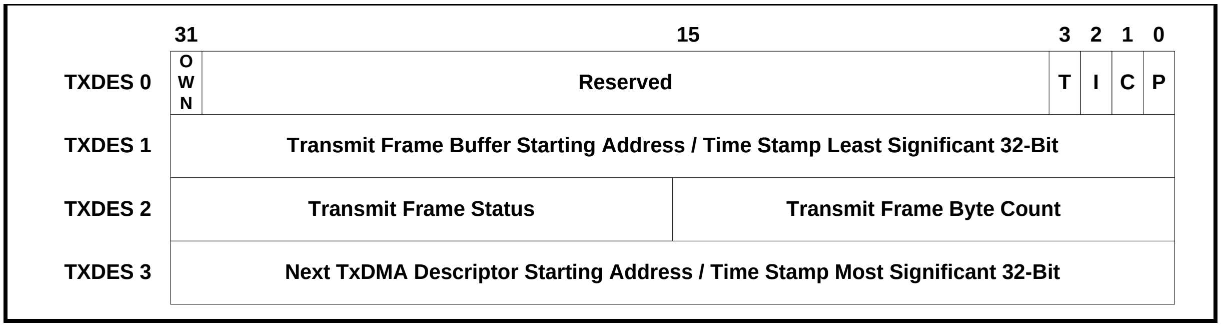
31	30	29	28	27	26	25	24
NRXDSA/TSMSB							
23	22	21	20	19	18	17	16
NRXDSA/TSMSB							
15	14	13	12	11	10	9	8
NRXDSA/TSMSB							
7	6	5	4	3	2	1	0
NRXDSA/TSMSB							

Bits	Description	
[31:0]	**NRXDSA**	**Next RX Descriptor Starting Address** NRXDSA is the starting address of the next RX descriptor. When Ethernet MAC controller fetches the next RX descriptor, it ignored the bits [1:0] of NRXDSA.

Bits	Description	
[31:0]	**TSMSB**	**Time Stamp Most Significant 32-bit** If TSEN (EMACn_TSCR[0]) enabled, Ethernet MAC controller would store time stamp most signification 32-bit value, register EMACn_TSMSR, into this field when it writes back RX Descriptor to system memory.

6.20.6.2 TXDMA Descriptor Data Structure

The TXDMA descriptor consists of four 32-bit words. The data structure of TXDMA descriptor shown in Figure 6.20-5.



	31	15	3	2	1	0
TXDES 0	OWN	Reserved	T	I	C	P
TXDES 1	Transmit Frame Buffer Starting Address / Time Stamp Least Significant 32-Bit					
TXDES 2	Transmit Frame Status	Transmit Frame Byte Count				
TXDES 3	Next TxDMA Descriptor Starting Address / Time Stamp Most Significant 32-Bit					

Figure 6.20-5 TXDMA Descriptor Data Structure

TXDES 0: TXDMA Descriptor Word 0

The TXDMA descriptor word 0 contains a descriptor ownership indicator. In addition, it also contains control bits for transmit frame padding, CRC append, interrupt enable and time stamping control. The detailed description of TXDES 0 is shown below.

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

Bits	Description	
[31]	**Owner**	**Ownership** The ownership field defines which one, the CPU or EMAC, is the owner of each TX descriptor. Only the owner has right to modify the TX descriptor and the other can read the TX descriptor only. If the O=1'b1 indicates the EMAC TXDMA is the owner of TX descriptor and the TX descriptor is available for frame transmission. After the frame transmission completed, EMAC TXDMA modify ownership field to 1'b0 and return the ownership of TX descriptor to CPU. If the O=1'b0 indicates the CPU is the owner of TX descriptor. After the CPU prepares new frame to wait transmission, it modifies the ownership field to 1'b1 and releases the TX descriptor to EMAC TXDMA. 0 = The owner is CPU. 1 = The owner is EMAC.
[30:4]	**Reserved**	Reserved.
[3]	**TTSEN**	**TX Time Stamp Enable Bit** When this bit set high and IEEE 1588 PTP function is also enabled, the embedded time stamping circuit would stamp this frame when SFD of frame is transmitted out on MII/RMII. 0 = IEEE 1588 time stamp function Disabled for this frame. 1 = IEEE 1588 time stamp function Enabled for this frame.
[2]	**INTEN**	**Transmit Interrupt Enable Bit** The INTEN controls the interrupt trigger circuit after the frame transmission completed. If the INTEN enabled, the EMAC will trigger interrupt after frame transmission completed. Otherwise, the interrupt doesn't be triggered. 0 = Frame transmission interrupt masked. 1 = Frame transmission interrupt Enabled.

[1]	**CRCAPP**	**CRC Append** The CRCAPP control the CRC append during frame transmission. If CRCAPP is enabled, the 4-bytes CRC checksum will be appended to frame at the end of frame transmission. 0 = 4-bytes CRC appending Disabled. 1 = 4-bytes CRC appending Enabled.
[0]	**PADEN**	**Padding Enable Bit** The PADEN control the PAD bits appending while the length of transmission frame is less than 60 bytes. If PADEN is enabled, EMAC does the padding automatically. 0 = PAD bits appending Disabled. 1 = PAD bits appending Enabled.

TXDES 1: TXDMA Descriptor Word 1

The TXDMA descriptor word 1 contains the transmit frame buffer starting address or time stamp least significant 32-bit value. The detailed description of TXDES 1 is shown below.

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

Bits	Description	
[31:2]	TXBSA	**Transmit Buffer Starting Address** The TXBSA is the starting address of buffer where transmit packet data stored.

Bits	Description	
[31:0]	TSLSB	**Time Stamp Least Significant 32-bit** If TSEN (EMACn_TSCR[0]) and TTSEN of TX Descriptor word 0 both enabled, Ethernet MAC controller would store time stamp least signification 32-bit value, register EMACn_TSLSR, into this field when it writes back TX Descriptor to system memory.

TXDES 2: TXDMA Descriptor Word 2

The TXDMA descriptor word 2 contains transmit frame status, and transmit frame byte count. The detailed description of TXDES 2 is shown below.

31	30	29	28	27	26	25	24
CCNT				TTSAS	SQE	PAU	TXHA
23	22	21	20	19	18	17	16
LC	TXABT	NCS	EXDEF	TXCP	Reserved	DEF	TXINTR
15	14	13	12	11	10	9	8
TBC							
7	6	5	4	3	2	1	0
TBC							

Bits	Description	
[31:28]	CCNT	**Collision Count** The CCNT indicates the how many collisions found consecutively during a packet transmission. If the packet incurred 16 consecutive collisions during transmission, the CCNT is 0x0 and bit TXABT is set high.
[27]	TTSAS	**TX Time Stamp Active Status** This bit is to indicate the time stamping circuit stamped this frame successfully. When this bit set high, TX Descriptor Word 1 and TX Descriptor Word 3 keep the time stamp value recorded when SFD of frame is transmitted out on MII/RMII. 0 = TX Descriptor Word 1 and TX Descriptor Word 3 does not keep the time stamp value. 1 = TX Descriptor Word 1 and TX Descriptor Word 3 keep the time stamp value.
[26]	SQE	**SQE Error** The SQE indicates the SQE error found at end of packet transmission on 10Mbps half-duplex mode. The SQE error check will only be done while both bit EnSQE of MCMDR is enabled and EMAC is operating on 10Mbps half-duplex mode. 0 = No SQE error found at end of packet transmission. 1 = SQE error found at end of packet transmission.
[25]	PAU	**Transmission Paused** The PAU indicates the next normal packet transmission process will be paused temporally because EMAC received a PAUSE control frame, or software sets the bit SDPZ of MCMDR and enables EMAC to transmit a PAUSE control frame out. 0 = Next normal packet transmission process continue normally. 1 = Next normal packet transmission process paused.
[24]	TXHA	**Transmission Halted** The TXHA indicates the next normal packet transmission process will be halted because the bit TXON of MCMDR is disabled by software. 0 = Next normal packet transmission process continue normally. 1 = Next normal packet transmission process halted.

[23]	**LC**	**Late Collision** The LC indicates the collision found in the outside of 64 bytes collision window. This means after the 64 bytes of a frame has been transmitted out to the network, the collision still found. The late collision check will only be done while EMAC is operating on half-duplex mode. 0 = No collision found in the outside of 64 bytes collision window. 1 = Collision found in the outside of 64 bytes collision window.
[22]	**TXABT**	**Transmission Abort** The TXABT indicates the packet incurred 16 consecutive collisions during transmission, and then the transmission process for this packet is aborted. The transmission abort is only available while EMAC is operating on half-duplex mode. 0 = Packet does not incur 16 consecutive collisions during transmission. 1 = Packet incurred 16 consecutive collisions during transmission.
[21]	**NCS**	**No Carrier Sense** The NCS indicates the MII I/F signal CRS does not active at the start of or during the packet transmission. The NCS is only available while EMAC is operating on half-duplex mode. 0 = CRS signal does not active at the start of or during the packet transmission. 1 = CRS signal actives correctly.
[20]	**EXDEF**	**Defer Exceed** The EXDEF indicates the frame waiting for transmission has deferred over 0.32768ms on 100Mbps mode, or 3.2768ms on 10Mbps mode. The deferral exceed check will only be done while bit NDEF of MCMDR is disabled, and EMAC is operating on half-duplex mode. 0 = Frame waiting for transmission did not defer over 0.32768ms (100Mbps) or 3.2768ms (10Mbps). 1 = Frame waiting for transmission deferred over 0.32768ms (100Mbps) or 3.2768ms (10Mbps).
[19]	**TXCP**	**Transmission Complete** The TXCP indicates the packet transmission has completed correctly. 0 = The packet transmission does not complete. 1 = The packet transmission completed.
[18]	**Reserved**	Reserved.
[17]	**DEF**	**Transmission Deferred** The DEF indicates the packet transmission has deferred once. The DEF is only available while EMAC is operating on half-duplex mode. 0 = Packet transmission does not defer. 1 = Packet transmission deferred once.
[16]	**TXINTR**	**Transmit Interrupt** The TXINTR indicates the packet transmission would trigger an interrupt condition. 0 = The packet transmission would not trigger an interrupt. 1 = The packet transmission would trigger an interrupt.
[15:0]	**TBC**	**Transmit Byte Count** The TBC indicates the byte count of the frame stored in the data buffer pointed by TX descriptor for transmission.

TXDES 3: TXDMA Descriptor Word 3

The TXDMA descriptor word 3 contains the next TXDMA descriptor starting address or time stamp most significant 32-bit value. The detailed description of TXDES 3 is shown below.

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

Bits	Description	
[31:0]	**NTXDSA**	**Next TX Descriptor Starting Address** NTXDSA is the starting address of the next TX descriptor. When Ethernet MAC controller fetch the next TX descriptor, it ignored the bits [1:0] of NTXDSA.

Bits	Description	
[31:0]	**TSMSB**	**Time Stamp Most Significant 32-bit** If TSEN (EMACn_TSCR[0]) and TTSEN of TX Descriptor word 0 are both enabled, Ethernet MAC controller would store time stamp most signification 32-bit value, register EMACn_TSMSR, into this field when it writes back TX Descriptor to system memory.

6.20.7 Register Map

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

Register	Offset	R/W	Description	Reset Value
EMAC Base Address: **EMAC0_BA = 0xB001_2000** **EMAC1_BA = 0xB002_2000**				
EMACn_CAMCMR **n=0,1**	EMACn_BA+0x000	R/W	EMAC n CAM Command Register	0x0000_0000
EMACn_CAMEN **n=0,1**	EMACn_BA+0x004	R/W	EMAC n CAM Enable Register	0x0000_0000
EMACn_CAM0M **n=0,1**	EMACn_BA+0x008	R/W	EMAC n CAM 0 Most Significant Word Register	0x0000_0000
EMACn_CAM0L **n=0,1**	EMACn_BA+0x00C	R/W	EMAC n CAM 0 Least Significant Word Register	0x0000_0000
EMACn_CAM1M **n=0,1**	EMACn_BA+0x010	R/W	EMAC n CAM 1 Most Significant Word Register	0x0000_0000
EMACn_CAM1L **n=0,1**	EMACn_BA+0x014	R/W	EMAC n CAM 1 Least Significant Word Register	0x0000_0000
EMACn_CAM2M **n=0,1**	EMACn_BA+0x018	R/W	EMAC n CAM 2 Most Significant Word Register	0x0000_0000
EMACn_CAM2L **n=0,1**	EMACn_BA+0x01C	R/W	EMAC n CAM 2 Least Significant Word Register	0x0000_0000
EMACn_CAM3M **n=0,1**	EMACn_BA+0x020	R/W	EMAC n CAM 3 Most Significant Word Register	0x0000_0000
EMACn_CAM3L **n=0,1**	EMACn_BA+0x024	R/W	EMAC n CAM 3 Least Significant Word Register	0x0000_0000
EMACn_CAM4M **n=0,1**	EMACn_BA+0x028	R/W	EMAC n CAM 4 Most Significant Word Register	0x0000_0000
EMACn_CAM4L **n=0,1**	EMACn_BA+0x02C	R/W	EMAC n CAM 4 Least Significant Word Register	0x0000_0000
EMACn_CAM5M **n=0,1**	EMACn_BA+0x030	R/W	EMAC n CAM 5 Most Significant Word Register	0x0000_0000
EMACn_CAM5L **n=0,1**	EMACn_BA+0x034	R/W	EMAC n CAM 5 Least Significant Word Register	0x0000_0000
EMACn_CAM6M **n=0,1**	EMACn_BA+0x038	R/W	EMAC n CAM 6 Most Significant Word Register	0x0000_0000
EMACn_CAM6L **n=0,1**	EMACn_BA+0x03C	R/W	EMAC n CAM 6 Least Significant Word Register	0x0000_0000

EMACn_CAM7M **n=0,1**	EMACn_BA+0x040	R/W	EMAC n CAM 7 Most Significant Word Register	0x0000_0000
EMACn_CAM7L **n=0,1**	EMACn_BA+0x044	R/W	EMAC n CAM 7 Least Significant Word Register	0x0000_0000
EMACn_CAM8M **n=0,1**	EMACn_BA+0x048	R/W	EMAC n CAM 8 Most Significant Word Register	0x0000_0000
EMACn_CAM8L **n=0,1**	EMACn_BA+0x04C	R/W	EMAC n CAM 8 Least Significant Word Register	0x0000_0000
EMACn_CAM9M **n=0,1**	EMACn_BA+0x050	R/W	EMAC n CAM 9 Most Significant Word Register	0x0000_0000
EMACn_CAM9L **n=0,1**	EMACn_BA+0x054	R/W	EMAC n CAM 9 Least Significant Word Register	0x0000_0000
EMACn_CAM10M **n=0,1**	EMACn_BA+0x058	R/W	EMAC n CAM 10 Most Significant Word Register	0x0000_0000
EMACn_CAM10L **n=0,1**	EMACn_BA+0x05C	R/W	EMAC n CAM 10 Least Significant Word Register	0x0000_0000
EMACn_CAM11M **n=0,1**	EMACn_BA+0x060	R/W	EMAC n CAM 11 Most Significant Word Register	0x0000_0000
EMACn_CAM11L **n=0,1**	EMACn_BA+0x064	R/W	EMAC n CAM 11 Least Significant Word Register	0x0000_0000
EMACn_CAM12M **n=0,1**	EMACn_BA+0x068	R/W	EMAC n CAM 12 Most Significant Word Register	0x0000_0000
EMACn_CAM12L **n=0,1**	EMACn_BA+0x06C	R/W	EMAC n CAM 12 Least Significant Word Register	0x0000_0000
EMACn_CAM13M **n=0,1**	EMACn_BA+0x070	R/W	EMAC n CAM 13 Most Significant Word Register	0x0000_0000
EMACn_CAM13L **n=0,1**	EMACn_BA+0x074	R/W	EMAC n CAM 13 Least Significant Word Register	0x0000_0000
EMACn_CAM14M **n=0,1**	EMACn_BA+0x078	R/W	EMAC n CAM 14 Most Significant Word Register	0x0000_0000
EMACn_CAM14L **n=0,1**	EMACn_BA+0x07C	R/W	EMAC n CAM 14 Least Significant Word Register	0x0000_0000
EMACn_CAM15M **n=0,1**	EMACn_BA+0x080	R/W	EMAC n CAM 15 Most Significant Word Register	0x0000_0000
EMACn_CAM15L **n=0,1**	EMACn_BA+0x084	R/W	EMAC n CAM 15 Least Significant Word Register	0x0000_0000
EMACn_TXDLSA **n=0,1**	EMACn_BA+0x088	R/W	EMAC n Transmit Descriptor Link List Start Address Register	0xFFFF_FFFC
EMACn_RXDLSA **n=0,1**	EMACn_BA+0x08C	R/W	EMAC n Receive Descriptor Link List Start Address Register	0xFFFF_FFFC

EMACn_MCMDR **n=0,1**	EMACn_BA+0x090	R/W	EMAC n MAC Command Register	0x0040_0000
EMACn_MIID **n=0,1**	EMACn_BA+0x094	R/W	EMAC n MII Management Data Register	0x0000_0000
EMACn_MIIDA **n=0,1**	EMACn_BA+0x098	R/W	EMAC n MII Management Control and Address Register	0x0090_0000
EMACn_FFTCR **n=0,1**	EMACn_BA+0x09C	R/W	EMAC n FIFO Threshold Control Register	0x0000_0101
EMACn_TSDR **n=0,1**	EMACn_BA+0x0A0	W	EMAC n Transmit Start Demand Register	Undefined
EMACn_RSDR **n=0,1**	EMACn_BA+0x0A4	W	EMAC n Receive Start Demand Register	Undefined
EMACn_DMARFC **n=0,1**	EMACn_BA+0x0A8	R/W	EMAC n Maximum Receive Frame Control Register	0x0000_0800
EMACn_MIEN **n=0,1**	EMACn_BA+0x0AC	R/W	EMAC n MAC Interrupt Enable Register	0x0000_0000
EMACn_MISTA **n=0,1**	EMACn_BA+0x0B0	R/W	EMAC n MAC Interrupt Status Register	0x0000_0000
EMACn_MGSTA **n=0,1**	EMACn_BA+0x0B4	R/W	EMAC n MAC General Status Register	0x0000_0000
EMACn_MPCNT **n=0,1**	EMACn_BA+0x0B8	R/W	EMAC n Missed Packet Count Register	0x0000_7FFF
EMACn_MRPC **n=0,1**	EMACn_BA+0x0BC	R	EMAC n MAC Receive Pause Count Register	0x0000_0000
EMACn_DMARFS **n=0,1**	EMACn_BA+0x0C8	R/W	EMAC n DMA Receive Frame Status Register	0x0000_0000
EMACn_CTXDSA **n=0,1**	EMACn_BA+0x0CC	R	EMAC n Current Transmit Descriptor Start Address Register	0x0000_0000
EMACn_CTXBSA **n=0,1**	EMACn_BA+0x0D0	R	EMAC n Current Transmit Buffer Start Address Register	0x0000_0000
EMACn_CRXDSA **n=0,1**	EMACn_BA+0x0D4	R	EMAC n Current Receive Descriptor Start Address Register	0x0000_0000
EMACn_CRXBSA **n=0,1**	EMACn_BA+0x0D8	R	EMAC n Current Receive Buffer Start Address Register	0x0000_0000
EMACn_TSCTL **n=0,1**	EMACn_BA+0x100	R/W	EMAC n Time Stamp Control Register	0x0000_0000
EMACn_TSSEC **n=0,1**	EMACn_BA+0x110	R	EMAC n Time Stamp Counter Second Register	0x0000_0000
EMACn_TSSUBSEC **n=0,1**	EMACn_BA+0x114	R	EMAC n Time Stamp Counter Sub Second Register	0x0000_0000

EMACn_TSINC **n=0,1**	EMACn_BA+0x118	R/W	EMAC n Time Stamp Increment Register	0x0000_0000
EMACn_TSADDEND **n=0,1**	EMACn_BA+0x11C	R/W	EMAC n Time Stamp Addend Register	0x0000_0000
EMACn_UPDSEC **n=0,1**	EMACn_BA+0x120	R/W	EMAC n Time Stamp Update Second Register	0x0000_0000
EMACn_UPDSSEC **n=0,1**	EMACn_BA+0x124	R/W	EMAC n Time Stamp Update Sub Second Register	0x0000_0000
EMACn_ALMSEC **n=0,1**	EMACn_BA+0x128	R/W	EMAC n Time Stamp Alarm Second Register	0x0000_0000
EMACn_ALMSSEC **n=0,1**	EMACn_BA+0x12C	R/W	EMAC n Time Stamp Alarm Sub Second Register	0x0000_0000

6.20.8 Register Description

EMAC n CAM Command Register (EMACn_CAMCMR)

The EMAC supports CAM function for destination MAC address recognition. The EMACn_CAMCMR control the CAM comparison function, and unicast, multicast, and broadcast packet reception.

Register	Offset	R/W	Description	Reset Value
EMACn_CAMCMR **n=0,1**	EMACn_BA+0x000	R/W	EMAC n CAM 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			**ECMP**	**CCAM**	**ABP**	**AMP**	**AUP**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**ECMP**	**CAM Compare Enable Bit** The ECMP controls the enable of CAM comparison function for destination MAC address recognition. If software wants to receive a packet with specific destination MAC address, configures the MAC address into CAM 12~0, then enables that CAM entry and set ECMP to 1. 0 = CAM comparison function for destination MAC address recognition Disabled. 1 = CAM comparison function for destination MAC address recognition Enabled.
[3]	**CCAM**	**Complement CAM Compare** The CCAM controls the complement of the CAM comparison result. If the ECMP and CCAM are both enabled, the incoming packet with specific destination MAC address configured in CAM entry will be dropped. And the incoming packet with destination MAC address does not configured in any CAM entry will be received. 0 = The CAM comparison result does not complement. 1 = The CAM comparison result complemented.
[2]	**ABP**	**Accept Broadcast Packet** The ABP controls the broadcast packet reception. If ABP is enabled, EMAC receives all incoming packet its destination MAC address is a broadcast address. 0 = EMAC receives packet depends on the CAM comparison result. 1 = EMAC receives all broadcast packets.

[1]	**AMP**	**Accept Multicast Packet** The AMP controls the multicast packet reception. If AMP is enabled, EMAC receives all incoming packet its destination MAC address is a multicast address. 0 = EMAC receives packet depends on the CAM comparison result. 1 = EMAC receives all multicast packets.
[0]	**AUP**	**Accept Unicast Packet** The AUP controls the unicast packet reception. If AUP is enabled, EMAC receives all incoming packet its destination MAC address is a unicast address. 0 = EMAC receives packet depends on the CAM comparison result. 1 = EMAC receives all unicast packets.

CAMCMR Setting and Comparison Results

Figure 6.20-6 is the address recognition result in different CAMCMR configuration. The column Result shows the incoming packet type that can pass the address recognition in specific CAM configuration. The C, U, M and B represents the:

C: Indicates the destination MAC address of incoming packet has been configured in CAM entry.

U: Indicates the incoming packet is a unicast packet.

M: Indicates the incoming packet is a multicast packet.

B: Indicates the incoming packet is a broadcast packet.

ECMP	CCAM	AUP	AMP	ABP	Result			
0	0	0	0	0	No Packet			
0	0	0	0	1	B			
0	0	0	1	0	M			
0	0	0	1	1	M	B		
0	0	1	0	0	C	U		
0	0	1	0	1	C	U	B	
0	0	1	1	0	C	U	M	
0	0	1	1	1	C	U	M	B
0	1	0	0	0	C	U	M	B
0	1	0	0	1	C	U	M	B
0	1	0	1	0	C	U	M	B
0	1	0	1	1	C	U	M	B
0	1	1	0	0	C	U	M	B
0	1	1	0	1	C	U	M	B
0	1	1	1	0	C	U	M	B
0	1	1	1	1	C	U	M	B
1	0	0	0	0	C			
1	0	0	0	1	C	B		

1	0	0	1	0	C	M		
1	0	0	1	1	C	M	B	
1	0	1	0	0	C	U		
1	0	1	0	1	C	U	B	
1	0	1	1	0	C	U	M	
1	0	1	1	1	C	U	M	B
1	1	0	0	0	U	M	B	
1	1	0	0	1	U	M	B	
1	1	0	1	0	U	M	B	
1	1	0	1	1	U	M	B	
1	1	1	0	0	C	U	M	B
1	1	1	0	1	C	U	M	B
1	1	1	1	0	C	U	M	B
1	1	1	1	1	C	U	M	B

Figure 6.20-6 Different CAMCMR Setting and Type of Received Packet

EMAC n CAM Enable Register (EMACn_CAMEN)

The EMACn_CAMEN controls the validation of each CAM entry. Each CAM entry must be enabled first before it participates in the destination MAC address recognition.

Register	Offset	R/W	Description	Reset Value
EMACn_CAMEN **n=0,1**	EMACn_BA+0x004	R/W	EMAC n CAM 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**
CAM15EN	**CAM14EN**	**CAM13EN**	**CAM12EN**	**CAM11EN**	**CAM10EN**	**CAM9EN**	**CAM8EN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CAM7EN	**CAM6EN**	**CAM5EN**	**CAM4EN**	**CAM3EN**	**CAM2EN**	**CAM1EN**	**CAM0EN**

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CAMxEN**	**CAM Entry x Enable Bit** The CAMxEN controls the validation of CAM entry x. The x can be 0 to 15. The CAM entry 13, 14 and 15 are for PAUSE control frame transmission. If software wants to transmit a PAUSE control frame out to network, the enable bits of these three CAM entries all must be enabled first. 0 = CAM entry x Disabled. 1 = CAM entry x Enabled.

EMAC n CAM x Most Significant Word Register (EMACn_CAMxM)

The EMAC is equipped with 16 CAM entries. In these 16 CAM entries, 13 entries (entry 0~12) are to keep destination MAC address for packet recognition, and the other 3 entries (entry 13~15) are for PAUSE control frame transmission. Each CAM entry consists of 6 bytes. Consequently, 2 register are used for each CAM entry.

For packet recognition, a register pair {EMACn_CAMxM, EMACn_CAMxL} represents a CAM entry and can keep a destination MAC address. The corresponding CAM enable bit CAMxEN (EMACn_CAMEN[x]) is also needed be enabled. The x can be the 0 to 12.

The register pairs {EMACn_CAM13M, EMACn_CAM13L}, {EMACn_CAM14M, EMACn_CAM14L} and {EMACn_CAM15M, EMACn_CAM15L} are used for flow control function.

Register	Offset	R/W	Description	Reset Value
EMACn_CAM0M **n=0,1**	EMACn_BA+0x008	R/W	EMAC n CAM 0 Most Significant Word Register	0x0000_0000
EMACn_CAM1M **n=0,1**	EMACn_BA+0x010	R/W	EMAC n CAM 1 Most Significant Word Register	0x0000_0000
EMACn_CAM2M **n=0,1**	EMACn_BA+0x018	R/W	EMAC n CAM 2 Most Significant Word Register	0x0000_0000
EMACn_CAM3M **n=0,1**	EMACn_BA+0x020	R/W	EMAC n CAM 3 Most Significant Word Register	0x0000_0000
EMACn_CAM4M **n=0,1**	EMACn_BA+0x028	R/W	EMAC n CAM 4 Most Significant Word Register	0x0000_0000
EMACn_CAM5M **n=0,1**	EMACn_BA+0x030	R/W	EMAC n CAM 5 Most Significant Word Register	0x0000_0000
EMACn_CAM6M **n=0,1**	EMACn_BA+0x038	R/W	EMAC n CAM 6 Most Significant Word Register	0x0000_0000
EMACn_CAM7M **n=0,1**	EMACn_BA+0x040	R/W	EMAC n CAM 7 Most Significant Word Register	0x0000_0000
EMACn_CAM8M **n=0,1**	EMACn_BA+0x048	R/W	EMAC n CAM 8 Most Significant Word Register	0x0000_0000
EMACn_CAM9M **n=0,1**	EMACn_BA+0x050	R/W	EMAC n CAM 9 Most Significant Word Register	0x0000_0000
EMACn_CAM10M **n=0,1**	EMACn_BA+0x058	R/W	EMAC n CAM 10 Most Significant Word Register	0x0000_0000
EMACn_CAM11M **n=0,1**	EMACn_BA+0x060	R/W	EMAC n CAM 11 Most Significant Word Register	0x0000_0000
EMACn_CAM12M **n=0,1**	EMACn_BA+0x068	R/W	EMAC n CAM 12 Most Significant Word Register	0x0000_0000
EMACn_CAM13M **n=0,1**	EMACn_BA+0x070	R/W	EMAC n CAM 13 Most Significant Word Register	0x0000_0000
EMACn_CAM14M	EMACn_BA+0x078	R/W	EMAC n CAM 14 Most Significant Word Register	0x0000_0000

n=0,1				

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

Bits	Description	
[31:0]	**CAMxM**	**CAMx Most Significant Word** The CAMxM keeps the bit 47~16 of MAC address. The x can be the 0~12. The register pair {EMACn_CAMxM, EMACn_CAMxL} represents a CAM entry and keeps a MAC address. For example, if the MAC address 00-50-BA-33-BA-44 kept in CAM entry 1, the register EMACn_CAM1M is 32"h0050_BA33 and EMACn_CAM1L is 32"hBA44_0000.

EMAC n CAM x Least Significant Word Register (EMACn_CAMxL)

The EMAC is equipped with 16 CAM entries. In these 16 CAM entries, 13 entries (entry 0~12) are to keep destination MAC address for packet recognition, and the other 3 entries (entry 13~15) are for PAUSE control frame transmission. Each CAM entry consists of 6 bytes. Consequently, 2 register are used for each CAM entry.

For packet recognition, a register pair {EMACn_CAMxM, EMACn_CAMxL} represents a CAM entry and can keep a destination MAC address. The corresponding CAM enable bit CAMxEN (EMACn_CAMEN[x]) is also needed be enabled. The x can be the 0 to 12.

The register pairs {EMACn_CAM13M, EMACn_CAM13L}, {EMACn_CAM14M, EMACn_CAM14L} and {EMACn_CAM15M, EMACn_CAM15L} are used for flow control function.

Register	Offset	R/W	Description	Reset Value
EMACn_CAM0L **n=0,1**	EMACn_BA+0x00C	R/W	EMAC n CAM 0 Least Significant Word Register	0x0000_0000
EMACn_CAM1L **n=0,1**	EMACn_BA+0x014	R/W	EMAC n CAM 1 Least Significant Word Register	0x0000_0000
EMACn_CAM2L **n=0,1**	EMACn_BA+0x01C	R/W	EMAC n CAM 2 Least Significant Word Register	0x0000_0000
EMACn_CAM3L **n=0,1**	EMACn_BA+0x024	R/W	EMAC n CAM 3 Least Significant Word Register	0x0000_0000
EMACn_CAM4L **n=0,1**	EMACn_BA+0x02C	R/W	EMAC n CAM 4 Least Significant Word Register	0x0000_0000
EMACn_CAM5L **n=0,1**	EMACn_BA+0x034	R/W	EMAC n CAM 5 Least Significant Word Register	0x0000_0000
EMACn_CAM6L **n=0,1**	EMACn_BA+0x03C	R/W	EMAC n CAM 6 Least Significant Word Register	0x0000_0000
EMACn_CAM7L **n=0,1**	EMACn_BA+0x044	R/W	EMAC n CAM 7 Least Significant Word Register	0x0000_0000
EMACn_CAM8L **n=0,1**	EMACn_BA+0x04C	R/W	EMAC n CAM 8 Least Significant Word Register	0x0000_0000
EMACn_CAM9L **n=0,1**	EMACn_BA+0x054	R/W	EMAC n CAM 9 Least Significant Word Register	0x0000_0000
EMACn_CAM10L **n=0,1**	EMACn_BA+0x05C	R/W	EMAC n CAM 10 Least Significant Word Register	0x0000_0000
EMACn_CAM11L **n=0,1**	EMACn_BA+0x064	R/W	EMAC n CAM 11 Least Significant Word Register	0x0000_0000
EMACn_CAM12L **n=0,1**	EMACn_BA+0x06C	R/W	EMAC n CAM 12 Least Significant Word Register	0x0000_0000
EMACn_CAM13L **n=0,1**	EMACn_BA+0x074	R/W	EMAC n CAM 13 Least Significant Word Register	0x0000_0000
EMACn_CAM14L	EMACn_BA+0x07C	R/W	EMAC n CAM 14 Least Significant Word Register	0x0000_0000

NUC980 SERIES TECHNICAL REFERENCE MANUAL

n=0,1				

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

Bits	Description	
[31:16]	**CAMxL**	**CAMx Least Significant Word** The CAMxL keeps the bit 15~0 of MAC address. The x can be the 0~14. The register pair {EMACn_CAMxM, EMACn_CAMxL} represents a CAM entry and keeps a MAC address. For example, if the MAC address 00-50-BA-33-BA-44 kept in CAM entry 1, the register EMACn_CAM1M is 32"h0050_BA33 and EMACn_CAM1L is 32"hBA44_0000.
[15:0]	**Reserved**	Reserved.

EMAC n CAM 15 Most Significant Word Register (EMACn_CAM15M)

The EMAC is equipped with 16 CAM entries. In these 16 CAM entries, 13 entries (entry 0~12) are to keep destination MAC address for packet recognition, and the other 3 entries (entry 13~15) are for PAUSE control frame transmission. Each CAM entry consists of 6 bytes. Consequently, 2 register are used for each CAM entry.

For packet recognition, a register pair {EMACn_CAMxM, EMACn_CAMxL} represents a CAM entry and can keep a destination MAC address. The corresponding CAM enable bit CAMxEN (EMACn_CAMEN[x]) is also needed be enabled. The x can be the 0 to 12.

The register pairs {EMACn_CAM13M, EMACn_CAM13L}, {EMACn_CAM14M, EMACn_CAM14L} and {EMACn_CAM15M, EMACn_CAM15L} are used for flow control function.

Register	Offset	R/W	Description	Reset Value
EMACn_CAM15M **n=0,1**	EMACn_BA+0x080	R/W	EMAC n CAM 15 Most Significant Word Register	0x0000_0000

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

Bits	Description	
[31:16]	**Len_Type**	**Length/Type Field of PAUSE Control Frame** In the PAUSE control frame, a length/type field defined and is 16‟h8808.
[15:0]	**OP_Code**	**OP Code Field of PAUSE Control Frame** In the PAUSE control frame, an op code field defined and is 16‟h0001.

EMAC n CAM 15 Least Significant Word Register (EMACn_CAM15L)

The EMAC is equipped with 16 CAM entries. In these 16 CAM entries, 13 entries (entry 0~12) are to keep destination MAC address for packet recognition, and the other 3 entries (entry 13~15) are for PAUSE control frame transmission. Each CAM entry consists of 6 bytes. Consequently, 2 register are used for each CAM entry.

For packet recognition, a register pair {EMACn_CAMxM, EMACn_CAMxL} represents a CAM entry and can keep a destination MAC address. The corresponding CAM enable bit CAMxEN (EMACn_CAMEN[x]) is also needed be enabled. The x can be the 0 to 12.

The register pairs {EMACn_CAM13M, EMACn_CAM13L}, {EMACn_CAM14M, EMACn_CAM14L} and {EMACn_CAM15M, EMACn_CAM15L} are used for flow control function.

Register	Offset	R/W	Description	Reset Value
EMACn_CAM15L **n=0,1**	EMACn_BA+0x084	R/W	EMAC n CAM 15 Least Significant Word Register	0x0000_0000

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

Bits	Description	
[31:24]	**Operand**	**Pause Parameter** In the PAUSE control frame, an operand field defined and controls how much time the destination Ethernet MAC Controller paused. The unit of the operand is a slot time, the 512 bits time.
[23:0]	**Reserved**	Reserved.

EMAC n Transmit Descriptor Link List Start Address Register (EMACn_TXDLSA)

The TX descriptor defined in EMAC is a link-list data structure. The EMACn_TXDLSA keeps the starting address of this link-list. In other words, the EMACn_TXDLSA keeps the starting address of the 1st TX descriptor. EMACn_TXDLSA must be configured by software before the bit TXON (EMACn_MCMDR[8]) is enabled.

Register	Offset	R/W	Description	Reset Value
EMACn_TXDLSA **n=0,1**	EMACn_BA+0x088	R/W	EMAC n Transmit Descriptor Link List Start Address Register	0xFFFF_FFFC

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

Bits	Description	
[31:0]	**TXDLSA**	**Transmit Descriptor Link-list Start Address** The TXDLSA keeps the start address of transmit descriptor link-list. If the software enables the bit TXON (EMACn_MCMDR[8]), the content of TXDLSA will be loaded into the current transmit descriptor start address register (EMACn_CTXDSA). The TXDLSA does not be updated by EMAC. During the operation, EMAC will ignore the bits [1:0] of TXDLSA. This means that each TX descriptor always must locate at word boundary memory address.

EMAC n Receive Descriptor Link List Start Address Register (EMACn_RXDLSA)

The RX descriptor defined in EMAC is a link-list data structure. The EMACn_RXDLSA keeps the starting address of this link-list. In other words, the EMACn_RXDLSA keeps the starting address of the 1st RX descriptor. EMACn_RXDLSA must be configured by software before the bit RXON (EMACn_MCMDR[0]) is enabled.

Register	Offset	R/W	Description	Reset Value
EMACn_RXDLSA **n=0,1**	EMACn_BA+0x08C	R/W	EMAC n Receive Descriptor Link List Start Address Register	0xFFFF_FFFC

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

Bits	Description	
[31:0]	**RXDLSA**	**Receive Descriptor Link-list Start Address** The RXDLSA keeps the start address of receive descriptor link-list. If the S/W enables the bit RXON (EMACn_MCMDR[0]), the content of RXDLSA will be loaded into the current receive descriptor start address register (EMACn_CRXDSA). The RXDLSA does not be updated by EMAC. During the operation, EMAC will ignore the bits [1:0] of RXDLSA. This means that each RX descriptor always must locate at word boundary memory address.

EMAC n MAC Command Register (EMACn_MCMDR)

The EMACn_MCMDR provides the control information for EMAC. Some command settings affect both frame transmission and reception, such as bit FDUP (EMACn_MCMDR[18]), the full/half duplex mode selection, or bit OPMOD (EMACn_MCMDR[20]), the 100/10M bps mode selection. Some command settings control frame transmission and reception separately, likes bit TXON (EMACn_MCMDR[8]) and RXON (EMACn_MCMDR[0]).

Register	Offset	R/W	Description	Reset Value
EMACn_MCMDR **n=0,1**	EMACn_BA+0x090	R/W	EMAC n MAC Command Register	0x0040_0000

31	30	29	28	27	26	25	24
Reserved							**SWR**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
REFCLKI			**OPMOD**	**Reserved**	**FDUP**	**SQECHKEN**	**SDPZ**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**NDEF**	**TXON**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PTP_SRC	**MGP_WAKE**	**SPCRC**	**AEP**	**ACP**	**ARP**	**ALP**	**RXON**

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**SWR**	**Software Reset** The SWR implements a reset function to make the EMAC return default state. The SWR is a self-clear bit. This means after the software reset finished, the SWR will be cleared automatically. Enable SWR can also reset all control and status registers, exclusive of the control bits EnRMII (EMACn_MCMDR[22]), LBK (EMACn_MCMDR[21]) and OPMOD (EMACn_MCMDR[20]). The EMAC re-initial is necessary after the software reset completed. 0 = Software reset completed. 1 = Software reset Enabled.
[23:21]	**Reserved**	Reserved.
[20]	**OPMOD**	**Operation Mode Selection** The OPMOD defines that if the EMAC is operating on 10M or 100M bps mode. The SWR would not affect OPMOD value. 0 = EMAC operates in 10Mbps mode. 1 = EMAC operates in 100Mbps mode.
[19]	**Reserved**	Reserved.

[18]	**FDUP**	**Full Duplex Mode Selection** The FDUP controls that if EMAC is operating on full or half duplex mode. 0 = EMAC operates in half duplex mode. 1 = EMAC operates in full duplex mode.
[17]	**SQECHKEN**	**SQE Checking Enable Bit** The SQECHKEN controls the enable of SQE checking. The SQE checking is only available while EMAC is operating on 10M bps and half duplex mode. In other words, the SQECHKEN cannot affect EMAC operation, if the EMAC is operating on 100M bps or full duplex mode. 0 = SQE checking Disabled while EMAC is operating in 10Mbps and Half Duplex mode. 1 = SQE checking Enabled while EMAC is operating in 10Mbps and Half Duplex mode.
[16]	**SDPZ**	**Send PAUSE Frame** The SDPZ controls the PAUSE control frame transmission. If S/W wants to send a PAUSE control frame out, the CAM entry 13, 14 and 15 must be configured first and the corresponding CAM enable bit of CAMEN register also must be set. Then, set SDPZ to 1 enables the PAUSE control frame transmission. The SDPZ is a self-clear bit. This means after the PAUSE control frame transmission has completed, the SDPZ will be cleared automatically. It is recommended that only enabling SPDZ while EMAC is operating in Full Duplex mode. 0 = PAUSE control frame transmission completed. 1 = PAUSE control frame transmission Enabled.
[15:10]	**Reserved**	Reserved.
[9]	**NDEF**	**No Deferral** The NDEF controls the enable of deferral exceed counter. If NDEF is set to high, the deferral exceed counter is disabled. The NDEF is only useful while EMAC is operating on half duplex mode. 0 = The deferral exceed counter Enabled. 1 = The deferral exceed counter Disabled.
[8]	**TXON**	**Frame Transmission ON** The TXON controls the normal packet transmission of EMAC. If the TXON is set to high, the EMAC starts the packet transmission process, including the TX descriptor fetching, packet transmission and TX descriptor modification. It is must to finish EMAC initial sequence before enable TXON. Otherwise, the EMAC operation is undefined. If the TXON is disabled during EMAC is transmitting a packet out, the EMAC stops the packet transmission process after the current packet transmission finished. 0 = Packet transmission process stopped. 1 = Packet transmission process started.
[7]	**PTP_SRC**	**PTP Counter Source Selection** This bit control the PTP counter source is from EMC0 or from EMC1 internally. 1’b0: The PTP counter source is from EMC1 internally 1’b1: The PTP counter source is from EMC0. **Note:** This bit is only available in EMAC1. In EMAC0, this bit is reserved.

[6]	**MGP_WAKE**	**Magic Packet Wake-up Enable Bit** The MGP_WAKE high enables the functionality that Ethernet MAC controller checked if the incoming packet is Magic Packet and wakeup system from Power-down mode. If incoming packet was a Magic Packet and the system was in Power-down, the Ethernet MAC controller would generate a wakeup event to wake system up from Power-down mode. 0 = Wake-up by Magic Packet function Disabled. 1 = Wake-up by Magic Packet function Enabled.
[5]	**SPCRC**	**Strip CRC Checksum** The SPCRC controls if the length of incoming packet is calculated with 4 bytes CRC checksum. If the SPCRC is set to high, 4 bytes CRC checksum is excluded from length calculation of incoming packet. 0 = The 4 bytes CRC checksum is included in packet length calculation. 1 = The 4 bytes CRC checksum is excluded in packet length calculation.
[4]	**AEP**	**Accept CRC Error Packet** The AEP controls the EMAC accepts or drops the CRC error packet. If the AEP is set to high, the incoming packet with CRC error will be received by EMAC as a good packet. 0 = Ethernet MAC controller dropped the CRC error packet. 1 = Ethernet MAC controller received the CRC error packet.
[3]	**ACP**	**Accept Control Packet** The ACP controls the control frame reception. If the ACP is set to high, the EMAC will accept the control frame. Otherwise, the control frame will be dropped. It is recommended that S/W only enable ACP while EMAC is operating on full duplex mode. 0 = Ethernet MAC controller dropped the control frame. 1 = Ethernet MAC controller received the control frame.
[2]	**ARP**	**Accept Runt Packet** The ARP controls the runt packet, which length is less than 64 bytes, reception. If the ARP is set to high, the EMAC will accept the runt packet. Otherwise, the runt packet will be dropped. 0 = Ethernet MAC controller dropped the runt packet. 1 = Ethernet MAC controller received the runt packet.
[1]	**ALP**	**Accept Long Packet** The ALP controls the long packet, which packet length is greater than 1518 bytes, reception. If the ALP is set to high, the EMAC will accept the long packet. Otherwise, the long packet will be dropped. 0 = Ethernet MAC controller dropped the long packet. 1 = Ethernet MAC controller received the long packet.
[0]	**RXON**	**Frame Reception ON** The RXON controls the normal packet reception of EMAC. If the RXON is set to high, the EMAC starts the packet reception process, including the RX descriptor fetching, packet reception and RX descriptor modification. It is necessary to finish EMAC initial sequence before enable RXON. Otherwise, the EMAC operation is undefined. If the RXON is disabled during EMAC is receiving an incoming packet, the EMAC stops the packet reception process after the current packet reception finished. 0 = Packet reception process stopped. 1 = Packet reception process started.

EMAC n MII Management Data Register (EMACn_MIID)

The EMAC provides MII management function to access the control and status registers of the external PHY. The EMACn_MIID register is used to store the data that will be written into the registers of external PHY for write command or the data that is read from the registers of external PHY for read command.

Register	Offset	R/W	Description	Reset Value
EMACn_MIID **n=0,1**	EMACn_BA+0x094	R/W	EMAC n MII Management 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
DATA							
7	6	5	4	3	2	1	0
DATA							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**DATA**	**MII Management Data** The MIIData is the 16 bits data that will be written into the registers of external PHY for MII Management write command or the data from the registers of external PHY for MII Management read command.

EMAC n MII Management Control and Address Register (EMACn_MIIDA)

The EMAC provides MII management function to access the control and status registers of the external PHY. The EMACn_MIIDA register is used to keep the MII management command information, like the register address, external PHY address, MDC clocking rate, read/write etc.

Register	Offset	R/W	Description	Reset Value
EMACn_MIIDA **n=0,1**	EMACn_BA+0x098	R/W	EMAC n MII Management Control and Address Register	0x0090_0000

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

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19]	**MDCON**	**MDC Clock ON Always** The MDC controls the MDC clock generation. If the MDCON is set to high, the MDC clock actives always. Otherwise, the MDC will only active while S/W issues a MII management command. 0 = MDC clock only actives while S/W issues a MII management command. 1 = MDC clock actives always.
[18]	**PREAMSP**	**Preamble Suppress** The PREAMSP controls the preamble field generation of MII management frame. If the PREAMSP is set to high, the preamble field generation of MII management frame is skipped. 0 = Preamble field generation of MII management frame not skipped. 1 = Preamble field generation of MII management frame skipped.
[17]	**BUSY**	**Busy Bit** The BUSY controls the enable of the MII management frame generation. If S/W wants to access registers of external PHY, it set BUSY to high and EMAC generates the MII management frame to external PHY through MII Management I/F. The BUSY is a self-clear bit. This means the BUSY will be cleared automatically after the MII management command finished. 0 = MII management command generation finished. 1 = MII management command generation Enabled.

[16]	**WRITE**	**Write Command** The WRITE defines the MII management command is a read or write. 0 = MII management command is a read command. 1 = MII management command is a write command.
[15:13]	**Reserved**	Reserved.
[12:8]	**PHYAD**	**PHY Address** The PHYAD keeps the address to differentiate which external PHY is the target of the MII management command.
[7:5]	**Reserved**	Reserved.
[4:0]	**PHYRAD**	**PHY Register Address** The PHYRAD keeps the address to indicate which register of external PHY is the target of the MII management command.

MII Management Function Frame Format

In IEEE Std. 802.3 clause 22.2.4, the MII management function is defined. The MII management function is used for the purpose of controlling the PHY and gathering status from the PHY. The MII management frame format is shown as Figure 6.20-7.

	Management Frame Fields							
	PRE	ST	OP	PHYAD	REGAD	TA	DATA	IDLE
READ	1…1	01	10	AAAAA	RRRRR	Z0	DDDDDDDDDDDDDDDD	Z
WRITE	1…1	01	01	AAAAA	RRRRR	10	DDDDDDDDDDDDDDDD	Z

Figure 6.20-7 MII Management Frame Format

MII Management Function Configure Sequence

Read	**Write**
1. Set appropriate EMACn_MDCCR. 2. Set PHYAD and PHYRAD. 3. Set Write to 1‟b0 4. Set bit BUSY (EMACn_MIIDA[17]) to 1‟b1 to send a MII management frame out. 5. Wait BUSY (EMACn_MIIDA[17]) to become 1‟b0. 6. Read data from EMACn_MIID register. 7. Finish the read command.	1. Write data to EMACn_MIID register 2. Set appropriate EMACn_MDCCR. 3. Set PHYAD and PHYRAD. 4. Set Write to 1‟b1 5. Set bit BUSY (EMACn_MIIDA[17]) to 1‟b1 to send a MII management frame out. 6. Wait BUSY (EMACn_MIIDA[17]) to become 1‟b0. 7. Finish the write command.

Figure 6.20-8 MII Management Function Configure Sequence

EMAC n FIFO Threshold Control Register (EMACn_FFTCR)

The EMACn_FFTCR defines the high and low threshold of internal FIFOs, including TXFIFO and RXFIFO. The threshold of internal FIFOs is related to EMAC request generation and when the frame transmission starts. The EMACn_FFTCR also defines the burst length of AHB bus cycle for system memory access.

Register	Offset	R/W	Description	Reset Value
EMACn_FFTCR **n=0,1**	EMACn_BA+0x09C	R/W	EMAC n FIFO Threshold Control Register	0x0000_0101

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

Bits	Description	
[31:22]	**Reserved**	Reserved.
[21:20]	**BURSTLEN**	**DMA Burst Length** This defines the burst length of AHB bus cycle while EMAC accesses system memory. 00 = 4 words. 01 = 8 words. 10 = 16 words. 11 = 16 words.
[19:10]	**Reserved**	Reserved.

[9:8]	**TXTHD**	**TXFIFO Low Threshold** The TXTHD controls when TXDMA requests internal arbiter for data transfer between system memory and TXFIFO. The TXTHD defines not only the low threshold of TXFIFO, but also the high threshold. The high threshold is the twice of low threshold always. During the packet transmission, if the TXFIFO reaches the high threshold, the TXDMA stops generate request to transfer frame data from system memory to TXFIFO. If the frame data in TXFIFO is less than low threshold, TXDMA starts to transfer frame data from system memory to TXFIFO. The TXTHD also defines when the TXMAC starts to transmit frame out to network. The TXMAC starts to transmit the frame out while the TXFIFO first time reaches the high threshold during the transmission of the frame. If the frame data length is less than TXFIFO high threshold, the TXMAC starts to transmit the frame out after the frame data are all inside the TXFIFO. 00 = Undefined. 01 = TXFIFO low threshold is 64B and high threshold is 128B. 10 = TXFIFO low threshold is 80B and high threshold is 160B. 11 = TXFIFO low threshold is 96B and high threshold is 192B.
[7:2]	**Reserved**	Reserved.
[1:0]	**RXTHD**	**RXFIFO Low Threshold** The RXTHD controls when RXDMA requests internal arbiter for data transfer between RXFIFO and system memory. The RXTHD defines not only the high threshold of RXFIFO, but also the low threshold. The low threshold is the half of high threshold always. During the packet reception, if the RXFIFO reaches the high threshold, the RXDMA starts to transfer frame data from RXFIFO to system memory. If the frame data in RXFIFO is less than low threshold, RXDMA stops to transfer the frame data to system memory. 00 = Depend on the burst length setting. If the burst length is 8 words, high threshold is 8 words, too. 01 = RXFIFO high threshold is 64B and low threshold is 32B. 10 = RXFIFO high threshold is 128B and low threshold is 64B. 11 = RXFIFO high threshold is 192B and low threshold is 96B.

EMAC n Transmit Start Demand Register (EMACn_TSDR)

S/W issues a write command to EMACn_TSDR register to make TXDMA to leave Halt state and continue the frame transmission.

Register	Offset	R/W	Description	Reset Value
EMACn_TSDR **n=0,1**	EMACn_BA+0x0A0	W	EMAC n Transmit Start Demand Register	Undefined

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

Bits	Description	
[31:0]	**TSD**	**Transmit Start Demand** If the TX descriptor is not available for use of TXDMA after the TXON (EMACn_MCMDR[8]) is enabled, the FSM (Finite State Machine) of TXDMA enters the Halt state and the frame transmission is halted. After the S/W has prepared the new TX descriptor for frame transmission, it must issue a write command to EMACn_TSDR register to make TXDMA to leave Halt state and continue the frame transmission. The EMACn_TSDR is a write only register and the value read from this register is undefined. The write to EMACn_TSDR register takes effect only when TXDMA stayed at Halt state.

EMAC n Receive Start Demand Register (EMACn_RSDR)

S/W issues a write command to EMACn_RSDR register to make RXDMA to leave Halt state and continue the frame reception.

Register	Offset	R/W	Description	Reset Value
EMACn_RSDR **n=0,1**	EMACn_BA+0x0A4	W	EMAC n Receive Start Demand Register	Undefined

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

Bits	Description	
[31:0]	**RSD**	**Receive Start Demand** If the RX descriptor is not available for use of RXDMA after the RXON (EMACn_MCMDR[0]) is enabled, the FSM (Finite State Machine) of RXDMA enters the Halt state and the frame reception is halted. After the S/W has prepared the new RX descriptor for frame reception, it must issue a write command to EMACn_RSDR register to make RXDMA to leave Halt state and continue the frame reception. The EMACn_RSDR is a write only register and the value read from this register is undefined. The write to EMACn_RSDR register takes effect only when RXDMA stayed at Halt state.

EMAC n Maximum Receive Frame Control Register (EMACn_DMARFC)

The EMACn_DMARFC defines the maximum frame length for a received frame that can be stored in the system memory. It is recommend that only use this register while S/W wants to receive a frame which length is greater than 1518 bytes.

Register	Offset	R/W	Description	Reset Value
EMACn_DMARFC **n=0,1**	EMACn_BA+0x0A8	R/W	EMAC n Maximum Receive Frame Control Register	0x0000_0800

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**RXMS**	**Maximum Receive Frame Length** The RXMS defines the maximum frame length for received frame. If the frame length of received frame is greater than RXMS, and bit MFLEIEN (EMACn_MIEN[8]) is also enabled, the bit DFO (EMACn_MISTA[8]) is set and the RX interrupt is triggered. It is recommended that only use RXMS to qualify the length of received frame while S/W wants to receive a frame which length is greater than 1518 bytes.

EMAC n MAC Interrupt Enable Register (EMACn_MIEN)

The EMACn_MIEN controls the enable of EMAC interrupt status to generate interrupt. Two interrupts, RXINTR for frame reception and TXINTR for frame transmission, are generated from EMAC to CPU.

Register	Offset	R/W	Description	Reset Value
EMACn_MIEN **n=0,1**	EMACn_BA+0x0AC	R/W	EMAC n MAC Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			TSALMIEN	Reserved			TXBEIEN
23	22	21	20	19	18	17	16
TDUIEN	LCIEN	TXABTIEN	NCSIEN	EXDEFIEN	TXCPIEN	TXUDIEN	TXIEN
15	14	13	12	11	10	9	8
WOLIEN	CFRIEN	Reserved		RXBEIEN	RDUIEN	DENIEN	MFLEIEN
7	6	5	4	3	2	1	0
MMPIEN	RPIEN	ALIEIEN	RXGDIEN	LPIEN	RXOVIEN	CRCEIEN	RXIEN

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28]	**TSALMIEN**	**Time Stamp Alarm Interrupt Enable Bit** The TSALMIEN controls the TSALS (EMACn_MISTA[28]) interrupt generation. If TSALS (EMACn_MISTA[28]) is set, and both TSALMIEN and TXIEN (EMACn_MIEN[16]) enabled, the EMAC generates the TX interrupt to CPU. If TSALMIEN or TXIEN (EMACn_MIEN[16]) disabled, no TX interrupt generated to CPU even the TSALS (EMACn_MISTA[28]) is set. 0 = TSALS (EMACn_MISTA[28]) trigger TX interrupt Disabled. 1 = TSALS (EMACn_MISTA[28]) trigger TX interrupt Enabled.
[27:25]	**Reserved**	Reserved.
[24]	**TXBEIEN**	**Transmit Bus Error Interrupt Enable Bit** The TXBEIEN controls the TXBERR (EMACn_MISTA[24]) interrupt generation. If TXBERR (EMACn_MISTA[24]) is set, and both TXBEIEN and TXIEN (EMACn_MIEN[16]) are enabled, the EMAC generates the TX interrupt to CPU. If TXBEIEN or TXIEN (EMACn_MIEN[16]) is disabled, no TX interrupt is generated to CPU even the TXBERR (EMACn_MISTA[24]) is set. 0 = TXBERR (EMACn_MISTA[24]) trigger TX interrupt Disabled. 1 = TXBERR (EMACn_MISTA[24]) trigger TX interrupt Enabled.
[23]	**TDUIEN**	**Transmit Descriptor Unavailable Interrupt Enable Bit** The TDUIEN controls the TDU (EMACn_MISTA[23]) interrupt generation. If TDU (EMACn_MISTA[23]) is set, and both TDUIEN and TXIEN (EMACn_MIEN[16]) are enabled, the EMAC generates the TX interrupt to CPU. If TDUIEN or TXIEN (EMACn_MIEN[16]) is disabled, no TX interrupt is generated to CPU even the TDU (EMACn_MISTA[23]) is set. 0 = TDU (EMACn_MISTA[23]) trigger TX interrupt Disabled. 1 = TDU (EMACn_MISTA[23]) trigger TX interrupt Enabled.

[22]	**LCIEN**	**Late Collision Interrupt Enable Bit** The LCIEN controls the LC (EMACn_MISTA[22]) interrupt generation. If LC (EMACn_MISTA[22]) is set, and both LCIEN and TXIEN (EMACn_MIEN[16]) are enabled, the EMAC generates the TX interrupt to CPU. If LCIEN or TXIEN (EMACn_MIEN[16]) is disabled, no TX interrupt is generated to CPU even the LC (EMACn_MISTA[22]) is set. 0 = LC (EMACn_MISTA[22]) trigger TX interrupt Disabled. 1 = LC (EMACn_MISTA[22]) trigger TX interrupt Enabled.
[21]	**TXABTIEN**	**Transmit Abort Interrupt Enable Bit** The TXABTIEN controls the TXABT (EMACn_MISTA[21]) interrupt generation. If TXABT (EMACn_MISTA[21]) is set, and both TXABTIEN and TXIEN (EMACn_MIEN[16]) are enabled, the EMAC generates the TX interrupt to CPU. If TXABTIEN or TXIEN (EMACn_MIEN[16]) is disabled, no TX interrupt is generated to CPU even the TXABT (EMACn_MISTA[21]) is set. 0 = TXABT (EMACn_MISTA[21]) trigger TX interrupt Disabled. 1 = TXABT (EMACn_MISTA[21]) trigger TX interrupt Enabled.
[20]	**NCSIEN**	**No Carrier Sense Interrupt Enable Bit** The NCSIEN controls the NCS (EMACn_MISTA[20]) interrupt generation. If NCS (EMACn_MISTA[20]) is set, and both NCSIEN and TXIEN (EMACn_MIEN[16]) are enabled, the EMAC generates the TX interrupt to CPU. If NCSIEN or TXIEN (EMACn_MIEN[16]) is disabled, no TX interrupt is generated to CPU even the NCS (EMACn_MISTA[20]) is set. 0 = NCS (EMACn_MISTA[20]) trigger TX interrupt Disabled. 1 = NCS (EMACn_MISTA[20]) trigger TX interrupt Enabled.
[19]	**EXDEFIEN**	**Defer Exceed Interrupt Enable Bit** The EXDEFIEN controls the EXDEF (EMACn_MISTA[19]) interrupt generation. If EXDEF (EMACn_MISTA[19]) is set, and both EXDEFIEN and TXIEN (EMACn_MIEN[16]) are enabled, the EMAC generates the TX interrupt to CPU. If EXDEFIEN or TXIEN (EMACn_MIEN[16]) is disabled, no TX interrupt is generated to CPU even the EXDEF (EMACn_MISTA[19]) is set. 0 = EXDEF (EMACn_MISTA[19]) trigger TX interrupt Disabled. 1 = EXDEF (EMACn_MISTA[19]) trigger TX interrupt Enabled.
[18]	**TXCPIEN**	**Transmit Completion Interrupt Enable Bit** The TXCPIEN controls the TXCP (EMACn_MISTA[18]) interrupt generation. If TXCP (EMACn_MISTA[18]) is set, and both TXCPIEN and TXIEN (EMACn_MIEN[16]) are enabled, the EMAC generates the TX interrupt to CPU. If TXCPIEN or TXIEN (EMACn_MIEN[16]) is disabled, no TX interrupt is generated to CPU even the TXCP (EMACn_MISTA[18]) is set. 0 = TXCP (EMACn_MISTA[18]) trigger TX interrupt Disabled. 1 = TXCP (EMACn_MISTA[18]) trigger TX interrupt Enabled.
[17]	**TXUDIEN**	**Transmit FIFO Underflow Interrupt Enable Bit** The TXUDIEN controls the TXEMP (EMACn_MISTA[17]) interrupt generation. If TXEMP (EMACn_MISTA[17]) is set, and both TXUDIEN and TXIEN (EMACn_MIEN[16]) are enabled, the EMAC generates the TX interrupt to CPU. If TXUDIEN or TXIEN (EMACn_MIEN[16]) is disabled, no TX interrupt is generated to CPU even the TXEMP (EMACn_MISTA[17]) is set. 0 = TXEMP (EMACn_MISTA[17]) trigger TX interrupt Disabled. 1 = TXEMP (EMACn_MISTA[17]) trigger TX interrupt Enabled.

[16]	**TXIEN**	**Transmit Interrupt Enable Bit** The TXIEN controls the TX interrupt generation. If TXIEN is enabled and TXINTR (EMACn_MISTA[16]) is high, EMAC generates the TX interrupt to CPU. If TXIEN is disabled, no TX interrupt is generated to CPU even any status bit of EMACn_MISTA[24:17] is set and the corresponding bit of EMACn_MIEN is enabled. In other words, if S/W wants to receive TX interrupt from EMAC, this bit must be enabled. And, if S/W doesn‟t want to receive any TX interrupt from EMAC, disables this bit. 0 = TXINTR (EMACn_MISTA[16]) is masked and TX interrupt generation Dsiabled. 1 = TXINTR (EMACn_MISTA[16]) is not masked and TX interrupt generation Enabled.
[15]	**WOLIEN**	**Magic Packet Receive Interrupt Enable Bit** The WOLIEN controls the MPR (EMACn_MISTA[15]) interrupt generation. If MPR (EMACn_MISTA[15]) is set, and both WOLIEN and RXIEN (EMACn_MIEN[0]) are enabled, the EMAC generates the RX interrupt to CPU. If WOLIEN or RXIEN (EMACn_MIEN[0]) is disabled, no RX interrupt is generated to CPU even the MPR (EMACn_MISTA[15]) is set. 0 = MPR (EMACn_MISTA[15]) trigger RX interrupt Disabled. 1 = MPR (EMACn_MISTA[15]) trigger RX interrupt Enabled.
[14]	**CFRIEN**	**Control Frame Receive Interrupt Enable Bit** The CFRIEN controls the CFR (EMACn_MISTA[14]) interrupt generation. If CFR (EMACn_MISTA[14]) is set, and both CFRIEN and RXIEN (EMACn_MIEN[0]) are enabled, the EMAC generates the RX interrupt to CPU. If CFRIEN or RXIEN (EMACn_MIEN[0]) is disabled, no RX interrupt is generated to CPU even the CFR (EMACn_MISTA[14]) is set. 0 = CFR (EMACn_MISTA[14]) trigger RX interrupt Disabled. 1 = CFR (EMACn_MISTA[14]) trigger RX interrupt Enabled.
[13:12]	**Reserved**	Reserved.
[11]	**RXBEIEN**	**Receive Bus Error Interrupt Enable Bit** The RXBEIEN controls the RXBERR (EMACn_MISTA[11]) interrupt generation. If RXBERR (EMACn_MISTA[11]) is set, and both RXBEIEN and RXIEN (EMACn_MIEN[0]) are enabled, the EMAC generates the RX interrupt to CPU. If RXBEIEN or RXIEN (EMACn_MIEN[0]) is disabled, no RX interrupt is generated to CPU even the RXBERR (EMACn_MISTA[11]) is set. 0 = RXBERR (EMACn_MISTA[11]) trigger RX interrupt Disabled. 1 = RXBERR (EMACn_MISTA[11]) trigger RX interrupt Enabled.
[10]	**RDUIEN**	**Receive Descriptor Unavailable Interrupt Enable Bit** The RDUIEN controls the RDU (EMACn_MISTA[10]) interrupt generation. If RDU (EMACn_MISTA[10]) is set, and both RDUIEN and RXIEN (EMACn_MIEN[0]) are enabled, the EMAC generates the RX interrupt to CPU. If RDUIEN or RXIEN (EMACn_MIEN[0]) is disabled, no RX interrupt is generated to CPU even the RDU (EMACn_MISTA[10]) is set. 0 = RDU (EMACn_MISTA[10]) trigger RX interrupt Disabled. 1 = RDU (EMACn_MISTA[10]) trigger RX interrupt Enabled.
[9]	**DENIEN**	**DMA Early Notification Interrupt Enable Bit** The DENIEN controls the DEN (EMACn_MISTA[9]) interrupt generation. If DEN (EMACn_MISTA[9]) is set, and both DENIEN and RXIEN (EMACn_MIEN[0]) are enabled, the EMAC generates the RX interrupt to CPU. If DENIEN or RXIEN (EMACn_MIEN[0]) is disabled, no RX interrupt is generated to CPU even the DEN (EMACn_MISTA[9]) is set. 0 = DEN (EMACn_MISTA[9]) trigger RX interrupt Disabled. 1 = DEN (EMACn_MISTA[9]) trigger RX interrupt Enabled.

[8]	**MFLEIEN**	**Maximum Frame Length Exceed Interrupt Enable Bit** The MFLEIEN controls the DFO (EMACn_MISTA[8]) interrupt generation. If DFO (EMACn_MISTA[8]) is set, and both MFLEIEN and RXIEN (EMACn_MIEN[0]) are enabled, the EMAC generates the RX interrupt to CPU. If MFLEIEN or RXIEN (EMACn_MIEN[0]) is disabled, no RX interrupt is generated to CPU even the DFO (EMACn_MISTA[8]) is set. 0 = DFO (EMACn_MISTA[8]) trigger RX interrupt Disabled. 1 = DFO (EMACn_MISTA[8]) trigger RX interrupt Enabled.
[7]	**MMPIEN**	**Miss Packet Counter Overrun Interrupt Enable Bit** The MMPIEN controls the MMP (EMACn_MISTA[7]) interrupt generation. If MMP (EMACn_MISTA[7]) is set, and both MMPIEN and RXIEN (EMACn_MIEN[0]) are enabled, the EMAC generates the RX interrupt to CPU. If MMPIEN or RXIEN (EMACn_MIEN[0]) is disabled, no RX interrupt is generated to CPU even the MMP (EMACn_MISTA[7]) is set. 0 = MMP (EMACn_MISTA[7]) trigger RX interrupt Disabled. 1 = MMP (EMACn_MISTA[7]) trigger RX interrupt Enabled.
[6]	**RPIEN**	**Runt Packet Interrupt Enable Bit** The RPIEN controls the RP (EMACn_MISTA[6]) interrupt generation. If RP (EMACn_MISTA[6]) is set, and both RPIEN and RXIEN (EMACn_MIEN[0]) are enabled, the EMAC generates the RX interrupt to CPU. If RPIEN or RXIEN (EMACn_MIEN[0]) is disabled, no RX interrupt is generated to CPU even the RP (EMACn_MISTA[6]) is set. 0 = RP (EMACn_MISTA[6]) trigger RX interrupt Disabled. 1 = RP (EMACn_MISTA[6]) trigger RX interrupt Enabled.
[5]	**ALIEIEN**	**Alignment Error Interrupt Enable Bit** The ALIEIEN controls the ALIE (EMACn_MISTA[5]) interrupt generation. If ALIE (EMACn_MISTA[5]) is set, and both ALIEIEN and RXIEN (EMACn_MIEN[0]) are enabled, the EMAC generates the RX interrupt to CPU. If ALIEIEN or RXIEN (EMACn_MIEN[0]) is disabled, no RX interrupt is generated to CPU even the ALIE (EMACn_MISTA[5]) is set. 0 = ALIE (EMACn_MISTA[5]) trigger RX interrupt Disabled. 1 = ALIE (EMACn_MISTA[5]) trigger RX interrupt Enabled.
[4]	**RXGDIEN**	**Receive Good Interrupt Enable Bit** The RXGDIEN controls the RXGD (EMACn_MISTA[4]) interrupt generation. If RXGD (EMACn_MISTA[4]) is set, and both RXGDIEN and RXIEN (EMACn_MIEN[0]) are enabled, the EMAC generates the RX interrupt to CPU. If RXGDIEN or RXIEN (EMACn_MIEN[0]) is disabled, no RX interrupt is generated to CPU even the RXGD (EMACn_MISTA[4]) is set. 0 = RXGD (EMACn_MISTA[4]) trigger RX interrupt Disabled. 1 = RXGD (EMACn_MISTA[4]) trigger RX interrupt Enabled.
[3]	**LPIEN**	**Long Packet Interrupt Enable Bit** The LPIEN controls the PTLE (EMACn_MISTA[3]) interrupt generation. If PTLE (EMACn_MISTA[3]) is set, and both LPIEN and RXIEN (EMACn_MIEN[0]) are enabled, the EMAC generates the RX interrupt to CPU. If LPIEN or RXIEN (EMACn_MIEN[0]) is disabled, no RX interrupt is generated to CPU even the PTLE (EMACn_MISTA[3]) is set. 0 = PTLE (EMACn_MISTA[3]) trigger RX interrupt Disabled. 1 = PTLE (EMACn_MISTA[3]) trigger RX interrupt Enabled.

[2]	**RXOVIEN**	**Receive FIFO Overflow Interrupt Enable Bit** The RXOVIEN controls the RXOV (EMACn_MISTA[2]) interrupt generation. If RXOV (EMACn_MISTA[2]) is set, and both RXOVIEN and RXIEN (EMACn_MIEN[0]) are enabled, the EMAC generates the RX interrupt to CPU. If RXOVIEN or RXIEN (EMACn_MIEN[0]) is disabled, no RX interrupt is generated to CPU even the RXOV (EMACn_MISTA[2]) is set. 0 = RXOV (EMACn_MISTA[2]) trigger RX interrupt Disabled. 1 = RXOV (EMACn_MISTA[2]) trigger RX interrupt Enabled.
[1]	**CRCEIEN**	**CRC Error Interrupt Enable Bit** The CRCEIEN controls the CRCE (EMACn_MISTA[1]) interrupt generation. If CRCE (EMACn_MISTA[1]) is set, and both CRCEIEN and RXIEN (EMACn_MIEN[0]) are enabled, the EMAC generates the RX interrupt to CPU. If CRCEIEN or RXIEN (EMACn_MIEN[0]) is disabled, no RX interrupt is generated to CPU even the CRCE (EMACn_MISTA[1]) is set. 0 = CRCE (EMACn_MISTA[1]) trigger RX interrupt Disabled. 1 = CRCE (EMACn_MISTA[1]) trigger RX interrupt Enabled.
[0]	**RXIEN**	**Receive Interrupt Enable Bit** The RXIEN controls the RX interrupt generation. If RXIEN is enabled and RXINTR (EMACn_MISTA[0]) is high, EMAC generates the RX interrupt to CPU. If RXIEN is disabled, no RX interrupt is generated to CPU even any status bit EMACn_MISTA[15:1] is set and the corresponding bit of EMACn_MIEN is enabled. In other words, if S/W wants to receive RX interrupt from EMAC, this bit must be enabled. And, if S/W doesn"t want to receive any RX interrupt from EMAC, disables this bit. 0 = RXINTR (EMACn_MISTA[0]) is masked and RX interrupt generation Disabled. 1 = RXINTR (EMACn_MISTA[0]) is not masked and RX interrupt generation Enabled.

EMAC n MAC Interrupt Status Register (EMACn_MISTA)

The EMACn_MISTA keeps much EMAC statuses, such as frame transmission, reception status and internal FIFO status. The statuses kept in EMACn_MISTA will trigger the reception or transmission interrupt. The EMACn_MISTA is a write clear register and write 1 to corresponding bit clears the status and also clears the interrupt.

Register	Offset	R/W	Description	Reset Value
EMACn_MISTA **n=0,1**	EMACn_BA+0x0B0	R/W	EMAC n MAC Interrupt Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			TSALS	Reserved			TXBERR
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TDU	LC	TXABT	NCS	EXDEF	TXCP	TXEMP	TXINTR
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
MGPR	CFR	Reserved		RXBERR	RDU	DEN	DFO
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MMP	RP	ALIE	RXGD	PTLE	RXOV	CRCE	RXINTR

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28]	**TSALS**	**Time Stamp Alarm Interrupt** The TSALS high indicates the EMACn_TSMSR register value equals to EMACn_TSMSAR register and EMACn_TSLSR register value equals to register EMACn_TSMLSR. If TSALS is high and TXALMIEN (EMACn_MIEN[28]) enabled, the TXINTR will be high. Write 1 to this bit clears the TSALS status. 0 = EMACn_TSMSR did not equal EMACn_TSMSAR or EMACn_TSLSR did not equal EMACn_TSLSAR. 1 = EMACn_TSMSR equals EMACn_TSMSAR and EMACn_TSLSR equals EMACn_TSLSAR.
[27:25]	**Reserved**	Reserved.
[24]	**TXBERR**	**Transmit Bus Error Interrupt** The TXBERR high indicates the memory controller replies ERROR response while EMAC access system memory through TXDMA during packet transmission process. Reset EMAC is recommended while TXBERR status is high. If the TXBERR is high and TXBEIEN (EMACn_MIEN[24]) is enabled, the TXINTR will be high. Write 1 to this bit clears the TXBERR status. 0 = No ERROR response is received. 1 = ERROR response is received.

[23]	**TDU**	**Transmit Descriptor Unavailable Interrupt** The TDU high indicates that there is no available TX descriptor for packet transmission and TXDMA will stay at Halt state. Once, the TXDMA enters the Halt state, S/W must issues a write command to TSDR register to make TXDMA leave Halt state while new TX descriptor is available. If the TDU is high and TDUIEN (EMACn_MIEN[23]) is enabled, the TXINTR will be high. Write 1 to this bit clears the TDU status. 0 = TX descriptor is available. 1 = TX descriptor is unavailable.
[22]	**LC**	**Late Collision Interrupt** The LC high indicates the collision occurred in the outside of 64 bytes collision window. This means after the 64 bytes of a frame has been transmitted out to the network, the collision still occurred. The late collision check will only be done while EMAC is operating on half-duplex mode. If the LC is high and LCIEN (EMACn_MIEN[22]) is enabled, the TXINTR will be high. Write 1 to this bit clears the LC status. 0 = No collision occurred in the outside of 64 bytes collision window. 1 = Collision occurred in the outside of 64 bytes collision window.
[21]	**TXABT**	**Transmit Abort Interrupt** The TXABT high indicates the packet incurred 16 consecutive collisions during transmission, and then the transmission process for this packet is aborted. The transmission abort is only available while EMAC is operating on half-duplex mode. If the TXABT is high and TXABTIEN (EMACn_MIEN[21]) is enabled, the TXINTR will be high. Write 1 to this bit clears the TXABT status. 0 = Packet does not incur 16 consecutive collisions during transmission. 1 = Packet incurred 16 consecutive collisions during transmission.
[20]	**NCS**	**No Carrier Sense Interrupt** The NCS high indicates the MII I/F signal CRS does not active at the start of or during the packet transmission. The NCS is only available while EMAC is operating on half-duplex mode. If the NCS is high and NCSIEN (EMACn_MIEN[20]) is enabled, the TXINTR will be high. Write 1 to this bit clears the NCS status. 0 = CRS signal actives correctly. 1 = CRS signal does not active at the start of or during the packet transmission.
[19]	**EXDEF**	**Defer Exceed Interrupt** The EXDEF high indicates the frame waiting for transmission has deferred over 0.32768ms on 100Mbps mode, or 3.2768ms on 10Mbps mode. The deferral exceed check will only be done while bit NDEF of MCMDR is disabled, and EMAC is operating on half-duplex mode. If the EXDEF is high and EXDEFIEN (EMACn_MIEN[19]) is enabled, the TXINTR will be high. Write 1 to this bit clears the EXDEF status. 0 = Frame waiting for transmission has not deferred over 0.32768ms (100Mbps) or 3.2768ms (10Mbps). 1 = Frame waiting for transmission has deferred over 0.32768ms (100Mbps) or 3.2768ms (10Mbps).
[18]	**TXCP**	**Transmit Completion Interrupt** The TXCP indicates the packet transmission has completed correctly. If the TXCP is high and TXCPIEN (EMACn_MIEN[18]) is enabled, the TXINTR will be high. Write 1 to this bit clears the TXCP status. 0 = The packet transmission not completed. 1 = The packet transmission has completed.

[17]	**TXEMP**	**Transmit FIFO Underflow Interrupt** The TXEMP high indicates the TXFIFO underflow occurred during packet transmission. While the TXFIFO underflow occurred, the EMAC will retransmit the packet automatically without S/W intervention. If the TXFIFO underflow occurred often, it is recommended that modify TXFIFO threshold control, the TXTHD of FFTCR register, to higher level. If the TXEMP is high and TXUDIEN (EMACn_MIEN[17]) is enabled, the TXINTR will be high. Write 1 to this bit clears the TXEMP status. 0 = No TXFIFO underflow occurred during packet transmission. 1 = TXFIFO underflow occurred during packet transmission.
[16]	**TXINTR**	**Transmit Interrupt** The TXINTR indicates the TX interrupt status. If TXINTR high and its corresponding enable bit, TXIEN (EMACn_MIEN[16]), is also high indicates the EMAC generates TX interrupt to CPU. If TXINTR is high but TXIEN (EMACn_MIEN[16]) is disabled, no TX interrupt is generated. The TXINTR is logic OR result of bit logic AND result of EMACn_MISTA[28:17] and EMACn_MIEN[28:17]. In other words, if any bit of EMACn_MISTA[28:17] is high and its corresponding enable bit in EMACn_MIEN[28:17] is also enabled, the TXINTR will be high. Because the TXINTR is a logic OR result, clears EMC_MISTA[28:17] makes TXINTR be cleared, too. 0 = No status bit in EMACn_MISTA[28:17] is set or no enable bit in EMACn_MIEN[28:17] is enabled. 1 = At least one status in EMACn_MISTA[28:17] is set and its corresponding enable bit in EMACn_MIEN[28:17] is enabled, too.
[15]	**MGPR**	**Magic Packet Received Interrupt** The MPR high indicates EMAC receives a Magic Packet. The CFR only available while system is in Power-down mode and MGP_WAKE is set high. If the MPR is high and WOLIEN (EMACn_MIEN[15]) is enabled, the RXINTR will be high. Write 1 to this bit clears the MPR status. 0 = The EMAC does not receive the Magic Packet. 1 = The EMAC receives a Magic Packet.
[14]	**CFR**	**Control Frame Receive Interrupt** The CFR high indicates EMAC receives a flow control frame. The CFR only available while EMAC is operating on full duplex mode. If the CFR is high and CFRIEN (EMACn_MIEN[14]) is enabled, the RXINTR will be high. Write 1 to this bit clears the CFR status. 0 = The EMAC does not receive the flow control frame. 1 = The EMAC receives a flow control frame.
[13:12]	**Reserved**	Reserved.
[11]	**RXBERR**	**Receive Bus Error Interrupt** The RXBERR high indicates the memory controller replies ERROR response while EMAC access system memory through RXDMA during packet reception process. Reset EMAC is recommended while RXBERR status is high. If the RXBERR is high and RXBEIEN (EMACn_MIEN[11]) is enabled, the RXINTR will be high. Write 1 to this bit clears the RXBERR status. 0 = No ERROR response is received. 1 = ERROR response is received.

[10]	**RDU**	**Receive Descriptor Unavailable Interrupt** The RDU high indicates that there is no available RX descriptor for packet reception and RXDMA will stay at Halt state. Once, the RXDMA enters the Halt state, S/W must issues a write command to RSDR register to make RXDMA leave Halt state while new RX descriptor is available. If the RDU is high and RDUIEN (EMACn_MIEN[10]) is enabled, the RXINTR will be high. Write 1 to this bit clears the RDU status. 0 = RX descriptor is available. 1 = RX descriptor is unavailable.
[9]	**DEN**	**DMA Early Notification Interrupt** The DEN high indicates the EMAC has received the Length/Type field of the incoming packet. If the DEN is high and DENIEN (EMACn_MIEN[9]) is enabled, the RXINTR will be high. Write 1 to this bit clears the DEN status. 0 = The Length/Type field of incoming packet has not received yet. 1 = The Length/Type field of incoming packet has received.
[8]	**DFO**	**Maximum Frame Length Interrupt** The DFO high indicates the length of the incoming packet has exceeded the length limitation configured in DMARFC register and the incoming packet is dropped. If the DFO is high and MFLEIEN (EMACn_MIEN[8]) is enabled, the RXINTR will be high. Write 1 to this bit clears the DFO status. 0 **=** The length of the incoming packet doesn"t exceed the length limitation configured in DMARFC. 1 = The length of the incoming packet has exceeded the length limitation configured in DMARFC.
[7]	**MMP**	**More Missed Packet Interrupt** The MMP high indicates the MPCNT, Missed Packet Count, has overflow. If the MMP is high and MMPIEN (EMACn_MIEN[7]) is enabled, the RXINTR will be high. Write 1 to this bit clears the MMP status. 0 = The MPCNT has not rolled over yet. 1 = The MPCNT has rolled over yet.
[6]	**RP**	**Runt Packet Interrupt** The RP high indicates the length of the incoming packet is less than 64 bytes and the packet is dropped. If the ARP (EMACn_MCMDR[2]) is set, the short packet is regarded as a good packet and RP will not be set. If the RP is high and RPIEN (EMACn_MIEN[6]) is enabled, the RXINTR will be high. Write 1 to this bit clears the RP status. 0 = The incoming frame is not a short frame or S/W wants to receive a short frame. 1 = The incoming frame is a short frame and dropped.
[5]	**ALIE**	**Alignment Error Interrupt** The ALIE high indicates the length of the incoming frame is not a multiple of byte. If the ALIE is high and ALIEIEN (EMACn_MIEN[5]) is enabled, the RXINTR will be high. Write 1 to this bit clears the ALIE status. 0 = The frame length is a multiple of byte. 1 = The frame length is not a multiple of byte.
[4]	**RXGD**	**Receive Good Interrupt** The RXGD high indicates the frame reception has completed. If the RXGD is high and RXGDIEN (EMACn_MIEN[4]) is enabled, the RXINTR will be high. Write 1 to this bit clears the RXGD status. 0 = The frame reception has not complete yet. 1 = The frame reception has completed.

[3]	**PTLE**	**Packet Too Long Interrupt** The PTLE high indicates the length of the incoming packet is greater than 1518 bytes and the incoming packet is dropped. If the ALP (EMACn_MCMDR[1]) is set, the long packet will be regarded as a good packet and PTLE will not be set. If the PTLE is high and LPIEN(EMACn_MIEN[3]) is enabled, the RXINTR will be high. Write 1 to this bit clears the PTLE status. 0 = The incoming frame is not a long frame or S/W wants to receive a long frame. 1 = The incoming frame is a long frame and dropped.
[2]	**RXOV**	**Receive FIFO Overflow Interrupt** The RXOV high indicates the RXFIFO overflow occurred during packet reception. While the RXFIFO overflow occurred, the EMAC drops the current receiving packer. If the RXFIFO overflow occurred often, it is recommended that modify RXFIFO threshold control, the RXTHD of FFTCR register, to higher level. If the RXOV is high and RXOVIEN (EMACn_MIEN[2]) is enabled, the RXINTR will be high. Write 1 to this bit clears the RXOV status. 0 = No RXFIFO overflow occurred during packet reception. 1 = RXFIFO overflow occurred during packet reception.
[1]	**CRCE**	**CRC Error Interrupt** The CRCE high indicates the incoming packet incurred the CRC error and the packet is dropped. If the AEP (EMACn_MCMDR[4]) is set, the CRC error packet will be regarded as a good packet and CRCE will not be set. If the CRCE is high and CRCEIEN (EMACn_MIEN[1]) is enabled, the RXINTR will be high. Write 1 to this bit clears the CRCE status. 0 = The frame does not incur CRC error. 1 = The frame incurred CRC error.
[0]	**RXINTR**	**Receive Interrupt** The RXINTR indicates the RX interrupt status. If RXINTR high and its corresponding enable bit, RXIEN (EMACn_MIEN[0]), is also high indicates the EMAC generates RX interrupt to CPU. If RXINTR is high but RXIEN (EMACn_MIEN[0]) is disabled, no RX interrupt is generated. The RXINTR is logic OR result of bit logic AND result of EMACn_MISTA[15:1] and EMACn_MIEN[15:1]. In other words, if any bit of EMACn_MISTA[15:1] is high and its corresponding enable bit in EMACn_MIEN[15:1] is also enabled, the RXINTR will be high. Because the RXINTR is a logic OR result, clears EMACn_MISTA[15:1] makes RXINTR be cleared, too. 0 = No status bit in EMACn_MISTA[15:1] is set or no enable bit in EMACn_MIEN[15:1] is enabled. 1 = At least one status in EMACn_MISTA[15:1] is set and its corresponding enable bit in EMACn_MIEN[15:1] is enabled, too.

EMAC n MAC General Status Register (EMACn_MGSTA)

The EMACn_MGSTA also keeps the statuses of EMAC. But the statuses in the EMACn_MGSTA will not trigger any interrupt. The EMACn_MGSTA is a write clear register and write 1 to corresponding bit clears the status.

Register	Offset	R/W	Description	Reset Value
EMACn_MGSTA **n=0,1**	EMACn_BA+0x0B4	R/W	EMAC n MAC General 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			**RPAU**	**TXHA**	**SQE**	**PAU**	**DEF**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCNT				**Reserved**	**RXFFULL**	**RXHA**	**CFR**

Bits	Description	
[31:13]	**Reserved**	Reserved.
[12]	**RPAU**	**Remote Pause Status** The RPAU indicates that remote pause counter down counting actives. After Ethernet MAC controller sent PAUSE frame out successfully, it starts the remote pause counter down counting. When this bit high, it’s predictable that remote Ethernet MAC controller wouldn’t start the packet transmission until the down counting done. 0 = Remote pause counter down counting done. 1 = Remote pause counter down counting actives.
[11]	**TXHA**	**Transmission Halted** The TXHA high indicates the next normal packet transmission process will be halted because the bit TXON (EMACn_MCMDR[8]) is disabled be S/W. 0 = Next normal packet transmission process will go on. 1 = Next normal packet transmission process will be halted.
[10]	**SQE**	**Signal Quality Error** The SQE high indicates the SQE error found at end of packet transmission on 10Mbps half-duplex mode. The SQE error check will only be done while both bit EnSQE (EMACn_MCMDR[17]) is enabled and EMAC is operating on 10Mbps half-duplex mode. 0 = No SQE error found at end of packet transmission. 1 = SQE error found at end of packet transmission.

[9]	**PAU**	**Transmission Paused** The PAU high indicates the next normal packet transmission process will be paused temporally because EMAC received a PAUSE control frame. 0 = Next normal packet transmission process will go on. 1 = Next normal packet transmission process will be paused.
[8]	**DEF**	**Deferred Transmission** The DEF high indicates the packet transmission has deferred once. The DEF is only available while EMAC is operating on half-duplex mode. 0 = Packet transmission does not defer. 1 = Packet transmission has deferred once.
[7:4]	**CCNT**	**Collision Count** The CCNT indicates that how many collisions occurred consecutively during a packet transmission. If the packet incurred 16 consecutive collisions during transmission, the CCNT will be 4"h0 and bit TXABT will be set to 1.
[3]	**Reserved**	Reserved.
[2]	**RXFFULL**	**RXFIFO Full** The RXFFULL indicates the RXFIFO is full due to four 64-byte packets are kept in RXFIFO and the following incoming packet will be dropped. 0 = The RXFIFO is not full. 1 = The RXFIFO is full and the following incoming packet will be dropped.
[1]	**RXHA**	**Receive Halted** The RXHA high indicates the next normal packet reception process will be halted because the bit RXON of MCMDR is disabled be S/W. 0 = Next normal packet reception process will go on. 1 = Next normal packet reception process will be halted.
[0]	**CFR**	**Control Frame Received** The CFR high indicates EMAC receives a flow control frame. The CFR only available while EMAC is operating on full duplex mode. 0 = The EMAC does not receive the flow control frame. 1 = The EMAC receives a flow control frame.

EMAC n Missed Packet Count Register (EMACn_MPCNT)

The EMACn_MPCNT keeps the number of packets that were dropped due to various types of receive errors. The EMACn_MPCNT is a read clear register. In addition, S/W also can write an initial value to EMACn_MPCNT and the missed packet counter will start counting from that initial value. If the missed packet counter is overflow, the MMP (EMACn_MISTA[7]) will be set.

Register	Offset	R/W	Description	Reset Value
EMACn_MPCNT **n=0,1**	EMACn_BA+0x0B8	R/W	EMAC n Missed Packet Count Register	0x0000_7FFF

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**MPC**	**Miss Packet Count** The MPC indicates the number of packets that were dropped due to various types of receive errors. The following type of receiving error makes missed packet counter increase: 1. Incoming packet is incurred RXFIFO overflow. 2. Incoming packet is dropped due to RXON is disabled. 3. Incoming packet is incurred CRC error.

EMAC n MAC Receive Pause Count Register (EMACn_MRPC)

The EMAC supports the PAUSE control frame reception and recognition. If EMAC received a PAUSE control frame, the operand field of the PAUSE control frame will be extracted and stored in the EMACn_MRPC register. The EMACn_MRPC register will keep the same while TX of EMAC is pausing due to the PAUSE control frame is received. The EMACn_MRPC is read only and write to this register has no effect.

Register	Offset	R/W	Description	Reset Value
EMACn_MRPC **n=0,1**	EMACn_BA+0x0BC	R	EMAC n MAC Receive Pause Count Register	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**MRPC**	**MAC Receive Pause Count** The MRPC keeps the operand field of the PAUSE control frame. It indicates how many slot time (512 bit time) the TX of EMAC will be paused.

EMAC n DMA Receive Frame Status Register (EMACn_DMARFS)

The EMACn_DMARFS is used to keep the Length/Type field of each incoming Ethernet packet.

Register	Offset	R/W	Description	Reset Value
EMACn_DMARFS **n=0,1**	EMACn_BA+0x0C8	R/W	EMAC n DMA Receive Frame 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**
RXFLT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RXFLT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**RXFLT**	**Receive Frame Length/Type** The RXFLT keeps the Length/Type field of each incoming Ethernet packet. If the bit DENIEN (EMACn_MIEN[9]) is enabled and the Length/Type field of incoming packet has received, the bit DEN (EMACn_MISTA[9]) will be set and trigger interrupt. And, the content of Length/Type field will be stored in RXFLT.

NUC980 SERIES TECHNICAL REFERENCE MANUAL

EMAC n Current Transmit Descriptor Start Address Register (EMACn_CTXDSA)

Register	Offset	R/W	Description	Reset Value
EMACn_CTXDSA **n=0,1**	EMACn_BA+0x0CC	R	EMAC n Current Transmit Descriptor Start Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**CTXDSA**	**Current Transmit Descriptor Start Address** The CTXDSA keeps the start address of TX descriptor that is used by TXDMA currently. The EMACn_CTXDSA is read only and write to this register has no effect.

EMAC n Current Transmit Buffer Start Address Register (EMACn_CTXBSA)

Register	Offset	R/W	Description	Reset Value
EMACn_CTXBSA **n=0,1**	EMACn_BA+0x0D0	R	EMAC n Current Transmit Buffer Start Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**CTXBSA**	**Current Transmit Buffer Start Address** The CTXDSA keeps the start address of TX frame buffer that is used by TXDMA currently. The EMACn_CTXBSA is read only and write to this register has no effect.

EMAC n Current Receive Descriptor Start Address Register (EMACn_CRXDSA)

Register	Offset	R/W	Description	Reset Value
EMACn_CRXDSA **n=0,1**	EMACn_BA+0x0D4	R	EMAC n Current Receive Descriptor Start Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**CRXDSA**	**Current Receive Descriptor Start Address** The CRXDSA keeps the start address of RX descriptor that is used by RXDMA currently. The EMACn_CRXDSA is read only and write to this register has no effect.

EMAC n Current Receive Buffer Start Address Register (EMACn_CRXBSA)

Register	Offset	R/W	Description	Reset Value
EMACn_CRXBSA **n=0,1**	EMACn_BA+0x0D8	R	EMAC n Current Receive Buffer Start Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**CRXBSA**	**Current Receive Buffer Start Address** The CRXBSA keeps the start address of RX frame buffer that is used by RXDMA currently. The EMACn_CRXBSA is read only and write to this register has no effect.

EMAC n Time Stamp Control Register (EMACn_TSCTL)

Register	Offset	R/W	Description	Reset Value
EMACn_TSCTL **n=0,1**	EMACn_BA+0x100	R/W	EMAC n Time Stamp 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		**TSALMEN**	**Reserved**	**TSUPDATE**	**TSMODE**	**TSIEN**	**TSEN**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5]	**TSALMEN**	**Time Stamp Alarm Enable Bit** Set this bit high enable Ethernet MAC controller to set TSALS (EMACn_MISTA[28]) high when EMACn_TSSEC equals to EMACn_ALMSEC and EMACn_TSSUBSEC equals to EMACn_ALMSSEC. 0 = Alarm Disabled when EMACn_TSSEC equals to EMACn_ALMSEC and EMACn_TSSUBSEC equals to EMACn_ALMSSEC. 1 = Alarm Enabled when EMACn_TSSEC equals to EMACn_ALMSEC and EMACn_TSSUBSEC equals to EMACn_ALMSSEC.
[4]	**Reserved**	Reserved.
[3]	**TSUPDATE**	**Time Stamp Counter Time Update Enable Bit** Set this bit high enables Ethernet MAC controller to add value of register EMACn_UPDSEC and EMACn_UPDSSEC to PTP time stamp counter. After the add operation finished, Ethernet MAC controller clear this bit to low automatically. 0 = No action. 1 = EMACn_UPDSEC updated to EMACn_TSSEC and EMACn_UPDSSEC updated to EMACn_TSSUBSEC.
[2]	**TSMODE**	**Time Stamp Fine Update Enable Bit** This bit chooses the time stamp counter update mode. 0 = Time stamp counter is in coarse update mode. 1 = Time stamp counter is in fine update mode.

[1]	**TSIEN**	**Time Stamp Counter Initialization Enable Bit** Set this bit high enables Ethernet MAC controller to load value of register EMACn_UPDSEC and EMACn_UPDSSEC to PTP time stamp counter. After the load operation finished, Ethernet MAC controller clear this bit to low automatically. 0 = Time stamp counter initialization done. 1 = Time stamp counter initialization Enabled.
[0]	**TSEN**	**Time Stamp Function Enable Bit** This bit controls if the IEEE 1588 PTP time stamp function is enabled or not. Set this bit high to enable IEEE 1588 PTP time stamp function while set this bit low to disable IEEE 1588 PTP time stamp function. 0 = IEEE 1588 PTP time stamp function Disabled. 1 = IEEE 1588 PTP time stamp function Enabled.

EMAC n Time Stamp Counter Second Register (EMACn_TSSEC)

Register	Offset	R/W	Description	Reset Value
EMACn_TSSEC **n=0,1**	EMACn_BA+0x110	R	EMAC n Time Stamp Counter Second Register	0x0000_0000

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

Bits	Description	
[31:0]	**SEC**	**Time Stamp Counter Second** This register reflects the bit [63:32] value of 64-bit reference timing counter. This 32-bit value is used as the second part of time stamp when TSEN (EMACn_TSCTL[0]) is high.

EMAC n Time Stamp Counter Sub Second Register (EMACn_TSSUBSEC)

Register	Offset	R/W	Description	Reset Value
EMACn_TSSUBSEC **n=0,1**	EMACn_BA+0x114	R	EMAC n Time Stamp Counter Sub Second Register	0x0000_0000

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

Bits	Description	
[31:0]	**SUBSEC**	**Time Stamp Counter Sub-second** This register reflects the bit [31:0] value of 64-bit reference timing counter. This 32-bit value is used as the sub-second part of time stamp when TSEN (EMACn_TSCTL[0]) is high.

EMAC n Time Stamp Increment Register (EMACn_TSINC)

Register	Offset	R/W	Description	Reset Value
EMACn_TSINC **n=0,1**	EMACn_BA+0x118	R/W	EMAC n Time Stamp Increment Register	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**CNTINC**	**Time Stamp Counter Increment** Time stamp counter increment value. If TSEN (EMACn_TSCTL[0]) is high, EMAC adds EMACn_TSSUBSEC with this 8-bit value every time when it wants to increase the EMACn_TSSUBSEC value.

EMAC n Time Stamp Addend Register (EMACn_TSADDEND)

Register	Offset	R/W	Description	Reset Value
EMACn_TSADDEND **n=0,1**	EMACn_BA+0x11C	R/W	EMAC n Time Stamp Addend Register	0x0000_0000

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

Bits	Description	
[31:0]	**ADDEND**	**Time Stamp Counter Addend** This register keeps a 32-bit value for accumulator to enable increment of EMACn_TSSUBSEC. If TSEN (EMACn_TSCTL[0]) and TSMODE (EMACn_TSCTL[2]) are both high, EMAC increases accumulator with this 32-bit value in each HCLK. Once the accumulator is overflow, it generates an enable to increase EMACn_TSSUBSEC with an 8-bit value kept in register EMACn_TSINC.

EMAC n Time Stamp Update Second Register (EMACn_UPDSEC)

Register	Offset	R/W	Description	Reset Value
EMACn_UPDSEC **n=0,1**	EMACn_BA+0x120	R/W	EMAC n Time Stamp Update Second Register	0x0000_0000

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

Bits	Description	
[31:0]	**SEC**	**Time Stamp Counter Second Update** When TSIEN (EMACn_TSCTL[1]) is high. EMAC loads this 32-bit value to EMACn_TSSEC directly. When TSUPDATE (EMACn_TSCTL[3]) is high, EMAC increases EMACn_TSSEC with this 32-bit value.

EMAC n Time Stamp Update Sub Second Register (EMACn_UPDSSEC)

Register	Offset	R/W	Description	Reset Value
EMACn_UPDSSEC **n=0,1**	EMACn_BA+0x124	R/W	EMAC n Time Stamp Update Sub Second Register	0x0000_0000

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

Bits	Description	
[31:0]	**SUBSEC**	**Time Stamp Counter Sub-second Update** When TSIEN (EMACn_TSCTL[1]) is high. EMAC loads this 32-bit value to EMACn_TSSUBSEC directly. When TSUPDATE (EMACn_TSCTL[3]) is high, EMAC increases EMACn_TSSUBSEC with this 32-bit value.

EMAC n Time Stamp Alarm Second Register (EMACn_ALMSEC)

Register	Offset	R/W	Description	Reset Value
EMACn_ALMSEC **n=0,1**	EMACn_BA+0x128	R/W	EMAC n Time Stamp Alarm Second Register	0x0000_0000

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

Bits	Description	
[31:0]	**SEC**	**Time Stamp Counter Second Alarm** Time stamp counter second part alarm value. This value is only useful when TSALMEN (EMACn_TSCTL[5]) high. If TSALMEN (EMACn_TSCTL[5]) is high, EMACn_TSSEC equals to EMACn_ALMSEC and EMACn_TSSUBSEC equals to EMACn_ALMSSEC, Ethernet MAC controller set TSALS (EMACn_MISTA[28]) high.

EMAC n Time Stamp Alarm Sub Second Register (EMACn_ALMSSEC)

Register	Offset	R/W	Description	Reset Value
EMACn_ALMSSEC **n=0,1**	EMACn_BA+0x12C	R/W	EMAC n Time Stamp Alarm Sub Second Register	0x0000_0000

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

Bits	Description	
[31:0]	**SUBSEC**	**Time Stamp Counter Sub-second Alarm** Time stamp counter sub-second part alarm value. This value is only useful when TSALMEN (EMACn_TSCTL[5]) high. If TSALMEN (EMACn_TSCTL[5]) is high, EMACn_TSSEC equals to EMACn_ALMSEC and EMACn_TSSUBSEC equals to EMACn_ALMSSEC, Ethernet MAC controller set TSALS (EMACn_MISTA[28]) high.

6.21 High Speed USB 2.0 Device Controller (HSUSBD)

6.21.1 Overview

The USB device controller interfaces the AHB bus and the UTMI bus. The USB controller contains both the AHB master interface and AHB slave interface. CPU programs the USB controller registers through the AHB slave interface. For IN or OUT transfer, the USB device controller needs to write data to memory or read data from memory through the AHB master interface. The USB device controller is complaint with USB 2.0 specification and it contains 12 configurable endpoints in addition to control endpoint. These endpoints could be configured to BULK, INTERRUPT or ISOCHRONOUS. The USB device controller has a built-in DMA to relieve the load of CPU.

6.21.2 Features

- USB Specification reversion 2.0 compliant
- Supports 12 configurable endpoints in addition to Control Endpoint
- Each of the endpoints can be Isochronous, Bulk or Interrupt and either IN or OUT direction
- Three different operation modes of an in-endpoint — Auto Validation mode, Manual Validation mode, Fly mode
- Supports DMA operation
- 4096 Bytes Configurable RAM used as endpoint buffer
- Supports Endpoint Maximum Packet Size up to 1024 bytes

6.21.3 Block Diagram

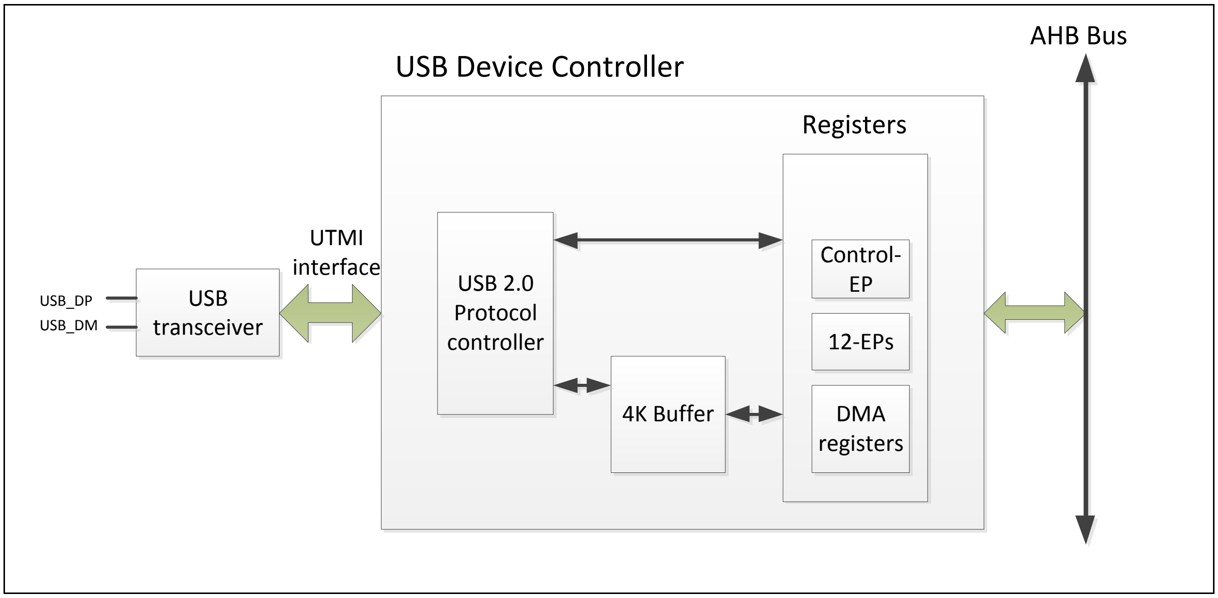


Figure 6.21-1 USB Device Controller Block Diagram

6.21.4 Basic Configuration

- Clock Source Configuration
  - Enable HSUSBD clock in USBD (CLK_HCLKEN[19])
- Reset Configuration
  - Reset HSUSBD controller in USBDRST (SYS_AHBIPRST[19])
- Pin Configuration

Group	Pin Name	GPIO	MFP
USB0	USB0_VBUSVLD	PE.11	MFP1

6.21.5 Functional Description

6.21.5.1 Operation of different In-transfer modes

The data for any in-transfer is written into the internal buffer when in turn is sent to the host on receipt of an in-token. There are three different modes by which the data sent to the host is validated by CPU.

- Auto-Validation Mode
- Manual-Validation Mode
- Fly Mode

6.21.5.2 Auto-Validation Mode

If an endpoint is selected to be operating in auto-validation mode, the endpoint responds only with data payload to be equal to EPMPS register. The endpoint controller wait until the amount of data is equal to EPMPS value and then validates the data. If CPU needs to send a short-packet at the end of a transfer, the SHORTTXEN bit of HSUSBD_EPxRSPCTL[6] should be set. When this bit set, any remaining data in the buffer is validated and is sent to the host, for the forthcoming in-token.

This mode requires least intervention of CPU, as most of the work is done by the USB device controller. The mode can be selected, when the data payload sent to host is always equal to MPS size.

SHORTTXEN	Data Availability In Buffer	Data Sent/NAK Sent
0	< Max. Packet Size	NAK sent
0	>= Max. Packet Size	Data payload of max. packet size
1	< Max. Packet Size	Available data of < max. packet size
1	>= Max. Packet Size	Data payload of max. packet size sent

6.21.5.3 Manual-Validation Mode

If the endpoint is selected to be operating in manual-validation mode, the endpoint responds only when the data in the buffer is validated by CPU every time. The CPU has to write data into the buffer and then write the count of the data into EPxTXCNT register. Once the data is validating by writing a count into the EPxTXCNT register, the data is sent to the host on receipt of an in-token.

This mode requires intervention of CPU for each transfer. But this would be useful, if the data-count to be sent each time is not fixed, and it is being decided by CPU.

EPxTXCNT Written	Data Availability In Buffer	Data Sent/NAK Sent
NO	-	NAK
YES	EPxTXCNT	Data payload of EPxTXCNT sent

6.21.5.4 Fly Mode

The fly mode of operation is simplest mode of operation, where there is no validation procedure. The buffer is being filled by CPU. If an in-token is send from the host, the data in the buffer is automatically validated and sent to the host. If the data in the buffer spans more than one packet of maximum packet size, the controller automatically packs to equal to MPS and send it to the host.

This mode requires the least intervention by CPU. This mode is best suited for isochronous data transfer, where the speed of data transfer is more important than the packet size.

Data Availability In Buffer	Data Sent
< Max. Packet Size	Data available sent
>= Max. Packet Size	Data payload of max. packet size sent

6.21.5.5 Scatter-Gather function

When enabling the scatter gather DMA function, setting SGEN high and HSUSBD_DMACNT set 8 bytes, DMA will be enabled to fetch the descriptor which describes the real memory address and length. The descriptor will be an 8-byte format, like the following:

	Format			
	[31]	[30]	[29:0]	
Word0	MEM_ADDR[31:0]			
Word1	EOT	RD	Reserved	Count[19:0]

MEM_ADDR: It specifies the memory address (AHB address).

EOT: end of transfer. When this bit is set to high, it means this is the last descriptor.

RD: “1” means read from memory into buffer. “0” means read from buffer into memory.

6.21.6 Registers Map

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

Register	Offset	R/W	Description	Reset Value
HSUSBD Base Address: **HSUSBD_BA = 0xB001_6000**				
HSUSBD_GINTSTS	HSUSBD_BA+0x000	R	Global Interrupt Status Register	0x0000_0000
HSUSBD_GINTEN	HSUSBD_BA+0x008	R/W	Global Interrupt Enable Register	0x0000_0001
HSUSBD_BUSINTSTS	HSUSBD_BA+0x010	R/W	USB Bus Interrupt Status Register	0x0000_0000
HSUSBD_BUSINTEN	HSUSBD_BA+0x014	R/W	USB Bus Interrupt Enable Register	0x0000_0040
HSUSBD_OPER	HSUSBD_BA+0x018	R/W	USB Operational Register	0x0000_0002
HSUSBD_FRAMECNT	HSUSBD_BA+0x01C	R	USB Frame Count Register	0x0000_0000
HSUSBD_FADDR	HSUSBD_BA+0x020	R/W	USB Function Address Register	0x0000_0000
HSUSBD_TEST	HSUSBD_BA+0x024	R/W	USB Test Mode Register	0x0000_0000
HSUSBD_CEPDAT	HSUSBD_BA+0x028	R/W	Control Endpoint Data Buffer	0x0000_0000
HSUSBD_CEPCTL	HSUSBD_BA+0x02C	R/W	Control Endpoint Control Register	0x0000_0000
HSUSBD_CEPINTEN	HSUSBD_BA+0x030	R/W	Control Endpoint Interrupt Enable	0x0000_0000
HSUSBD_CEPINTSTS	HSUSBD_BA+0x034	R/W	Control Endpoint Interrupt Status	0x0000_1800
HSUSBD_CEPTXCNT	HSUSBD_BA+0x038	R/W	Control Endpoint in Transfer Data Count	0x0000_0000
HSUSBD_CEPRXCNT	HSUSBD_BA+0x03C	R	Control Endpoint Out Transfer Data Count	0x0000_0000
HSUSBD_CEPDATCNT	HSUSBD_BA+0x040	R	Control Endpoint Data Count	0x0000_0000
HSUSBD_SETUP1_0	HSUSBD_BA+0x044	R	Setup1 & Setup0 Bytes	0x0000_0000
HSUSBD_SETUP3_2	HSUSBD_BA+0x048	R	Setup3 & Setup2 Bytes	0x0000_0000
HSUSBD_SETUP5_4	HSUSBD_BA+0x04C	R	Setup5 & Setup4 Bytes	0x0000_0000
HSUSBD_SETUP7_6	HSUSBD_BA+0x050	R	Setup7 & Setup6 Bytes	0x0000_0000
HSUSBD_CEPBUFSTART	HSUSBD_BA+0x054	R/W	Control Endpoint RAM Start Address Register	0x0000_0000
HSUSBD_CEPBUFEND	HSUSBD_BA+0x058	R/W	Control Endpoint RAM End Address Register	0x0000_0000
HSUSBD_DMACTL	HSUSBD_BA+0x05C	R/W	DMA Control Status Register	0x0000_0000
HSUSBD_DMACNT	HSUSBD_BA+0x060	R/W	DMA Count Register	0x0000_0000
HSUSBD_EPADAT	HSUSBD_BA+0x064	R/W	Endpoint A Data Register	0x0000_0000
HSUSBD_EPAINTSTS	HSUSBD_BA+0x068	R/W	Endpoint A Interrupt Status Register	0x0000_0003
HSUSBD_EPAINTEN	HSUSBD_BA+0x06C	R/W	Endpoint A Interrupt Enable Register	0x0000_0000
HSUSBD_EPADATCNT	HSUSBD_BA+0x070	R	Endpoint A Data Available Count	0x0000_0000

			Register	
HSUSBD_EPARSPCTL	HSUSBD_BA+0x074	R/W	Endpoint A Response Control Register	0x0000_0000
HSUSBD_EPAMPS	HSUSBD_BA+0x078	R/W	Endpoint A Maximum Packet Size Register	0x0000_0000
HSUSBD_EPATXCNT	HSUSBD_BA+0x07C	R/W	Endpoint A Transfer Count Register	0x0000_0000
HSUSBD_EPACFG	HSUSBD_BA+0x080	R/W	Endpoint A Configuration Register	0x0000_0012
HSUSBD_EPABUFSTART	HSUSBD_BA+0x084	R/W	Endpoint A RAM Start Address Register	0x0000_0000
HSUSBD_EPABUFEND	HSUSBD_BA+0x088	R/W	Endpoint A RAM End Address Register	0x0000_0000
HSUSBD_EPBDAT	HSUSBD_BA+0x08C	R/W	Endpoint B Data Register	0x0000_0000
HSUSBD_EPBINTSTS	HSUSBD_BA+0x090	R/W	Endpoint B Interrupt Status Register	0x0000_0003
HSUSBD_EPBINTEN	HSUSBD_BA+0x094	R/W	Endpoint B Interrupt Enable Register	0x0000_0000
HSUSBD_EPBDATCNT	HSUSBD_BA+0x098	R	Endpoint B Data Available Count Register	0x0000_0000
HSUSBD_EPBRSPCTL	HSUSBD_BA+0x09C	R/W	Endpoint B Response Control Register	0x0000_0000
HSUSBD_EPBMPS	HSUSBD_BA+0x0A0	R/W	Endpoint B Maximum Packet Size Register	0x0000_0000
HSUSBD_EPBTXCNT	HSUSBD_BA+0x0A4	R/W	Endpoint B Transfer Count Register	0x0000_0000
HSUSBD_EPBCFG	HSUSBD_BA+0x0A8	R/W	Endpoint B Configuration Register	0x0000_0022
HSUSBD_EPBBUFSTART	HSUSBD_BA+0x0AC	R/W	Endpoint B RAM Start Address Register	0x0000_0000
HSUSBD_EPBBUFEND	HSUSBD_BA+0x0B0	R/W	Endpoint B RAM End Address Register	0x0000_0000
HSUSBD_EPCDAT	HSUSBD_BA+0x0B4	R/W	Endpoint C Data Register	0x0000_0000
HSUSBD_EPCINTSTS	HSUSBD_BA+0x0B8	R/W	Endpoint C Interrupt Status Register	0x0000_0003
HSUSBD_EPCINTEN	HSUSBD_BA+0x0BC	R/W	Endpoint C Interrupt Enable Register	0x0000_0000
HSUSBD_EPCDATCNT	HSUSBD_BA+0x0C0	R	Endpoint C Data Available Count Register	0x0000_0000
HSUSBD_EPCRSPCTL	HSUSBD_BA+0x0C4	R/W	Endpoint C Response Control Register	0x0000_0000
HSUSBD_EPCMPS	HSUSBD_BA+0x0C8	R/W	Endpoint C Maximum Packet Size Register	0x0000_0000
HSUSBD_EPCTXCNT	HSUSBD_BA+0x0CC	R/W	Endpoint C Transfer Count Register	0x0000_0000
HSUSBD_EPCCFG	HSUSBD_BA+0x0D0	R/W	Endpoint C Configuration Register	0x0000_0032
HSUSBD_EPCBUFSTART	HSUSBD_BA+0x0D4	R/W	Endpoint C RAM Start Address Register	0x0000_0000
HSUSBD_EPCBUFEND	HSUSBD_BA+0x0D8	R/W	Endpoint C RAM End Address Register	0x0000_0000
HSUSBD_EPDDAT	HSUSBD_BA+0x0DC	R/W	Endpoint D Data Register	0x0000_0000
HSUSBD_EPDINTSTS	HSUSBD_BA+0x0E0	R/W	Endpoint D Interrupt Status Register	0x0000_0003
HSUSBD_EPDINTEN	HSUSBD_BA+0x0E4	R/W	Endpoint D Interrupt Enable Register	0x0000_0000
HSUSBD_EPDDATCNT	HSUSBD_BA+0x0E8	R	Endpoint D Data Available Count Register	0x0000_0000

HSUSBD_EPDRSPCTL	HSUSBD_BA+0x0EC	R/W	Endpoint D Response Control Register	0x0000_0000
HSUSBD_EPDMPS	HSUSBD_BA+0x0F0	R/W	Endpoint D Maximum Packet Size Register	0x0000_0000
HSUSBD_EPDTXCNT	HSUSBD_BA+0x0F4	R/W	Endpoint D Transfer Count Register	0x0000_0000
HSUSBD_EPDCFG	HSUSBD_BA+0x0F8	R/W	Endpoint D Configuration Register	0x0000_0042
HSUSBD_EPDBUFSTART	HSUSBD_BA+0x0FC	R/W	Endpoint D RAM Start Address Register	0x0000_0000
HSUSBD_EPDBUFEND	HSUSBD_BA+0x100	R/W	Endpoint D RAM End Address Register	0x0000_0000
HSUSBD_EPEDAT	HSUSBD_BA+0x104	R/W	Endpoint E Data Register	0x0000_0000
HSUSBD_EPEINTSTS	HSUSBD_BA+0x108	R/W	Endpoint E Interrupt Status Register	0x0000_0003
HSUSBD_EPEINTEN	HSUSBD_BA+0x10C	R/W	Endpoint E Interrupt Enable Register	0x0000_0000
HSUSBD_EPEDATCNT	HSUSBD_BA+0x110	R	Endpoint E Data Available Count Register	0x0000_0000
HSUSBD_EPERSPCTL	HSUSBD_BA+0x114	R/W	Endpoint E Response Control Register	0x0000_0000
HSUSBD_EPEMPS	HSUSBD_BA+0x118	R/W	Endpoint E Maximum Packet Size Register	0x0000_0000
HSUSBD_EPETXCNT	HSUSBD_BA+0x11C	R/W	Endpoint E Transfer Count Register	0x0000_0000
HSUSBD_EPECFG	HSUSBD_BA+0x120	R/W	Endpoint E Configuration Register	0x0000_0052
HSUSBD_EPEBUFSTART	HSUSBD_BA+0x124	R/W	Endpoint E RAM Start Address Register	0x0000_0000
HSUSBD_EPEBUFEND	HSUSBD_BA+0x128	R/W	Endpoint E RAM End Address Register	0x0000_0000
HSUSBD_EPFDAT	HSUSBD_BA+0x12C	R/W	Endpoint F Data Register	0x0000_0000
HSUSBD_EPFINTSTS	HSUSBD_BA+0x130	R/W	Endpoint F Interrupt Status Register	0x0000_0003
HSUSBD_EPFINTEN	HSUSBD_BA+0x134	R/W	Endpoint F Interrupt Enable Register	0x0000_0000
HSUSBD_EPFDATCNT	HSUSBD_BA+0x138	R	Endpoint F Data Available Count Register	0x0000_0000
HSUSBD_EPFRSPCTL	HSUSBD_BA+0x13C	R/W	Endpoint F Response Control Register	0x0000_0000
HSUSBD_EPFMPS	HSUSBD_BA+0x140	R/W	Endpoint F Maximum Packet Size Register	0x0000_0000
HSUSBD_EPFTXCNT	HSUSBD_BA+0x144	R/W	Endpoint F Transfer Count Register	0x0000_0000
HSUSBD_EPFCFG	HSUSBD_BA+0x148	R/W	Endpoint F Configuration Register	0x0000_0062
HSUSBD_EPFBUFSTART	HSUSBD_BA+0x14C	R/W	Endpoint F RAM Start Address Register	0x0000_0000
HSUSBD_EPFBUFEND	HSUSBD_BA+0x150	R/W	Endpoint F RAM End Address Register	0x0000_0000
HSUSBD_EPGDAT	HSUSBD_BA+0x154	R/W	Endpoint G Data Register	0x0000_0000
HSUSBD_EPGINTSTS	HSUSBD_BA+0x158	R/W	Endpoint G Interrupt Status Register	0x0000_0003
HSUSBD_EPGINTEN	HSUSBD_BA+0x15C	R/W	Endpoint G Interrupt Enable Register	0x0000_0000
HSUSBD_EPGDATCNT	HSUSBD_BA+0x160	R	Endpoint G Data Available Count Register	0x0000_0000
HSUSBD_EPGRSPCTL	HSUSBD_BA+0x164	R/W	Endpoint G Response Control Register	0x0000_0000

HSUSBD_EPGMPS	HSUSBD_BA+0x168	R/W	Endpoint G Maximum Packet Size Register	0x0000_0000
HSUSBD_EPGTXCNT	HSUSBD_BA+0x16C	R/W	Endpoint G Transfer Count Register	0x0000_0000
HSUSBD_EPGCFG	HSUSBD_BA+0x170	R/W	Endpoint G Configuration Register	0x0000_0072
HSUSBD_EPGBUFSTART	HSUSBD_BA+0x174	R/W	Endpoint G RAM Start Address Register	0x0000_0000
HSUSBD_EPGBUFEND	HSUSBD_BA+0x178	R/W	Endpoint G RAM End Address Register	0x0000_0000
HSUSBD_EPHDAT	HSUSBD_BA+0x17C	R/W	Endpoint H Data Register	0x0000_0000
HSUSBD_EPHINTSTS	HSUSBD_BA+0x180	R/W	Endpoint H Interrupt Status Register	0x0000_0003
HSUSBD_EPHINTEN	HSUSBD_BA+0x184	R/W	Endpoint H Interrupt Enable Register	0x0000_0000
HSUSBD_EPHDATCNT	HSUSBD_BA+0x188	R	Endpoint H Data Available Count Register	0x0000_0000
HSUSBD_EPHRSPCTL	HSUSBD_BA+0x18C	R/W	Endpoint H Response Control Register	0x0000_0000
HSUSBD_EPHMPS	HSUSBD_BA+0x190	R/W	Endpoint H Maximum Packet Size Register	0x0000_0000
HSUSBD_EPHTXCNT	HSUSBD_BA+0x194	R/W	Endpoint H Transfer Count Register	0x0000_0000
HSUSBD_EPHCFG	HSUSBD_BA+0x198	R/W	Endpoint H Configuration Register	0x0000_0082
HSUSBD_EPHBUFSTART	HSUSBD_BA+0x19C	R/W	Endpoint H RAM Start Address Register	0x0000_0000
HSUSBD_EPHBUFEND	HSUSBD_BA+0x1A0	R/W	Endpoint H RAM End Address Register	0x0000_0000
HSUSBD_EPIDAT	HSUSBD_BA+0x1A4	R/W	Endpoint I Data Register	0x0000_0000
HSUSBD_EPIINTSTS	HSUSBD_BA+0x1A8	R/W	Endpoint I Interrupt Status Register	0x0000_0003
HSUSBD_EPIINTEN	HSUSBD_BA+0x1AC	R/W	Endpoint I Interrupt Enable Register	0x0000_0000
HSUSBD_EPIDATCNT	HSUSBD_BA+0x1B0	R	Endpoint I Data Available Count Register	0x0000_0000
HSUSBD_EPIRSPCTL	HSUSBD_BA+0x1B4	R/W	Endpoint I Response Control Register	0x0000_0000
HSUSBD_EPIMPS	HSUSBD_BA+0x1B8	R/W	Endpoint I Maximum Packet Size Register	0x0000_0000
HSUSBD_EPITXCNT	HSUSBD_BA+0x1BC	R/W	Endpoint I Transfer Count Register	0x0000_0000
HSUSBD_EPICFG	HSUSBD_BA+0x1C0	R/W	Endpoint I Configuration Register	0x0000_0092
HSUSBD_EPIBUFSTART	HSUSBD_BA+0x1C4	R/W	Endpoint I RAM Start Address Register	0x0000_0000
HSUSBD_EPIBUFEND	HSUSBD_BA+0x1C8	R/W	Endpoint I RAM End Address Register	0x0000_0000
HSUSBD_EPJDAT	HSUSBD_BA+0x1CC	R/W	Endpoint J Data Register	0x0000_0000
HSUSBD_EPJINTSTS	HSUSBD_BA+0x1D0	R/W	Endpoint J Interrupt Status Register	0x0000_0003
HSUSBD_EPJINTEN	HSUSBD_BA+0x1D4	R/W	Endpoint J Interrupt Enable Register	0x0000_0000
HSUSBD_EPJDATCNT	HSUSBD_BA+0x1D8	R	Endpoint J Data Available Count Register	0x0000_0000
HSUSBD_EPJRSPCTL	HSUSBD_BA+0x1DC	R/W	Endpoint J Response Control Register	0x0000_0000
HSUSBD_EPJMPS	HSUSBD_BA+0x1E0	R/W	Endpoint J Maximum Packet Size Register	0x0000_0000

HSUSBD_EPJTXCNT	HSUSBD_BA+0x1E4	R/W	Endpoint J Transfer Count Register	0x0000_0000
HSUSBD_EPJCFG	HSUSBD_BA+0x1E8	R/W	Endpoint J Configuration Register	0x0000_00A2
HSUSBD_EPJBUFSTART	HSUSBD_BA+0x1EC	R/W	Endpoint J RAM Start Address Register	0x0000_0000
HSUSBD_EPJBUFEND	HSUSBD_BA+0x1F0	R/W	Endpoint J RAM End Address Register	0x0000_0000
HSUSBD_EPKDAT	HSUSBD_BA+0x1F4	R/W	Endpoint K Data Register	0x0000_0000
HSUSBD_EPKINTSTS	HSUSBD_BA+0x1F8	R/W	Endpoint K Interrupt Status Register	0x0000_0003
HSUSBD_EPKINTEN	HSUSBD_BA+0x1FC	R/W	Endpoint K Interrupt Enable Register	0x0000_0000
HSUSBD_EPKDATCNT	HSUSBD_BA+0x200	R	Endpoint K Data Available Count Register	0x0000_0000
HSUSBD_EPKRSPCTL	HSUSBD_BA+0x204	R/W	Endpoint K Response Control Register	0x0000_0000
HSUSBD_EPKMPS	HSUSBD_BA+0x208	R/W	Endpoint K Maximum Packet Size Register	0x0000_0000
HSUSBD_EPKTXCNT	HSUSBD_BA+0x20C	R/W	Endpoint K Transfer Count Register	0x0000_0000
HSUSBD_EPKCFG	HSUSBD_BA+0x210	R/W	Endpoint K Configuration Register	0x0000_00B2
HSUSBD_EPKBUFSTART	HSUSBD_BA+0x214	R/W	Endpoint K RAM Start Address Register	0x0000_0000
HSUSBD_EPKBUFEND	HSUSBD_BA+0x218	R/W	Endpoint K RAM End Address Register	0x0000_0000
HSUSBD_EPLDAT	HSUSBD_BA+0x21C	R/W	Endpoint L Data Register	0x0000_0000
HSUSBD_EPLINTSTS	HSUSBD_BA+0x220	R/W	Endpoint L Interrupt Status Register	0x0000_0003
HSUSBD_EPLINTEN	HSUSBD_BA+0x224	R/W	Endpoint L Interrupt Enable Register	0x0000_0000
HSUSBD_EPLDATCNT	HSUSBD_BA+0x228	R	Endpoint L Data Available Count Register	0x0000_0000
HSUSBD_EPLRSPCTL	HSUSBD_BA+0x22C	R/W	Endpoint L Response Control Register	0x0000_0000
HSUSBD_EPLMPS	HSUSBD_BA+0x230	R/W	Endpoint L Maximum Packet Size Register	0x0000_0000
HSUSBD_EPLTXCNT	HSUSBD_BA+0x234	R/W	Endpoint L Transfer Count Register	0x0000_0000
HSUSBD_EPLCFG	HSUSBD_BA+0x238	R/W	Endpoint L Configuration Register	0x0000_00C2
HSUSBD_EPLBUFSTART	HSUSBD_BA+0x23C	R/W	Endpoint L RAM Start Address Register	0x0000_0000
HSUSBD_EPLBUFEND	HSUSBD_BA+0x240	R/W	Endpoint L RAM End Address Register	0x0000_0000
HSUSBD_DMAADDR	HSUSBD_BA+0x700	R/W	AHB DMA Address Register	0x0000_0000
HSUSBD_PHYCTL	HSUSBD_BA+0x704	R/W	USB PHY Control Register	0x0000_0420

6.21.7 Register Description

Global Interrupt Status Register (HSUSBD_GINTSTS)

Register	Offset	R/W	Description	Reset Value
HSUSBD_GINTSTS	HSUSBD_BA+0x000	R	Global 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		**EPLIF**	**EPKIF**	**EPJIF**	**EPIIF**	**EPHIF**	**EPGIF**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EPFIF	**EPEIF**	**EPDIF**	**EPCIF**	**EPBIF**	**EPAIF**	**CEPIF**	**USBIF**

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13]	**EPLIF**	**Endpoints L Interrupt** When set, the corresponding Endpoint L's interrupt status register should be read to determine the cause of the interrupt. 0 = No interrupt event occurred. 1 = The related interrupt event occurred.
[12]	**EPKIF**	**Endpoints K Interrupt** When set, the corresponding Endpoint K's interrupt status register should be read to determine the cause of the interrupt. 0 = No interrupt event occurred. 1 = The related interrupt event occurred.
[11]	**EPJIF**	**Endpoints J Interrupt** When set, the corresponding Endpoint J's interrupt status register should be read to determine the cause of the interrupt. 0 = No interrupt event occurred. 1 = The related interrupt event occurred.
[10]	**EPIIF**	**Endpoints I Interrupt** When set, the corresponding Endpoint I's interrupt status register should be read to determine the cause of the interrupt. 0 = No interrupt event occurred. 1 = The related interrupt event occurred.

[9]	**EPHIF**	**Endpoints H Interrupt** When set, the corresponding Endpoint H’s interrupt status register should be read to determine the cause of the interrupt. 0 = No interrupt event occurred. 1 = The related interrupt event occurred.
[8]	**EPGIF**	**Endpoints G Interrupt** When set, the corresponding Endpoint G's interrupt status register should be read to determine the cause of the interrupt. 0 = No interrupt event occurred. 1 = The related interrupt event occurred.
[7]	**EPFIF**	**Endpoints F Interrupt** When set, the corresponding Endpoint F's interrupt status register should be read to determine the cause of the interrupt. 0 = No interrupt event occurred. 1 = The related interrupt event occurred.
[6]	**EPEIF**	**Endpoints E Interrupt** When set, the corresponding Endpoint E's interrupt status register should be read to determine the cause of the interrupt. 0 = No interrupt event occurred. 1 = The related interrupt event occurred.
[5]	**EPDIF**	**Endpoints D Interrupt** When set, the corresponding Endpoint D's interrupt status register should be read to determine the cause of the interrupt. 0 = No interrupt event occurred. 1 = The related interrupt event occurred.
[4]	**EPCIF**	**Endpoints C Interrupt** When set, the corresponding Endpoint C's interrupt status register should be read to determine the cause of the interrupt. 0 = No interrupt event occurred. 1 = The related interrupt event occurred.
[3]	**EPBIF**	**Endpoints B Interrupt** When set, the corresponding Endpoint B's interrupt status register should be read to determine the cause of the interrupt. 0 = No interrupt event occurred. 1 = The related interrupt event occurred.
[2]	**EPAIF**	**Endpoints a Interrupt** When set, the corresponding Endpoint A's interrupt status register should be read to determine the cause of the interrupt. 0 = No interrupt event occurred. 1 = The related interrupt event occurred.
[1]	**CEPIF**	**Control Endpoint Interrupt** This bit conveys the interrupt status for control endpoint. When set, Control-ep’s interrupt status register should be read to determine the cause of the interrupt. 0 = No interrupt event occurred. 1 = The related interrupt event occurred.

[0]	**USBIF**	**USB Interrupt** This bit conveys the interrupt status for USB specific events endpoint. When set, USB interrupt status register should be read to determine the cause of the interrupt. 0 = No interrupt event occurred. 1 = The related interrupt event occurred.

Global Interrupt Enable Register (HSUSBD_GINTEN)

Register	Offset	R/W	Description	Reset Value
HSUSBD_GINTEN	HSUSBD_BA+0x008	R/W	Global Interrupt Enable 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		EPLIEN	EPKIEN	EPJIEN	EPIIEN	EPHIEN	EPGIEN
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EPFIEN	EPEIEN	EPDIEN	EPCIEN	EPBIEN	EPAIEN	CEPIEN	USBIEN

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13]	**EPLIEN**	**Interrupt Enable Control for Endpoint L** When set, this bit enables a local interrupt to be generated when an interrupt is pending for the endpoint L 0 = The related interrupt Disabled. 1 = The related interrupt Enabled.
[12]	**EPKIEN**	**Interrupt Enable Control for Endpoint K** When set, this bit enables a local interrupt to be generated when an interrupt is pending for the endpoint K 0 = The related interrupt Disabled. 1 = The related interrupt Enabled.
[11]	**EPJIEN**	**Interrupt Enable Control for Endpoint J** When set, this bit enables a local interrupt to be generated when an interrupt is pending for the endpoint J 0 = The related interrupt Disabled. 1 = The related interrupt Enabled.
[10]	**EPIIEN**	**Interrupt Enable Control for Endpoint I** When set, this bit enables a local interrupt to be generated when an interrupt is pending for the endpoint I 0 = The related interrupt Disabled. 1 = The related interrupt Enabled.
[9]	**EPHIEN**	**Interrupt Enable Control for Endpoint H** When set, this bit enables a local interrupt to be generated when an interrupt is pending for the endpoint H 0 = The related interrupt Disabled. 1 = The related interrupt Enabled.

[8]	**EPGIEN**	**Interrupt Enable Control for Endpoint G** When set, this bit enables a local interrupt to be generated when an interrupt is pending for the endpoint G 0 = The related interrupt Disabled. 1 = The related interrupt Enabled.
[7]	**EPFIEN**	**Interrupt Enable Control for Endpoint F** When set, this bit enables a local interrupt to be generated when an interrupt is pending for the endpoint F 0 = The related interrupt Disabled. 1 = The related interrupt Enabled.
[6]	**EPEIEN**	**Interrupt Enable Control for Endpoint E** When set, this bit enables a local interrupt to be generated when an interrupt is pending for the endpoint E 0 = The related interrupt Disabled. 1 = The related interrupt Enabled.
[5]	**EPDIEN**	**Interrupt Enable Control for Endpoint D** When set, this bit enables a local interrupt to be generated when an interrupt is pending for the endpoint D 0 = The related interrupt Disabled. 1 = The related interrupt Enabled.
[4]	**EPCIEN**	**Interrupt Enable Control for Endpoint C** When set, this bit enables a local interrupt to be generated when an interrupt is pending for the endpoint C 0 = The related interrupt Disabled. 1 = The related interrupt Enabled.
[3]	**EPBIEN**	**Interrupt Enable Control for Endpoint B** When set, this bit enables a local interrupt to be generated when an interrupt is pending for the endpoint B 0 = The related interrupt Disabled. 1 = The related interrupt Enabled.
[2]	**EPAIEN**	**Interrupt Enable Control for Endpoint a** When set, this bit enables a local interrupt to be generated when an interrupt is pending for the endpoint A. 0 = The related interrupt Disabled. 1 = The related interrupt Enabled.
[1]	**CEPIEN**	**Control Endpoint Interrupt Enable Control** When set, this bit enables a local interrupt to be generated when an interrupt is pending for the control endpoint. 0 = The related interrupt Disabled. 1 = The related interrupt Enabled.
[0]	**USBIEN**	**USB Interrupt Enable Control** When set, this bit enables a local interrupt to be generated when a USB event occurs on the bus. 0 = The related interrupt Disabled. 1 = The related interrupt Enabled.

USB Bus Interrupt Status Register (HSUSBD_BUSINTSTS)

Register	Offset	R/W	Description	Reset Value
HSUSBD_BUSINTSTS	HSUSBD_BA+0x010	R/W	USB Bus 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							**VBUSDETIF**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**PHYCLKVLDIF**	**DMADONEIF**	**HISPDIF**	**SUSPENDIF**	**RESUMEIF**	**RSTIF**	**SOFIF**

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**VBUSDETIF**	**VBUS Detection Interrupt Status** 0 = No VBUS is plug-in. 1 = VBUS is plug-in. **Note:** Write 1 to clear this bit to 0.
[7]	**Reserved**	Reserved.
[6]	**PHYCLKVLDIF**	**Usable Clock Interrupt** 0 = Usable clock is not available. 1 = Usable clock is available from the transceiver. **Note:** Write 1 to clear this bit to 0.
[5]	**DMADONEIF**	**DMA Completion Interrupt** 0 = No DMA transfer over. 1 = DMA transfer is over. **Note:** Write 1 to clear this bit to 0.
[4]	**HISPDIF**	**High-speed Settle** 0 = No valid high-speed reset protocol is detected. 1 = Valid high-speed reset protocol is over and the device has settled in high-speed. **Note:** Write 1 to clear this bit to 0.
[3]	**SUSPENDIF**	**Suspend Request** This bit is set as default and it has to be cleared by writing ‘1’ before the USB reset. This bit is also set when a USB Suspend request is detected from the host. 0 = No USB Suspend request is detected from the host. 1= USB Suspend request is detected from the host. **Note:** Write 1 to clear this bit to 0.

[2]	**RESUMEIF**	**Resume** When set, this bit indicates that a device resume has occurred. 0 = No device resume has occurred. 1 = Device resume has occurred. **Note:** Write 1 to clear this bit to 0.
[1]	**RSTIF**	**Reset Status** When set, this bit indicates that either the USB root port reset is end. 0 = No USB root port reset is end. 1 = USB root port reset is end. **Note:** Write 1 to clear this bit to 0.
[0]	**SOFIF**	**SOF Receive Control** This bit indicates when a start-of-frame packet has been received. 0 = No start-of-frame packet has been received. 1 = Start-of-frame packet has been received. **Note:** Write 1 to clear this bit to 0.

USB Bus Interrupt Enable Register (HSUSBD_BUSINTEN)

Register	Offset	R/W	Description	Reset Value
HSUSBD_BUSINTEN	HSUSBD_BA+0x014	R/W	USB Bus Interrupt Enable 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							**VBUSDETIEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**PHYCLKVLDIEN**	**DMADONEIEN**	**HISPDIEN**	**SUSPENDIEN**	**RESUMEIEN**	**RSTIEN**	**SOFIEN**

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**VBUSDETIEN**	**VBUS Detection Interrupt Enable Control** This bit enables the VBUS floating detection interrupt. 0 = VBUS floating detection interrupt Disabled. 1 = VBUS floating detection interrupt Enabled.
[7]	**Reserved**	Reserved.
[6]	**PHYCLKVLDIEN**	**Usable Clock Interrupt** This bit enables the usable clock interrupt. 0 = Usable clock interrupt Disabled. 1 = Usable clock interrupt Enabled.
[5]	**DMADONEIEN**	**DMA Completion Interrupt** This bit enables the DMA completion interrupt 0 = DMA completion interrupt Disabled. 1 = DMA completion interrupt Enabled.
[4]	**HISPDIEN**	**High-speed Settle** This bit enables the high-speed settle interrupt. 0 = High-speed settle interrupt Disabled. 1 = High-speed settle interrupt Enabled.
[3]	**SUSPENDIEN**	**Suspend Request** This bit enables the Suspend interrupt. 0 = Suspend interrupt Disabled. 1 = Suspend interrupt Enabled.

[2]	**RESUMEIEN**	**Resume** This bit enables the Resume interrupt. 0 = Resume interrupt Disabled. 1 = Resume interrupt Enabled.
[1]	**RSTIEN**	**Reset Status** This bit enables the USB-Reset interrupt. 0 = USB-Reset interrupt Disabled. 1 = USB-Reset interrupt Enabled.
[0]	**SOFIEN**	**SOF Interrupt** This bit enables the SOF interrupt. 0 = SOF interrupt Disabled. 1 = SOF interrupt Enabled.

USB Operational Register (HSUSBD_OPER)

Register	Offset	R/W	Description	Reset Value
HSUSBD_OPER	HSUSBD_BA+0x018	R/W	USB Operational Register	0x0000_0002

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**CURSPD**	**USB Current Speed** 0 = The device has settled in Full Speed. 1 = The USB device controller has settled in High-speed.
[1]	**HISPDEN**	**USB High-speed** 0 = The USB device controller to suppress the chirp-sequence during reset protocol, thereby allowing the USB device controller to settle in full-speed, even though it is connected to a USB2.0 Host. 1 = The USB device controller to initiate a chirp-sequence during reset protocol.
[0]	**RESUMEEN**	**Generate Resume** 0 = No Resume sequence to be initiated to the host. 1 = A Resume sequence to be initiated to the host if device remote wakeup is enabled. This bit is self-clearing.

USB Frame Count Register (HSUSBD_FRAMECNT)

Register	Offset	R/W	Description	Reset Value
HSUSBD_FRAMECNT	HSUSBD_BA+0x01C	R	USB Frame Count Register	0x0000_0000

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

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13:3]	**FRAMECNT**	**Frame Counter** This field contains the frame count from the most recent start-of-frame packet.
[2:0]	**MFRAMECNT**	**Micro-frame Counter** This field contains the micro-frame number for the frame number in the frame counter field.

USB Function Address Register (HSUSBD_FADDR)

Register	Offset	R/W	Description	Reset Value
HSUSBD_FADDR	HSUSBD_BA+0x020	R/W	USB 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 Function Address** This field contains the current USB address of the device. This field is cleared when a root port reset is detected.

USB Test Mode Register (HSUSBD_TEST)

Register	Offset	R/W	Description	Reset Value
HSUSBD_TEST	HSUSBD_BA+0x024	R/W	USB Test Mode Register	0x0000_0000

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

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**TESTMODE**	**Test Mode Selection** 000 = Normal Operation. 001 = Test_J. 010 = Test_K. 011 = Test_SE0_NAK. 100 = Test_Packet. 101 = Test_Force_Enable. 110 = Reserved. 111 = Reserved. **Note:** This field is cleared when root port reset is detected.

Control Endpoint Data Buffer (HSUSBD_CEPDAT)

Register	Offset	R/W	Description	Reset Value
HSUSBD_CEPDAT	HSUSBD_BA+0x028	R/W	Control Endpoint Data Buffer	0x0000_0000

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

Bits	Description	
[31:0]	**DAT**	**Control-endpoint Data Buffer** Control endpoint data buffer for the buffer transaction (read or write). **Note:** Only word or byte access are supported.

Control Endpoint Control Register (HSUSBD_CEPCTL)

Register	Offset	R/W	Description	Reset Value
HSUSBD_CEPCTL	HSUSBD_BA+0x02C	R/W	Control Endpoint 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				FLUSH	ZEROLEN	STALLEN	NAKCLR

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**FLUSH**	**CEP-fLUSH Bit** 0 = No the packet buffer and its corresponding HSUSBD_CEPDATCNT register to be cleared. 1 = The packet buffer and its corresponding HSUSBD_CEPDATCNT register to be cleared. This bit is self-cleaning.
[2]	**ZEROLEN**	**Zero Packet Length** This bit is valid for Auto Validation mode only. 0 = No zero length packet to the host during Data stage to an IN token. 1 = USB device controller can send a zero length packet to the host during Data stage to an IN token. This bit gets cleared once the zero length data packet is sent. So, the local CPU need not write again to clear this bit.
[1]	**STALLEN**	**Stall Enable Control** When this stall bit is set, the control endpoint sends a stall handshake in response to any in or out token thereafter. This is typically used for response to invalid/unsupported requests. When this bit is being set the NAK clear bit has to be cleared at the same time since the NAK clear bit has highest priority than STALL. It is automatically cleared on receipt of a next setup-token. So, the local CPU need not write again to clear this bit. 0 = No sends a stall handshake in response to any in or out token thereafter. 1 = The control endpoint sends a stall handshake in response to any in or out token thereafter. **Note:** Only when CPU writes data[1:0] is 2’b10 or 2’b00, this bit can be updated.

[0]	**NAKCLR**	**No Acknowledge Control** This bit plays a crucial role in any control transfer. 0 = The bit is being cleared by the local CPU by writing zero, the USB device controller will be responding with NAKs for the subsequent status phase. This mechanism holds the host from moving to the next request, until the local CPU is also ready to process the next request. 1 = This bit is set to one by the USB device controller, whenever a setup token is received. The local CPU can take its own time to finish off any house-keeping work based on the request and then clear this bit. **Note:** Only when CPU writes data[1:0] is 2’b10 or 2’b00, this bit can be updated.

Control Endpoint Interrupt Enable (HSUSBD_CEPINTEN)

Register	Offset	R/W	Description	Reset Value
HSUSBD_CEPINTEN	HSUSBD_BA+0x030	R/W	Control Endpoint Interrupt Enable	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			BUFEMPTYIEN	BUFFULLIEN	STSDONEIEN	ERRIEN	STALLIEN
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NAKIEN	RXPKIEN	TXPKIEN	PINGIEN	INTKIEN	OUTTKIEN	SETUPPKIEN	SETUPTKIEN

Bits	Description	
[31:13]	**Reserved**	Reserved.
[12]	**BUFEMPTYIEN**	**Buffer Empty Interrupt** 0 = The buffer empty interrupt in Control Endpoint Disabled. 1= The buffer empty interrupt in Control Endpoint Enabled.
[11]	**BUFFULLIEN**	**Buffer Full Interrupt** 0 = The buffer full interrupt in Control Endpoint Disabled. 1 = The buffer full interrupt in Control Endpoint Enabled.
[10]	**STSDONEIEN**	**Status Completion Interrupt** 0 = The Status Completion interrupt in Control Endpoint Disabled. 1 = The Status Completion interrupt in Control Endpoint Enabled.
[9]	**ERRIEN**	**USB Error Interrupt** 0 = The USB Error interrupt in Control Endpoint Disabled. 1 = The USB Error interrupt in Control Endpoint Enabled.
[8]	**STALLIEN**	**STALL Sent Interrupt** 0 = The STALL sent interrupt in Control Endpoint Disabled. 1 = The STALL sent interrupt in Control Endpoint Enabled.
[7]	**NAKIEN**	**NAK Sent Interrupt** 0 = The NAK sent interrupt in Control Endpoint Disabled. 1 = The NAK sent interrupt in Control Endpoint Enabled.
[6]	**RXPKIEN**	**Data Packet Received Interrupt** 0 = The data received interrupt in Control Endpoint Disabled. 1 = The data received interrupt in Control Endpoint Enabled.

[5]	**TXPKIEN**	**Data Packet Transmitted Interrupt** 0 = The data packet transmitted interrupt in Control Endpoint Disabled. 1 = The data packet transmitted interrupt in Control Endpoint Enabled.
[4]	**PINGIEN**	**Ping Token Interrupt** 0 = The ping token interrupt in Control Endpoint Disabled. 1 = The ping token interrupt Control Endpoint Enabled.
[3]	**INTKIEN**	**In Token Interrupt** 0 = The IN token interrupt in Control Endpoint Disabled. 1 = The IN token interrupt in Control Endpoint Enabled.
[2]	**OUTTKIEN**	**Out Token Interrupt** 0 = The OUT token interrupt in Control Endpoint Disabled. 1 = The OUT token interrupt in Control Endpoint Enabled.
[1]	**SETUPPKIEN**	**Setup Packet Interrupt** 0 = The SETUP packet interrupt in Control Endpoint Disabled. 1 = The SETUP packet interrupt in Control Endpoint Enabled.
[0]	**SETUPTKIEN**	**Setup Token Interrupt Enable Control** 0 = The SETUP token interrupt in Control Endpoint Disabled. 1 = The SETUP token interrupt in Control Endpoint Enabled.

Control Endpoint Interrupt Status (HSUSBD_CEPINTSTS)

Register	Offset	R/W	Description	Reset Value
HSUSBD_CEPINTSTS	HSUSBD_BA+0x034	R/W	Control Endpoint Interrupt Status	0x0000_1800

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			BUFEMPTYIF	BUFFULLIF	STSDONEIF	ERRIF	STALLIF
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NAKIF	RXPKIF	TXPKIF	PINGIF	INTKIF	OUTTKIF	SETUPPKIF	SETUPTKIF

Bits	Description	
[31:13]	**Reserved**	Reserved.
[12]	**BUFEMPTYIF**	**Buffer Empty Interrupt** 0 = The control-endpoint buffer is not empty. 1 = The control-endpoint buffer is empty. **Note:** Write 1 to clear this bit to 0.
[11]	**BUFFULLIF**	**Buffer Full Interrupt** 0 = The control-endpoint buffer is not full. 1 = The control-endpoint buffer is full. **Note:** Write 1 to clear this bit to 0.
[10]	**STSDONEIF**	**Status Completion Interrupt** 0 = Not a USB transaction has completed successfully. 1 = The status stage of a USB transaction has completed successfully. **Note:** Write 1 to clear this bit to 0.
[9]	**ERRIF**	**USB Error Interrupt** 0 = No error had occurred during the transaction. 1 = An error had occurred during the transaction. **Note:** Write 1 to clear this bit to 0.
[8]	**STALLIF**	**STALL Sent Interrupt** 0 = Not a stall-token is sent in response to an IN/OUT token. 1 = A stall-token is sent in response to an IN/OUT token. **Note:** Write 1 to clear this bit to 0.

[7]	**NAKIF**	**NAK Sent Interrupt** 0 = Not a NAK-token is sent in response to an IN/OUT token. 1 = A NAK-token is sent in response to an IN/OUT token. **Note:** Write 1 to clear this bit to 0.
[6]	**RXPKIF**	**Data Packet Received Interrupt** 0 = Not a data packet is successfully received from the host for an OUT-token and an ACK is sent to the host. 1 = A data packet is successfully received from the host for an OUT-token and an ACK is sent to the host. **Note:** Write 1 to clear this bit to 0.
[5]	**TXPKIF**	**Data Packet Transmitted Interrupt** 0 = Not a data packet is successfully transmitted to the host in response to an IN-token and an ACK-token is received for the same. 1 = A data packet is successfully transmitted to the host in response to an IN-token and an ACK-token is received for the same. **Note:** Write 1 to clear this bit to 0.
[4]	**PINGIF**	**Ping Token Interrupt** 0 = The control-endpoint does not received a ping token from the host. 1 = The control-endpoint receives a ping token from the host. **Note:** Write 1 to clear this bit to 0.
[3]	**INTKIF**	**In Token Interrupt** 0 = The control-endpoint does not received an IN token from the host. 1 = The control-endpoint receives an IN token from the host. **Note:** Write 1 to clear this bit to 0.
[2]	**OUTTKIF**	**Out Token Interrupt** 0 = The control-endpoint does not received an OUT token from the host. 1 = The control-endpoint receives an OUT token from the host. **Note:** Write 1 to clear this bit to 0.
[1]	**SETUPPKIF**	**Setup Packet Interrupt** This bit must be cleared (by writing 1) before the next setup packet can be received. If the bit is not cleared, then the successive setup packets will be overwritten in the setup packet buffer. 0 = Not a Setup packet has been received from the host. 1 = A Setup packet has been received from the host. **Note:** Write 1 to clear this bit to 0.
[0]	**SETUPTKIF**	**Setup Token Interrupt** 0 = Not a Setup token is received. 1 = A Setup token is received. Writing 1 clears this status bit **Note:** Write 1 to clear this bit to 0.

Control Endpoint in Transfer Data Count (HSUSBD_CEPTXCNT)

Register	Offset	R/W	Description	Reset Value
HSUSBD_CEPTXCNT	HSUSBD_BA+0x038	R/W	Control Endpoint in Transfer Data Count	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**TXCNT**	**In-transfer Data Count** There is no mode selection for the control endpoint (but it operates like manual mode).The local-CPU has to fill the control-endpoint buffer with the data to be sent for an in-token and to write the count of bytes in this register. When zero is written into this field, a zero length packet is sent to the host. When the count written in the register is more than the MPS, the data sent will be of only MPS.

Control Endpoint Out Transfer Data Count (HSUSBD_CEPRXCNT)

Register	Offset	R/W	Description	Reset Value
HSUSBD_CEPRXCNT	HSUSBD_BA+0x03C	R	Control Endpoint Out Transfer Data Count	0x0000_0000

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**RXCNT**	**Out-transfer Data Count** The USB device controller maintains the count of the data received in case of an out transfer, during the control transfer.

Control Endpoint Data Count (HSUSBD_CEPDATCNT)

Register	Offset	R/W	Description	Reset Value
HSUSBD_CEPDATCNT	HSUSBD_BA+0x040	R	Control Endpoint Data Count	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**DATCNT**	**Control-endpoint Data Count** The USB device controller maintains the count of the data of control-endpoint.

Setup1 & Setup0 Bytes (HSUSBD_SETUP1_0)

Register	Offset	R/W	Description	Reset Value
HSUSBD_SETUP1_0	HSUSBD_BA+0x044	R	Setup1 & Setup0 Bytes	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SETUP1							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SETUP0							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**SETUP1**	**Setup Byte 1[15:8]** This register provides byte 1 of the last setup packet received. For a Standard Device Request, the following bRequest Code information is returned. 00000000 = Get Status. 00000001 = Clear Feature. 00000010 = Reserved. 00000011 = Set Feature. 00000100 = Reserved. 00000101 = Set Address. 00000110 = Get Descriptor. 00000111 = Set Descriptor. 00001000 = Get Configuration. 00001001 = Set Configuration. 00001010 = Get Interface. 00001011 = Set Interface. 00001100 = Synch Frame.

[7:0]	**SETUP0**	**Setup Byte 0[7:0]** This register provides byte 0 of the last setup packet received. For a Standard Device Request, the following bmRequestType information is returned. Bit 7(Direction): 0: Host to device 1: Device to host Bit 6-5 (Type): 00: Standard 01: Class 10: Vendor 11: Reserved Bit 4-0 (Recipient) 00000**:** Device 00001: Interface 00010: Endpoint 00011: Other Others: Reserved

Setup3 & Setup2 Bytes (HSUSBD_SETUP3_2)

Register	Offset	R/W	Description	Reset Value
HSUSBD_SETUP3_2	HSUSBD_BA+0x048	R	Setup3 & Setup2 Bytes	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SETUP3							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SETUP2							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**SETUP3**	**Setup Byte 3 [15:8]** This register provides byte 3 of the last setup packet received. For a Standard Device Request, the most significant byte of the wValue field is returned.
[7:0]	**SETUP2**	**Setup Byte 2 [7:0]** This register provides byte 2 of the last setup packet received. For a Standard Device Request, the least significant byte of the wValue field is returned.

Setup5 & Setup4 Bytes (HSUSBD_SETUP5_4)

Register	Offset	R/W	Description	Reset Value
HSUSBD_SETUP5_4	HSUSBD_BA+0x04C	R	Setup5 & Setup4 Bytes	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SETUP5							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SETUP4							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**SETUP5**	**Setup Byte 5[15:8]** This register provides byte 5 of the last setup packet received. For a Standard Device Request, the most significant byte of the wIndex field is returned.
[7:0]	**SETUP4**	**Setup Byte 4[7:0]** This register provides byte 4 of the last setup packet received. For a Standard Device Request, the least significant byte of the wIndex is returned.

Setup7 & Setup6 Bytes (HSUSBD_SETUP7_6)

Register	Offset	R/W	Description	Reset Value
HSUSBD_SETUP7_6	HSUSBD_BA+0x050	R	Setup7 & Setup6 Bytes	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SETUP7							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SETUP6							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**SETUP7**	**Setup Byte 7[15:8]** This register provides byte 7 of the last setup packet received. For a Standard Device Request, the most significant byte of the wLength field is returned.
[7:0]	**SETUP6**	**Setup Byte 6[7:0]** This register provides byte 6 of the last setup packet received. For a Standard Device Request, the least significant byte of the wLength field is returned.

Control Endpoint RAM Start Address Register (HSUSBD_CEPBUFSTART)

Register	Offset	R/W	Description	Reset Value
HSUSBD_CEPBUFSTART	HSUSBD_BA+0x054	R/W	Control Endpoint RAM Start 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				**SADDR**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SADDR							

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:0]	**SADDR**	**Control-endpoint Start Address** This is the start-address of the RAM space allocated for the control-endpoint.

Control Endpoint RAM End Address Register (HSUSBD_CEPBUFEND)

Register	Offset	R/W	Description	Reset Value
HSUSBD_CEPBUFEND	HSUSBD_BA+0x058	R/W	Control Endpoint RAM End 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				**EADDR**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EADDR							

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:0]	**EADDR**	**Control-endpoint End Address** This is the end-address of the RAM space allocated for the control-endpoint.

DMA Control Status Register (HSUSBD_DMACTL)

Register	Offset	R/W	Description	Reset Value
HSUSBD_DMACTL	HSUSBD_BA+0x05C	R/W	DMA Control 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							**SVINEP**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DMARST	**SGEN**	**DMAEN**	**DMARD**	**EPNUM**			

Bits	Description	
[31:8]	**Reserved**	Reserved.
[8]	**SVINEP**	**Serve IN Endpoint** **This bit is used to specify DMA serving endpoint-IN endpoint or OUT endpoint** 0 = DMA serves OUT endpoint. 1 = DMA serves IN endpoint.
[7]	**DMARST**	**Reset DMA State Machine** 0 = No reset the DMA state machine. 1 = Reset the DMA state machine.
[6]	**SGEN**	**Scatter Gather Function Enable Control** 0 = Scatter gather function Disabled. 1 = Scatter gather function Enabled.
[5]	**DMAEN**	**DMA Enable Control** 0 = DMA function Disabled. 1 = DMA function Enabled.
[4]	**DMARD**	**DMA Operation** 0 = The operation is a DMA write (read from USB buffer). DMA will check endpoint data available count (HSUSBD_EPxDATCNT) according to EPNM setting before to perform DMA write operation. 1 = The operation is a DMA read (write to USB buffer).
[3:0]	**EPNUM**	**DMA Endpoint Address Bits** Used to define the Endpoint Address

DMA Count Register (HSUSBD_DMACNT)

Register	Offset	R/W	Description	Reset Value
HSUSBD_DMACNT	HSUSBD_BA+0x060	R/W	DMA Count Register	0x0000_0000

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

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19:0]	**DMACNT**	**DMA Transfer Count** The transfer count of the DMA operation to be performed is written to this register.

Endpoint A~L Data Register (HSUSBD_EPADAT~ HSUSBD_EPLDAT)

Register	Offset	R/W	Description	Reset Value
HSUSBD_EPADAT	HSUSBD_BA+0x064	R/W	Endpoint A Data Register	0x0000_0000
HSUSBD_EPBDAT	HSUSBD_BA+0x08C	R/W	Endpoint B Data Register	0x0000_0000
HSUSBD_EPCDAT	HSUSBD_BA+0x0B4	R/W	Endpoint C Data Register	0x0000_0000
HSUSBD_EPDDAT	HSUSBD_BA+0x0DC	R/W	Endpoint D Data Register	0x0000_0000
HSUSBD_EPEDAT	HSUSBD_BA+0x104	R/W	Endpoint E Data Register	0x0000_0000
HSUSBD_EPFDAT	HSUSBD_BA+0x12C	R/W	Endpoint F Data Register	0x0000_0000
HSUSBD_EPGDAT	HSUSBD_BA+0x154	R/W	Endpoint G Data Register	0x0000_0000
HSUSBD_EPHDAT	HSUSBD_BA+0x17C	R/W	Endpoint H Data Register	0x0000_0000
HSUSBD_EPIDAT	HSUSBD_BA+0x1A4	R/W	Endpoint I Data Register	0x0000_0000
HSUSBD_EPJDAT	HSUSBD_BA+0x1CC	R/W	Endpoint J Data Register	0x0000_0000
HSUSBD_EPKDAT	HSUSBD_BA+0x1F4	R/W	Endpoint K Data Register	0x0000_0000
HSUSBD_EPLDAT	HSUSBD_BA+0x21C	R/W	Endpoint L Data Register	0x0000_0000

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

Bits	Description	
[31:0]	**EPDAT**	**Endpoint A~L Data Register** Endpoint A~L data buffer for the buffer transaction (read or write). **Note:** Only word or byte access are supported.

Endpoint A~L Interrupt Status Register (HSUSBD_EPAINTSTS~ HSUSBD_EPLINTSTS)

Register	Offset	R/W	Description	Reset Value
HSUSBD_EPAINTSTS	HSUSBD_BA+0x068	R/W	Endpoint A Interrupt Status Register	0x0000_0003
HSUSBD_EPBINTSTS	HSUSBD_BA+0x090	R/W	Endpoint B Interrupt Status Register	0x0000_0003
HSUSBD_EPCINTSTS	HSUSBD_BA+0x0B8	R/W	Endpoint C Interrupt Status Register	0x0000_0003
HSUSBD_EPDINTSTS	HSUSBD_BA+0x0E0	R/W	Endpoint D Interrupt Status Register	0x0000_0003
HSUSBD_EPEINTSTS	HSUSBD_BA+0x108	R/W	Endpoint E Interrupt Status Register	0x0000_0003
HSUSBD_EPFINTSTS	HSUSBD_BA+0x130	R/W	Endpoint F Interrupt Status Register	0x0000_0003
HSUSBD_EPGINTSTS	HSUSBD_BA+0x158	R/W	Endpoint G Interrupt Status Register	0x0000_0003
HSUSBD_EPHINTSTS	HSUSBD_BA+0x180	R/W	Endpoint H Interrupt Status Register	0x0000_0003
HSUSBD_EPIINTSTS	HSUSBD_BA+0x1A8	R/W	Endpoint I Interrupt Status Register	0x0000_0003
HSUSBD_EPJINTSTS	HSUSBD_BA+0x1D0	R/W	Endpoint J Interrupt Status Register	0x0000_0003
HSUSBD_EPKINTSTS	HSUSBD_BA+0x1F8	R/W	Endpoint K Interrupt Status Register	0x0000_0003
HSUSBD_EPLINTSTS	HSUSBD_BA+0x220	R/W	Endpoint L Interrupt Status Register	0x0000_0003

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			SHORTRXIF	ERRIF	NYETIF	STALLIF	NAKIF
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PINGIF	INTKIF	OUTTKIF	RXPKIF	TXPKIF	SHORTTXIF	BUFEMPTYIF	BUFFULLIF

Bits	Description	
[31:13]	**Reserved**	Reserved.
[12]	**SHORTRXIF**	**Bulk Out Short Packet Received** 0 = No bulk out short packet is received. 1 = Received bulk out short packet (including zero length packet). **Note:** Write 1 to clear this bit to 0.

[11]	**ERRIF**	**ERR Sent** 0 = No any error in the transaction. 1 = There occurs any error in the transaction. **Note:** Write 1 to clear this bit to 0.
[10]	**NYETIF**	**NYET Sent** 0 = The space available in the RAM is sufficient to accommodate the next on coming data packet. 1 = The space available in the RAM is not sufficient to accommodate the next on coming data packet. **Note:** Write 1 to clear this bit to 0.
[9]	**STALLIF**	**USB STALL Sent** 0 = The last USB packet could be accepted or provided because the endpoint was stalled, and was acknowledged with a STALL. 1 = The last USB packet could not be accepted or provided because the endpoint was stalled, and was acknowledged with a STALL. **Note:** Write 1 to clear this bit to 0.
[8]	**NAKIF**	**USB NAK Sent** 0 = The last USB IN packet could be provided, and was acknowledged with an ACK. 1 = The last USB IN packet could not be provided, and was acknowledged with a NAK. **Note:** Write 1 to clear this bit to 0.
[7]	**PINGIF**	**PING Token Interrupt** 0 = A Data PING token has not been received from the host. 1 = A Data PING token has been received from the host. **Note:** Write 1 to clear this bit to 0.
[6]	**INTKIF**	**Data IN Token Interrupt** 0 = Not Data IN token has been received from the host. 1 = A Data IN token has been received from the host. **Note:** Write 1 to clear this bit to 0.
[5]	**OUTTKIF**	**Data OUT Token Interrupt** 0 = A Data OUT token has not been received from the host. 1 = A Data OUT token has been received from the host. This bit also set by PING token (in high-speed only). **Note:** Write 1 to clear this bit to 0.
[4]	**RXPKIF**	**Data Packet Received Interrupt** 0 = No data packet is received from the host by the endpoint. 1 = A data packet is received from the host by the endpoint. **Note:** Write 1 to clear this bit to 0.
[3]	**TXPKIF**	**Data Packet Transmitted Interrupt** 0 = Not a data packet is transmitted from the endpoint to the host. 1 = A data packet is transmitted from the endpoint to the host. **Note:** Write 1 to clear this bit to 0.

[2]	**SHORTTXIF**	**Short Packet Transferred Interrupt** 0 = The length of the last packet was not less than the Maximum Packet Size (EPMPS). 1 = The length of the last packet was less than the Maximum Packet Size (EPMPS). **Note:** Write 1 to clear this bit to 0.
[1]	**BUFEMPTYIF**	**Buffer Empty** For an IN endpoint, a buffer is available to the local side for writing up to FIFO full of bytes. 0 = The endpoint buffer is not empty. 1 = The endpoint buffer is empty. For an OUT endpoint: 0 = The currently selected buffer has not a count of 0. 1 = The currently selected buffer has a count of 0, or no buffer is available on the local side (nothing to read). **Note:** This bit is read-only.
[0]	**BUFFULLIF**	**Buffer Full** For an IN endpoint, the currently selected buffer is full, or no buffer is available to the local side for writing (no space to write). For an OUT endpoint, there is a buffer available on the local side, and there are FIFO full of bytes available to be read (entire packet is available for reading). 0 = The endpoint packet buffer is not full. 1 = The endpoint packet buffer is full. **Note:** This bit is read-only.

Endpoint A~L Interrupt Enable Control Register (HSUSBD_EPAINTEN~ HSUSBD_EPLINTEN)

Register	Offset	R/W	Description	Reset Value
HSUSBD_EPAINTEN	HSUSBD_BA+0x06C	R/W	Endpoint A Interrupt Enable Register	0x0000_0000
HSUSBD_EPBINTEN	HSUSBD_BA+0x094	R/W	Endpoint B Interrupt Enable Register	0x0000_0000
HSUSBD_EPCINTEN	HSUSBD_BA+0x0BC	R/W	Endpoint C Interrupt Enable Register	0x0000_0000
HSUSBD_EPDINTEN	HSUSBD_BA+0x0E4	R/W	Endpoint D Interrupt Enable Register	0x0000_0000
HSUSBD_EPEINTEN	HSUSBD_BA+0x10C	R/W	Endpoint E Interrupt Enable Register	0x0000_0000
HSUSBD_EPFINTEN	HSUSBD_BA+0x134	R/W	Endpoint F Interrupt Enable Register	0x0000_0000
HSUSBD_EPGINTEN	HSUSBD_BA+0x15C	R/W	Endpoint G Interrupt Enable Register	0x0000_0000
HSUSBD_EPHINTEN	HSUSBD_BA+0x184	R/W	Endpoint H Interrupt Enable Register	0x0000_0000
HSUSBD_EPIINTEN	HSUSBD_BA+0x1AC	R/W	Endpoint I Interrupt Enable Register	0x0000_0000
HSUSBD_EPJINTEN	HSUSBD_BA+0x1D4	R/W	Endpoint J Interrupt Enable Register	0x0000_0000
HSUSBD_EPKINTEN	HSUSBD_BA+0x1FC	R/W	Endpoint K Interrupt Enable Register	0x0000_0000
HSUSBD_EPLINTEN	HSUSBD_BA+0x224	R/W	Endpoint L 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			SHORTRXIEN	ERRIEN	NYETIEN	STALLIEN	NAKIEN
7	6	5	4	3	2	1	0
PINGIEN	INTKIEN	OUTTKIEN	RXPKIEN	TXPKIEN	SHORTTXIEN	BUFEMPTYIEN	BUFFULLIEN

Bits	Description	
[31:13]	**Reserved**	Reserved.
[12]	**SHORTRXIEN**	**Bulk Out Short Packet Interrupt Enable Control** When set, this bit enables a local interrupt to be set whenever bulk out short packet occurs on the bus for this endpoint. 0 = Bulk out interrupt Disabled. 1 = Bulk out interrupt Enabled.

[11]	**ERRIEN**	**ERR Interrupt Enable Control** When set, this bit enables a local interrupt to be set whenever ERR condition occurs on the bus for this endpoint. 0 = Error event interrupt Disabled. 1 = Error event interrupt Enabled.
[10]	**NYETIEN**	**NYET Interrupt Enable Control** When set, this bit enables a local interrupt to be set whenever NYET condition occurs on the bus for this endpoint. 0 = NYET condition interrupt Disabled. 1 = NYET condition interrupt Enabled.
[9]	**STALLIEN**	**USB STALL Sent Interrupt Enable Control** When set, this bit enables a local interrupt to be set when a stall token is sent to the host. 0 = STALL token interrupt Disabled. 1 = STALL token interrupt Enabled.
[8]	**NAKIEN**	**USB NAK Sent Interrupt Enable Control** When set, this bit enables a local interrupt to be set when a NAK token is sent to the host. 0 = NAK token interrupt Disabled. 1 = NAK token interrupt Enabled.
[7]	**PINGIEN**	**PING Token Interrupt Enable Control** When set, this bit enables a local interrupt to be set when a PING token has been received from the host. 0 = PING token interrupt Disabled. 1 = PING token interrupt Enabled.
[6]	**INTKIEN**	**Data IN Token Interrupt Enable Control** When set, this bit enables a local interrupt to be set when a Data IN token has been received from the host. 0 = Data IN token interrupt Disabled. 1 = Data IN token interrupt Enabled.
[5]	**OUTTKIEN**	**Data OUT Token Interrupt Enable Control** When set, this bit enables a local interrupt to be set when a Data OUT token has been received from the host. 0 = Data OUT token interrupt Disabled. 1 = Data OUT token interrupt Enabled.
[4]	**RXPKIEN**	**Data Packet Received Interrupt Enable Control** When set, this bit enables a local interrupt to be set when a data packet has been received from the host. 0 = Data packet has been received from the host interrupt Disabled. 1 = Data packet has been received from the host interrupt Enabled.
[3]	**TXPKIEN**	**Data Packet Transmitted Interrupt Enable Control** When set, this bit enables a local interrupt to be set when a data packet has been transmitted to the host. 0 = Data packet has been transmitted to the host interrupt Disabled. 1 = Data packet has been transmitted to the host interrupt Enabled.

[2]	**SHORTTXIEN**	**Short Packet Transferred Interrupt Enable Control** When set, this bit enables a local interrupt to be set when a short data packet has been transferred to/from the host. 0 = Short data packet interrupt Disabled. 1 = Short data packet interrupt Enabled.
[1]	**BUFEMPTYIEN**	**Buffer Empty Interrupt** When set, this bit enables a local interrupt to be set when a buffer empty condition is detected on the bus. 0 = Buffer empty interrupt Disabled. 1 = Buffer empty interrupt Enabled.
[0]	**BUFFULLIEN**	**Buffer Full Interrupt** When set, this bit enables a local interrupt to be set when a buffer full condition is detected on the bus. 0 = Buffer full interrupt Disabled. 1 = Buffer full interrupt Enabled.

Endpoint A~L Data Available Count Register (HSUSBD_EPADATCNT~ HSUSBD_EPLDATCNT)

Register	Offset	R/W	Description	Reset Value
HSUSBD_EPADATCNT	HSUSBD_BA+0x070	R	Endpoint A Data Available Count Register	0x0000_0000
HSUSBD_EPBDATCNT	HSUSBD_BA+0x098	R	Endpoint B Data Available Count Register	0x0000_0000
HSUSBD_EPCDATCNT	HSUSBD_BA+0x0C0	R	Endpoint C Data Available Count Register	0x0000_0000
HSUSBD_EPDDATCNT	HSUSBD_BA+0x0E8	R	Endpoint D Data Available Count Register	0x0000_0000
HSUSBD_EPEDATCNT	HSUSBD_BA+0x110	R	Endpoint E Data Available Count Register	0x0000_0000
HSUSBD_EPFDATCNT	HSUSBD_BA+0x138	R	Endpoint F Data Available Count Register	0x0000_0000
HSUSBD_EPGDATCNT	HSUSBD_BA+0x160	R	Endpoint G Data Available Count Register	0x0000_0000
HSUSBD_EPHDATCNT	HSUSBD_BA+0x188	R	Endpoint H Data Available Count Register	0x0000_0000
HSUSBD_EPIDATCNT	HSUSBD_BA+0x1B0	R	Endpoint I Data Available Count Register	0x0000_0000
HSUSBD_EPJDATCNT	HSUSBD_BA+0x1D8	R	Endpoint J Data Available Count Register	0x0000_0000
HSUSBD_EPKDATCNT	HSUSBD_BA+0x200	R	Endpoint K Data Available Count Register	0x0000_0000
HSUSBD_EPLDATCNT	HSUSBD_BA+0x228	R	Endpoint L Data Available Count Register	0x0000_0000

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

Bits	Description	
[31]	**Reserved**	Reserved.

[30:16]	**DMALOOP**	**DMA Loop** This register is the remaining DMA loop to complete. Each loop means 32-byte transfer.
[15:0]	**DATCNT**	**Data Count** For an IN endpoint (EPDIR(HSUSBD_EPxCFG[3] is high.), this register returns the number of valid bytes in the IN endpoint packet buffer. For an OUT endpoint (EPDIR(HSUSBD_EPxCFG[3] is low.), this register returns the number of received valid bytes in the Host OUT transfer.

Endpoint A~L Response Control Register (HSUSBD_EPARSPCTL~ HSUSBD_EPLRSPCTL)

Register	Offset	R/W	Description	Reset Value
HSUSBD_EPARSPCTL	HSUSBD_BA+0x074	R/W	Endpoint A Response Control Register	0x0000_0000
HSUSBD_EPBRSPCTL	HSUSBD_BA+0x09C	R/W	Endpoint B Response Control Register	0x0000_0000
HSUSBD_EPCRSPCTL	HSUSBD_BA+0x0C4	R/W	Endpoint C Response Control Register	0x0000_0000
HSUSBD_EPDRSPCTL	HSUSBD_BA+0x0EC	R/W	Endpoint D Response Control Register	0x0000_0000
HSUSBD_EPERSPCTL	HSUSBD_BA+0x114	R/W	Endpoint E Response Control Register	0x0000_0000
HSUSBD_EPFRSPCTL	HSUSBD_BA+0x13C	R/W	Endpoint F Response Control Register	0x0000_0000
HSUSBD_EPGRSPCTL	HSUSBD_BA+0x164	R/W	Endpoint G Response Control Register	0x0000_0000
HSUSBD_EPHRSPCTL	HSUSBD_BA+0x18C	R/W	Endpoint H Response Control Register	0x0000_0000
HSUSBD_EPIRSPCTL	HSUSBD_BA+0x1B4	R/W	Endpoint I Response Control Register	0x0000_0000
HSUSBD_EPJRSPCTL	HSUSBD_BA+0x1DC	R/W	Endpoint J Response Control Register	0x0000_0000
HSUSBD_EPKRSPCTL	HSUSBD_BA+0x204	R/W	Endpoint K Response Control Register	0x0000_0000
HSUSBD_EPLRSPCTL	HSUSBD_BA+0x22C	R/W	Endpoint L Response 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**
DISBUF	**SHORTTXEN**	**ZEROLEN**	**HALT**	**TOGGLE**	**MODE**		**FLUSH**

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**DISBUF**	**Buffer Disable Control** This bit is used to receive unknown size OUT short packet. The received packet size is reference HSUSBD_EPxDATCNT register. 0 = Buffer Not Disabled when Bulk-OUT short packet is received. 1 = Buffer Disabled when Bulk-OUT short packet is received.

[6]	**SHORTTXEN**	**Short Packet Transfer Enable** This bit is applicable only in case of Auto-Validate Method. This bit is set to validate any remaining data in the buffer which is not equal to the MPS of the endpoint, and happens to be the last transfer. This bit gets cleared once the data packet is sent. 0 = Not validate any remaining data in the buffer which is not equal to the MPS of the endpoint. 1 = Validate any remaining data in the buffer which is not equal to the MPS of the endpoint.
[5]	**ZEROLEN**	**Zero Length** This bit is used to send a zero-length packet response to an IN-token. When this bit is set, a zero packet is sent to the host on reception of an IN-token. This bit gets cleared once the zero length data packet is sent. 0 = A zero packet is not sent to the host on reception of an IN-token. 1 = A zero packet is sent to the host on reception of an IN-token.
[4]	**HALT**	**Endpoint Halt** This bit is used to send a STALL handshake as response to the token from the host. When an Endpoint Set Feature (ep_halt) is detected by the local CPU, it must write a ‘1’ to this bit. 0 = Not send a STALL handshake as response to the token from the host. 1 = Send a STALL handshake as response to the token from the host.
[3]	**TOGGLE**	**Endpoint Toggle** This bit is used to clear the endpoint data toggle bit. Reading this bit returns the current state of the endpoint data toggle bit. The local CPU may use this bit to initialize the end-point’s toggle in case of reception of a Set Interface request or a Clear Feature (ep_halt) request from the host. Only when toggle bit is “1”, this bit can be written into the inversed write data bit[3]. 0 = Not clear the endpoint data toggle bit. 1 = Clear the endpoint data toggle bit.
[2:1]	**MODE**	**Mode Control** The two bits decide the operation mode of the in-endpoint. 00: Auto-Validate Mode 01: Manual-Validate Mode 10: Fly Mode 11: Reserved These bits are not valid for an out-endpoint. The auto validate mode will be activated when the reserved mode is selected.
[0]	**FLUSH**	**Buffer Flush** Writing 1 to this bit causes the packet buffer to be flushed and the corresponding EP_AVAIL register to be cleared. This bit is self-clearing. This bit should always be written after an configuration event. 0 = The packet buffer is not flushed. 1 = The packet buffer is flushed by user.

Endpoint A~L Maximum Packet Size Register (HSUSBD_EPAMPS~ HSUSBD_EPLMPS)

Register	Offset	R/W	Description	Reset Value
HSUSBD_EPAMPS	HSUSBD_BA+0x078	R/W	Endpoint A Maximum Packet Size Register	0x0000_0000
HSUSBD_EPBMPS	HSUSBD_BA+0x0A0	R/W	Endpoint B Maximum Packet Size Register	0x0000_0000
HSUSBD_EPCMPS	HSUSBD_BA+0x0C8	R/W	Endpoint C Maximum Packet Size Register	0x0000_0000
HSUSBD_EPDMPS	HSUSBD_BA+0x0F0	R/W	Endpoint D Maximum Packet Size Register	0x0000_0000
HSUSBD_EPEMPS	HSUSBD_BA+0x118	R/W	Endpoint E Maximum Packet Size Register	0x0000_0000
HSUSBD_EPFMPS	HSUSBD_BA+0x140	R/W	Endpoint F Maximum Packet Size Register	0x0000_0000
HSUSBD_EPGMPS	HSUSBD_BA+0x168	R/W	Endpoint G Maximum Packet Size Register	0x0000_0000
HSUSBD_EPHMPS	HSUSBD_BA+0x190	R/W	Endpoint H Maximum Packet Size Register	0x0000_0000
HSUSBD_EPIMPS	HSUSBD_BA+0x1B8	R/W	Endpoint I Maximum Packet Size Register	0x0000_0000
HSUSBD_EPJMPS	HSUSBD_BA+0x1E0	R/W	Endpoint J Maximum Packet Size Register	0x0000_0000
HSUSBD_EPKMPS	HSUSBD_BA+0x208	R/W	Endpoint K Maximum Packet Size Register	0x0000_0000
HSUSBD_EPLMPS	HSUSBD_BA+0x230	R/W	Endpoint L Maximum Packet Size Register	0x0000_0000

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

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10:0]	**EPMPS**	**Endpoint Maximum Packet Size** This field determines the Maximum Packet Size of the Endpoint.

Endpoint A~L Transfer Count Register (HSUSBD_EPATXCNT~ HSUSBD_EPLTXCNT)

Register	Offset	R/W	Description	Reset Value
HSUSBD_EPATXCNT	HSUSBD_BA+0x07C	R/W	Endpoint A Transfer Count Register	0x0000_0000
HSUSBD_EPBTXCNT	HSUSBD_BA+0x0A4	R/W	Endpoint B Transfer Count Register	0x0000_0000
HSUSBD_EPCTXCNT	HSUSBD_BA+0x0CC	R/W	Endpoint C Transfer Count Register	0x0000_0000
HSUSBD_EPDTXCNT	HSUSBD_BA+0x0F4	R/W	Endpoint D Transfer Count Register	0x0000_0000
HSUSBD_EPETXCNT	HSUSBD_BA+0x11C	R/W	Endpoint E Transfer Count Register	0x0000_0000
HSUSBD_EPFTXCNT	HSUSBD_BA+0x144	R/W	Endpoint F Transfer Count Register	0x0000_0000
HSUSBD_EPGTXCNT	HSUSBD_BA+0x16C	R/W	Endpoint G Transfer Count Register	0x0000_0000
HSUSBD_EPHTXCNT	HSUSBD_BA+0x194	R/W	Endpoint H Transfer Count Register	0x0000_0000
HSUSBD_EPITXCNT	HSUSBD_BA+0x1BC	R/W	Endpoint I Transfer Count Register	0x0000_0000
HSUSBD_EPJTXCNT	HSUSBD_BA+0x1E4	R/W	Endpoint J Transfer Count Register	0x0000_0000
HSUSBD_EPKTXCNT	HSUSBD_BA+0x20C	R/W	Endpoint K Transfer Count Register	0x0000_0000
HSUSBD_EPLTXCNT	HSUSBD_BA+0x234	R/W	Endpoint L Transfer Count Register	0x0000_0000

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

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10:0]	**TXCNT**	**Endpoint Transfer Count** For IN endpoints, this field determines the total number of bytes to be sent to the host in case of manual validation method. For OUT endpoints, this field has no effect.

Endpoint A~L Configuration Register (HSUSBD_EPACFG~ HSUSBD_EPLCFG)

Register	Offset	R/W	Description	Reset Value
HSUSBD_EPACFG	HSUSBD_BA+0x080	R/W	Endpoint A Configuration Register	0x0000_0012
HSUSBD_EPBCFG	HSUSBD_BA+0x0A8	R/W	Endpoint B Configuration Register	0x0000_0022
HSUSBD_EPCCFG	HSUSBD_BA+0x0D0	R/W	Endpoint C Configuration Register	0x0000_0032
HSUSBD_EPDCFG	HSUSBD_BA+0x0F8	R/W	Endpoint D Configuration Register	0x0000_0042
HSUSBD_EPECFG	HSUSBD_BA+0x120	R/W	Endpoint E Configuration Register	0x0000_0052
HSUSBD_EPFCFG	HSUSBD_BA+0x148	R/W	Endpoint F Configuration Register	0x0000_0062
HSUSBD_EPGCFG	HSUSBD_BA+0x170	R/W	Endpoint G Configuration Register	0x0000_0072
HSUSBD_EPHCFG	HSUSBD_BA+0x198	R/W	Endpoint H Configuration Register	0x0000_0082
HSUSBD_EPICFG	HSUSBD_BA+0x1C0	R/W	Endpoint I Configuration Register	0x0000_0092
HSUSBD_EPJCFG	HSUSBD_BA+0x1E8	R/W	Endpoint J Configuration Register	0x0000_00A2
HSUSBD_EPKCFG	HSUSBD_BA+0x210	R/W	Endpoint K Configuration Register	0x0000_00B2
HSUSBD_EPLCFG	HSUSBD_BA+0x238	R/W	Endpoint L Configuration Register	0x0000_00C2

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:4]	**EPNUM**	**Endpoint Number** This field selects the number of the endpoint. Valid numbers 1 to 15. **Note:** Two endpoints having the same endpoint number is not supported.
[3]	**EPDIR**	**Endpoint Direction** 0 = Out-endpoint (Host OUT to Device). 1 = In-endpoint (Host IN to Device). **Note:** A maximum of one OUT and IN endpoint is allowed for each endpoint number.

[2:1]	**EPTYPE**	**Endpoint Type** This field selects the type of this endpoint. Endpoint 0 is forced to a Control type. 00 = Reserved. 01 = Bulk. 10 = Interrupt. 11 = Isochronous.
[0]	**EPEN**	**Endpoint Valid** When set, this bit enables this endpoint. This bit has no effect on Endpoint 0, which is always enabled. 0 = The endpoint Disabled. 1 = The endpoint Enabled.

Endpoint A~L RAM Start Address Register (HSUSBD_EPABUFSTART~ HSUSBD_EPLBUFSTART)

Register	Offset	R/W	Description	Reset Value
HSUSBD_EPABUFSTART	HSUSBD_BA+0x084	R/W	Endpoint A RAM Start Address Register	0x0000_0000
HSUSBD_EPBBUFSTART	HSUSBD_BA+0x0AC	R/W	Endpoint B RAM Start Address Register	0x0000_0000
HSUSBD_EPCBUFSTART	HSUSBD_BA+0x0D4	R/W	Endpoint C RAM Start Address Register	0x0000_0000
HSUSBD_EPDBUFSTART	HSUSBD_BA+0x0FC	R/W	Endpoint D RAM Start Address Register	0x0000_0000
HSUSBD_EPEBUFSTART	HSUSBD_BA+0x124	R/W	Endpoint E RAM Start Address Register	0x0000_0000
HSUSBD_EPFBUFSTART	HSUSBD_BA+0x14C	R/W	Endpoint F RAM Start Address Register	0x0000_0000
HSUSBD_EPGBUFSTART	HSUSBD_BA+0x174	R/W	Endpoint G RAM Start Address Register	0x0000_0000
HSUSBD_EPHBUFSTART	HSUSBD_BA+0x19C	R/W	Endpoint H RAM Start Address Register	0x0000_0000
HSUSBD_EPIBUFSTART	HSUSBD_BA+0x1C4	R/W	Endpoint I RAM Start Address Register	0x0000_0000
HSUSBD_EPJBUFSTART	HSUSBD_BA+0x1EC	R/W	Endpoint J RAM Start Address Register	0x0000_0000
HSUSBD_EPKBUFSTART	HSUSBD_BA+0x214	R/W	Endpoint K RAM Start Address Register	0x0000_0000
HSUSBD_EPLBUFSTART	HSUSBD_BA+0x23C	R/W	Endpoint L RAM Start Address Register	0x0000_0000

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

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:0]	**SADDR**	**Endpoint Start Address** This is the start-address of the RAM space allocated for the endpoint A~L.

Endpoint A~L RAM End Address Register (HSUSBD_EPABUFEND~ HSUSBD_EPLBUFEND)

Register	Offset	R/W	Description	Reset Value
HSUSBD_EPABUFEND	HSUSBD_BA+0x088	R/W	Endpoint A RAM End Address Register	0x0000_0000
HSUSBD_EPBBUFEND	HSUSBD_BA+0x0B0	R/W	Endpoint B RAM End Address Register	0x0000_0000
HSUSBD_EPCBUFEND	HSUSBD_BA+0x0D8	R/W	Endpoint C RAM End Address Register	0x0000_0000
HSUSBD_EPDBUFEND	HSUSBD_BA+0x100	R/W	Endpoint D RAM End Address Register	0x0000_0000
HSUSBD_EPEBUFEND	HSUSBD_BA+0x128	R/W	Endpoint E RAM End Address Register	0x0000_0000
HSUSBD_EPFBUFEND	HSUSBD_BA+0x150	R/W	Endpoint F RAM End Address Register	0x0000_0000
HSUSBD_EPGBUFEND	HSUSBD_BA+0x178	R/W	Endpoint G RAM End Address Register	0x0000_0000
HSUSBD_EPHBUFEND	HSUSBD_BA+0x1A0	R/W	Endpoint H RAM End Address Register	0x0000_0000
HSUSBD_EPIBUFEND	HSUSBD_BA+0x1C8	R/W	Endpoint I RAM End Address Register	0x0000_0000
HSUSBD_EPJBUFEND	HSUSBD_BA+0x1F0	R/W	Endpoint J RAM End Address Register	0x0000_0000
HSUSBD_EPKBUFEND	HSUSBD_BA+0x218	R/W	Endpoint K RAM End Address Register	0x0000_0000
HSUSBD_EPLBUFEND	HSUSBD_BA+0x240	R/W	Endpoint L RAM End Address Register	0x0000_0000

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

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:0]	**EADDR**	**Endpoint End Address** This is the end-address of the RAM space allocated for the endpoint A~L.

AHB Address Register (HSUSBD_DMAADDR)

Register	Offset	R/W	Description	Reset Value
HSUSBD_DMAADDR	HSUSBD_BA+0x700	R/W	AHB DMA Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**DMAADDR**	**DMAADDR** The register specifies the address from which the DMA has to read / write. The address must WORD (32-bit) aligned.

USB PHY Control Register (HSUSBD_PHYCTL)

Register	Offset	R/W	Description	Reset Value
HSUSBD_PHYCTL	HSUSBD_BA+0x704	R/W	USB PHY Control Register	0x0000_0420

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

Bits	Description	
[31]	**VBUSDET**	**VBUS Status** 0 = The VBUS is not detected yet. 1 = The VBUS is detected.
[30:27]	**Reserved**	Reserved.
[26]	**STALL_REVERT**	**Stall Revert Write Pointer Enable Bit** 0 = The Stall revert write pointer function Disabled. 1 = The Stall revert write pointer function Enabled.
[25]	**LINESTATEWKEN**	**Line State Wake-up Enable Bit** 0 = The Line State wake-up function Disabled. 1 = The Line State wake-up function Enabled.
[24]	**VBUSWKEN**	**VBUS Wake-up Enable Bit** 0 = The wake-up function Disabled. 1 = The wake-up function Enabled.
[23:10]	**Reserved**	Reserved.
[9]	**PHYEN**	**PHY Suspend Enable Bit** 0 = The USB PHY is suspend. 1 = The USB PHY is not suspend.
[8]	**DPPUEN**	**DP Pull-up** 0 = Pull-up resistor on D+ Disabled. 1 = Pull-up resistor on D+ Enabled.
[7:0]	**Reserved**	Reserved.

6.22 USB 2.0 Host Controller (USBH)

6.22.1 Overview

This chip is equipped with a USB 2.0 HS/FS Host Controller (USBH) that supports Enhanced Host Controller Interface (EHCI) and Open Host Controller Interface (OpenHCI, OHCI) Specification, a register-level description of a host controller, to manage the devices and data transfer of Universal Serial Bus (USB).

The USBH supports an integrated Root Hub with eight USB ports (two ports with on-chip USB 2.0 high speed transceiver and up to six USB 1.1 Host Lite ports), a DMA for real-time data transfer between system memory and USB bus, port power control and port over current detection.

The USBH is responsible for detecting the connect and disconnect of USB devices, managing data transfer, collecting status and activity of USB bus, providing power control and detecting over current of attached USB devices.

6.22.2 Features

- Compliant with Universal Serial Bus (USB) Specification Revision 2.0.
- Supports Enhanced Host Controller Interface (EHCI) Specification Revision 1.0
- Supports Open Host Controller Interface (OpenHCI) Specification Revision 1.0.
- Supports high-speed (480Mbps), full-speed (12Mbps) and low-speed (1.5Mbps) USB devices.
- Supports Control, Bulk, Interrupt, Isochronous and Split transfers.
- Supports an integrated Root Hub.
- Supports a port routing logic to route full/low speed device to OHCI controller.
- Supports a USB host port with on-chip USB2.0 high speed transceiver shared with USB device (dual-role function).
- Supports a USB host only port with on-chip USB2.0 high speed transceiver.
- Supports up to six USB 1.1 Host Lite ports.
- Supports port power control and port over current detection.
- Supports DMA for real-time data transfer.

6.22.3 Block Diagram

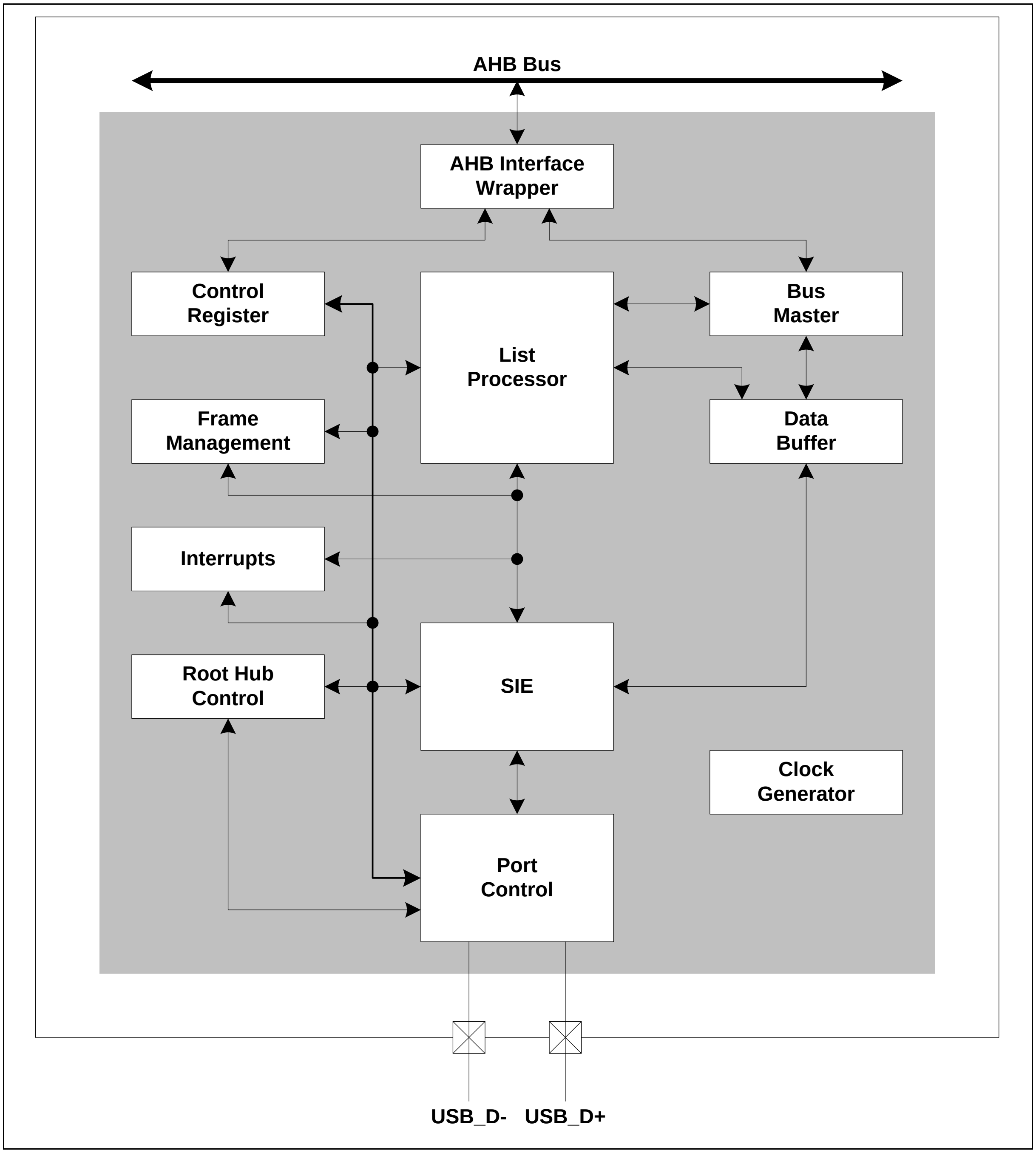


Figure 6.22-1 USB 2.0 FS Host Controller Block Diagram

6.22.4 Basic Configuration

6.22.4.1 USBH EHCI Controller Basic Configuration

- Clock Source Configuration
  - Enable USBH engine clock in USBH (CLK_HCLKEN[18]).
- Reset Configuration
  - Reset USBH controller in USBHRST (SYS_AHBIPRST[18]).

6.22.4.2 USBH OHCI Controller Basic Configuration

- Clock Source Configuration
  - Enable USBH engine clock in USBH (CLK_HCLKEN[18]).
- Reset Configuration
  - Reset USBH controller in USBHRST (SYS_AHBIPRST[18]).
- Pin Configuration

Group	Pin Name	GPIO	MFP	Type	Description
USBHL0	USBHL0_DM	PB.6	MFP4	A	USBH differential signal D-.
		PB.7	MFP4	A	
		PB.9	MFP4	A	
		PD.14	MFP5	A	
	USBHL0_DP	PB.4	MFP4	A	USBH differential signal D+.
		PB.5	MFP4	A	
		PB.10	MFP4	A	
		PD.15	MFP5	A	
USBHL1	USBHL1_DM	PF.0	MFP6	A	USBH differential signal D-.
		PE.0	MFP6	A	
	USBHL1_DP	PF.1	MFP6	A	USBH differential signal D+.
		PE.1	MFP6	A	
USBHL2	USBHL2_DM	PF.2	MFP6	A	USBH differential signal D-.
		PE.2	MFP6	A	
	USBHL2_DP	PF.3	MFP6	A	USBH differential signal D+.
		PE.3	MFP6	A	
USBHL3	USBHL3_DM	PF.4	MFP6	A	USBH differential signal D-.
		PE.4	MFP6	A	
	USBHL3_DP	PF.5	MFP6	A	USBH differential signal D+.
		PE.5	MFP6	A	
USBHL4	USBHL4_DM	PA.15	MFP4	A	USBH differential signal D-.

Group	Pin Name	GPIO	MFP	Type	Description
		PF.6	MFP6	A	
		PE.6	MFP6	A	
	USBHL4_DP	PG.10	MFP4	A	USBH differential signal D+.
		PB.13	MFP6	A	
		PF.7	MFP6	A	
		PE.7	MFP6	A	
USBHL5	USBHL5_DM	PA.13	MFP4	A	USBH differential signal D-.
		PB.11	MFP4	A	
		PF.8	MFP6	A	
		PE.8	MFP6	A	
	USBHL5_DP	PA.14	MFP4	A	USBH differential signal D+.
		PB.12	MFP4	A	
		PF.9	MFP6	A	
		PE.9	MFP6	A	
USBH	USBH_PWREN	PE.12	MFP1	O	HSUSB external VBUS regulator enable pin.

6.22.5 Functional Description

6.22.5.1 EHCI Controller

The EHCI is interfaced with the system through AHB interface. Whenever the CPU wants to initiate a register read or register write, it uses the AHB slave I/F signals and performs the necessary operation (register read writes). The CPU acts as a bus master, having initiated this transfer. At that time, EHCI acts as a target and responds to the transfer initiated by the system software. For example, if the CPU wants to write into one of the memory mapped registers of EHCI, it says the address and value to be written into that addressed register. EHCI targets the register by using that address and fills the register with the value specified by the software. If it is a register read, EHCI gets the value from the addressed register and puts it on the system bus.

Likewise, when the EHCI wants to perform a data transfer, it acts as a master and initiates a data transfer. At that time, the system memory acts as a bus target. EHCI, as a master can perform two types of data transfers, from EHCI to the system memory and from system memory to the EHCI. When the EHCI wants the data to be moved from the downstream USB2.0 device to the system memory, it initiates a memory write transfer by accessing the memory interfacing signals. EHCI writes the control word (write), data and data count to be moved to the system memory. The memory controller accepts the data and moves it to the memory. If the data has to be moved from memory to the downstream device, the EHCI issue a read transfer to system bus. The memory controller gives data through the memory interfacing signals. EHCI accepts the data and moves them to the downstream device.

6.22.5.2 OHCI Controller

AHB Interface

The OpenHCI Host Controller is connected to the system by the AHB bus. The design requires both master and slave bus operations. As a master, the Host Controller is responsible for running cycles on the AHB bus to access EDs and TDs as well as transferring data between memory and the local data buffer. As a slave, the Host Controller monitors the cycles on the AHB bus and determines when to respond to these cycles. Configuration and non-real-time control access to the Host Controller operational registers are through the AHB bus slave interface.

Host Controller

The host controller includes 5 functional blocks, including List Processing, Frame Management, Interrupt Processing, Host Controller Bus Master and Data Buffer.

The List Processor manages the data structures from the Host Controller Driver and coordinates all activity within the Host Controller.

The Frame Management is responsible for managing the frame specific tasks required by the USB specification and the OpenHCI specification. These tasks are:

- Management of the OpenHCI frame specific Operational Registers
- Operation of the Largest Data Packet Counter.
- Performing frame qualifications on USB Transaction requests to the SIE.
- Generate SOF token requests to the SIE.

Interrupts are the communication method for HC-initiated communication with the Host Controller Driver. There are several events that may trigger an interrupt from the Host Controller. Each specific event sets a specific bit in the HcInterruptStatus register.

The Host Controller Bus Master is the central block in the data path. The Host Controller Bus Master coordinates all access to the AHB Interface. There are two sources of bus mastering within Host Controller: the List Processor and the Data Buffer Engine.

The Data Buffer serves as the data interface between the Bus Master and the SIE. It is a combination of a 64-byte latched based bi-directional asynchronous FIFO and a single DWORD AHB Holding Register.

USB Interface

The USB interface includes the integrated Root Hub with an USB port, Port 1 as well as the Serial Interface Engine (SIE) and USB clock generator. The interface combines responsibility for executing bus transactions requested by the HC as well as the hub and port management specified by USB.

The SIE is responsible for managing all transactions to the USB. It controls the bus protocol, packet generation/extraction, data parallel-to-serial conversion, CRC coding, bit stuffing, and NRZI encoding. All transactions on the USB are requested from the List Processor and Frame Manager.

The Root Hub is a collection of ports that are individually controlled and a hub that maintains control/status over functions common to all ports.

6.22.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
USBH Base Address: **USBH_BA = 0xB001_7000** **HSUSBH_BA = 0xB001_5000**				
HcRevision	USBH_BA+0x000	R	Host Controller Revision Register	0x0000_0110
HcControl	USBH_BA+0x004	R/W	Host Controller Control Register	0x0000_0000
HcCommandStatus	USBH_BA+0x008	R/W	Host Controller Command Status Register	0x0000_0000
HcInterruptStatus	USBH_BA+0x00C	R/W	Host Controller Interrupt Status Register	0x0000_0000
HcInterruptEnable	USBH_BA+0x010	R/W	Host Controller Interrupt Enable Register	0x0000_0000
HcInterruptDisable	USBH_BA+0x014	R/W	Host Controller Interrupt Disable Register	0x0000_0000
HcHCCA	USBH_BA+0x018	R/W	Host Controller Communication Area Register	0x0000_0000
HcPeriodCurrentED	USBH_BA+0x01C	R/W	Host Controller Period Current ED Register	0x0000_0000
HcControlHeadED	USBH_BA+0x020	R/W	Host Controller Control Head ED Register	0x0000_0000
HcControlCurrentED	USBH_BA+0x024	R/W	Host Controller Control Current ED Register	0x0000_0000
HcBulkHeadED	USBH_BA+0x028	R/W	Host Controller Bulk Head ED Register	0x0000_0000
HcBulkCurrentED	USBH_BA+0x02C	R/W	Host Controller Bulk Current ED Register	0x0000_0000
HcDoneHead	USBH_BA+0x030	R/W	Host Controller Done Head Register	0x0000_0000
HcFmInterval	USBH_BA+0x034	R/W	Host Controller Frame Interval Register	0x0000_2EDF
HcFmRemaining	USBH_BA+0x038	R	Host Controller Frame Remaining Register	0x0000_0000
HcFmNumber	USBH_BA+0x03C	R	Host Controller Frame Number Register	0x0000_0000
HcPeriodicStart	USBH_BA+0x040	R/W	Host Controller Periodic Start Register	0x0000_0000
HcLSThreshold	USBH_BA+0x044	R/W	Host Controller Low-speed Threshold Register	0x0000_0628
HcRhDescriptorA	USBH_BA+0x048	R/W	Host Controller Root Hub Descriptor A Register	0x0000_0908
HcRhDescriptorB	USBH_BA+0x04C	R/W	Host Controller Root Hub Descriptor B Register	0x0000_0000
HcRhStatus	USBH_BA+0x050	R/W	Host Controller Root Hub Status Register	0x0000_0000
HcRhPortStatus0	USBH_BA+0x054	R/W	Host Controller Root Hub Port Status [0]	0x0000_0000
HcRhPortStatus1	USBH_BA+0x058	R/W	Host Controller Root Hub Port Status [1]	0x0000_0000
HcRhPortStatus2	USBH_BA+0x05C	R/W	Host Controller Root Hub Port Status [2]	0x0000_0000
HcRhPortStatus3	USBH_BA+0x060	R/W	Host Controller Root Hub Port Status [3]	0x0000_0000

Register	Offset	R/W	Description	Reset Value
HcRhPortStatus4	USBH_BA+0x064	R/W	Host Controller Root Hub Port Status [4]	0x0000_0000
HcRhPortStatus5	USBH_BA+0x068	R/W	Host Controller Root Hub Port Status [5]	0x0000_0000
HcRhPortStatus6	USBH_BA+0x06C	R/W	Host Controller Root Hub Port Status [6]	0x0000_0000
HcRhPortStatus7	USBH_BA+0x070	R/W	Host Controller Root Hub Port Status [7]	0x0000_0000
HcPhyControl	USBH_BA+0x200	R/W	Host Controller PHY Control Register	0x0000_0000
HcMiscControl	USBH_BA+0x204	R/W	Host Controller Miscellaneous Control Register	0x0000_0000
EHCVNR	HSUSBH_BA+0x000	R	EHCI Version Number Register	0x0095_0020
EHCSPR	HSUSBH_BA+0x004	R	EHCI Structural Parameters Register	0x0000_0012
EHCCPR	HSUSBH_BA+0x008	R	EHCI Capability Parameters Register	0x0000_0000
UCMDR	HSUSBH_BA+0x020	R/W	USB Command Register	0x0008_0000
USTSR	HSUSBH_BA+0x024	R/W	USB Status Register	0x0000_1000
UIENR	HSUSBH_BA+0x028	R/W	USB Interrupt Enable Register	0x0000_0000
UFINDR	HSUSBH_BA+0x02C	R/W	USB Frame Index Register	0x0000_0000
UPFLBAR	HSUSBH_BA+0x034	R/W	USB Periodic Frame List Base Address Register	0x0000_0000
UCALAR	HSUSBH_BA+0x038	R/W	USB Current Asynchronous List Address Register	0x0000_0000
UASSTR	HSUSBH_BA+0x03C	R/W	USB Asynchronous Schedule Sleep Timer Register	0x0000_0BD6
UCFGR	HSUSBH_BA+0x060	R/W	USB Configure Flag Register	0x0000_0000
UPSCR0	HSUSBH_BA+0x064	R/W	USB Port 0 Status and Control Register	0x0000_2000
UPSCR1	HSUSBH_BA+0x068	R/W	USB Port 1 Status and Control Register	0x0000_2000
USBPCR0	HSUSBH_BA+0x0C4	R/W	USB PHY 0 Control Register	0x0000_0060
USBPCR1	HSUSBH_BA+0x0C8	R/W	USB PHY 1 Control Register	0x0000_0020

6.22.7 Register Description

Host Controller Revision Register (HcRevision)

Register	Offset	R/W	Description	Reset Value
HcRevision	USBH_BA+0x000	R	Host Controller Revision 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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
REV							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**REV**	**Revision Number** Indicates the Open HCI Specification revision number implemented by the Hardware. Host Controller supports 1.1 specification. (X.Y = XYh).

Host Controller Control Register (HcControl)

Register	Offset	R/W	Description	Reset Value
HcControl	USBH_BA+0x004	R/W	Host Controller 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**
HCFS		**BLE**	**CLE**	**IE**	**PLE**	**CBSR**	

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:6]	**HCFS**	**Host Controller Functional State** This field sets the Host Controller state. The Controller may force a state change from USBSUSPEND to USBRESUME after detecting resume signaling from a downstream port. States are: 00 = USBRESET. 01 = USBRESUME. 10 = USBOPERATIONAL. 11 = USBSUSPEND.
[5]	**BLE**	**Bulk List Enable Bit** 0 = Processing of the Bulk list after next SOF (Start-Of-Frame) Disabled. 1 = Processing of the Bulk list in the next frame Enabled.
[4]	**CLE**	**Control List Enable Bit** 0 = Processing of the Control list after next SOF (Start-Of-Frame) Disabled. 1 = Processing of the Control list in the next frame Enabled.
[3]	**IE**	**Isochronous List Enable Bit** Both IE and PLE (HcControl[2]) high enables Host Controller to process the Isochronous list. Either IE or PLE (HcControl[2]) is low disables Host Controller to process the Isochronous list. 0 = Processing of the Isochronous list after next SOF (Start-Of-Frame) Disabled. 1 = Processing of the Isochronous list in the next frame Enabled, if the PLE (HcControl[2]) is high, too.

NUC980 SERIES TECHNICAL REFERENCE MANUAL

[2]	**PLE**	**Periodic List Enable Bit** When set, this bit enables processing of the Periodic (interrupt and isochronous) list. The Host Controller checks this bit prior to attempting any periodic transfers in a frame. 0 = Processing of the Periodic (Interrupt and Isochronous) list after next SOF (Start-Of-Frame) Disabled. 1 = Processing of the Periodic (Interrupt and Isochronous) list in the next frame Enabled. **Note:** To enable the processing of the Isochronous list, user has to set both PLE and IE (HcControl[3]) high.
[1:0]	**CBSR**	**Control Bulk Service Ratio** This specifies the service ratio between Control and Bulk EDs. Before processing any of the non-periodic lists, HC must compare the ratio specified with its internal count on how many nonempty Control EDs have been processed, in determining whether to continue serving another Control ED or switching to Bulk EDs. The internal count will be retained when crossing the frame boundary. In case of reset, HCD is responsible for restoring this value. 00 = Number of Control EDs over Bulk EDs served is 1:1. 01 = Number of Control EDs over Bulk EDs served is 2:1. 10 = Number of Control EDs over Bulk EDs served is 3:1. 11 = Number of Control EDs over Bulk EDs served is 4:1.

Host Controller Command Status Register (HcCommandStatus)

Register	Offset	R/W	Description	Reset Value
HcCommandStatus	USBH_BA+0x008	R/W	Host Controller Command Status Register	0x0000_0000

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

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17:16]	**SOC**	**Schedule Overrun Count** These bits are incremented on each scheduling overrun error. It is initialized to 00b and wraps around at 11b. This will be incremented when a scheduling overrun is detected even if SO (HcInterruptStatus[0]) has already been set.
[15:3]	**Reserved**	Reserved.
[2]	**BLF**	**Bulk List Filled** Set high to indicate there is an active TD on the Bulk list. This bit may be set by either software or the Host Controller and cleared by the Host Controller each time it begins processing the head of the Bulk list. 0 = No active TD found or Host Controller begins to process the head of the Bulk list. 1 = An active TD added or found on the Bulk list.
[1]	**CLF**	**Control List Filled** Set high to indicate there is an active TD on the Control List. It may be set by either software or the Host Controller and cleared by the Host Controller each time it begins processing the head of the Control List. 0 = No active TD found or Host Controller begins to process the head of the Control list. 1 = An active TD added or found on the Control list.
[0]	**HCR**	**Host Controller Reset** This bit is set to initiate the software reset of Host Controller. This bit is cleared by the Host Controller, upon completed of the reset operation. This bit, when set, didn't reset the Root Hub and no subsequent reset signaling be asserted to its downstream ports. 0 = Host Controller is not in software reset state. 1 = Host Controller is in software reset state.

Host Controller Interrupt Status Register (HcInterruptStatus)

Register	Offset	R/W	Description	Reset Value
HcInterruptStatus	USBH_BA+0x00C	R/W	Host Controller 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	**RHSC**	**FNO**	**Reserved**	**RD**	**SF**	**WDH**	**SO**

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**RHSC**	**Root Hub Status Change** This bit is set when the content of HcRhStatus or the content of HcRhPortStatus1 register has changed. 0 = The content of HcRhStatus and the content of HcRhPortStatus1 register didn’t change. 1 = The content of HcRhStatus or the content of HcRhPortStatus1 register has changed.
[5]	**FNO**	**Frame Number Overflow** This bit is set when bit 15 of Frame Number changes from 1 to 0 or from 0 to 1. 0 = The bit 15 of Frame Number didn’t change. 1 = The bit 15 of Frame Number changes from 1 to 0 or from 0 to 1.
[4]	**Reserved**	Reserved.
[3]	**RD**	**Resume Detected** Set when Host Controller detects resume signaling on a downstream port. 0 = No resume signaling detected on a downstream port. 1 = Resume signaling detected on a downstream port.
[2]	**SF**	**Start of Frame** Set when the Frame Management functional block signals a ‘Start of Frame’ event. Host Control generates a SOF token at the same time. 0 = Not the start of a frame. 1 = Indicate the start of a frame and Host Controller generates a SOF token.
[1]	**WDH**	**Write Back Done Head** Set after the Host Controller has written HcDoneHead to HccaDoneHead. Further updates of the HccaDoneHead will not occur until this bit has been cleared. 0 = Host Controller didn’t update HccaDoneHead. 1 = Host Controller has written HcDoneHead to HccaDoneHead.

[0]	**SO**	**Scheduling Overrun** Set when the List Processor determines a Schedule Overrun has occurred. 0 = Schedule Overrun didn’t occur. 1 = Schedule Overrun has occurred.

Host Controller Interrupt Enable Register (HcInterruptEnable)

Register	Offset	R/W	Description	Reset Value
HcInterruptEnable	USBH_BA+0x010	R/W	Host Controller Interrupt Enable Register	0x0000_0000

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

Bits	Description	
[31]	**MIE**	**Master Interrupt Enable Bit** This bit is a global interrupt enable. A write of '1' allows interrupts to be enabled via the specific enable bits listed above. Write Operation: 0 = No effect. 1 = Allow RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) or SO (HcInterruptStatus[0]) interrupts to be enabled via corresponding enable bits. Read Operation: 0 = Interrupt generation due to RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) or SO (HcInterruptStatus[0]) Disabled even if the corresponding bit in HcInterruptEnable is high. 1 = Interrupt generation due to RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) or SO (HcInterruptStatus[0]) Enabled if the corresponding bit in HcInterruptEnable is high.
[30:7]	**Reserved**	Reserved.
[6]	**RHSC**	**Root Hub Status Change Enable Bit** Write Operation: 0 = No effect. 1 = Enable RHSC (HcInterruptStatus[6]) interrupt. Read Operation: 0 = Interrupt generation due to RHSC (HcInterruptStatus[6]) Disabled. 1 = Interrupt generation due to RHSC (HcInterruptStatus[6]) Enabled.

[5]	**FNO**	**Frame Number Overflow Enable Bit** Write Operation: 0 = No effect. 1 = Enable FNO (HcInterruptStatus[5]) interrupt. Read Operation: 0 = Interrupt generation due to FNO (HcInterruptStatus[5]) Disabled. 1 = Interrupt generation due to FNO (HcInterruptStatus[5]) Enabled.
[4]	**Reserved**	Reserved.
[3]	**RD**	**Resume Detected Enable Bit** Write Operation: 0 = No effect. 1 = Enable RD (HcInterruptStatus[3]) interrupt. Read Operation: 0 = Interrupt generation due to RD (HcInterruptStatus[3]) Disabled. 1 = Interrupt generation due to RD (HcInterruptStatus[3]) Enabled.
[2]	**SF**	**Start of Frame Enable Bit** Write Operation: 0 = No effect. 1 = Enable SF (HcInterruptStatus[2]) interrupt. Read Operation: 0 = Interrupt generation due to SF (HcInterruptStatus[2]) Disabled. 1 = Interrupt generation due to SF (HcInterruptStatus[2]) Enabled.
[1]	**WDH**	**Write Back Done Head Enable Bit** Write Operation: 0 = No effect. 1 = Enable WDH (HcInterruptStatus[1]) interrupt. Read Operation: 0 = Interrupt generation due to WDH (HcInterruptStatus[1]) Disabled. 1 = Interrupt generation due to WDH (HcInterruptStatus[1]) Enabled.
[0]	**SO**	**Scheduling Overrun Enable Bit** Write Operation: 0 = No effect. 1 = Enable SO (HcInterruptStatus[0]) interrupt. Read Operation: 0 = Interrupt generation due to SO (HcInterruptStatus[0]) Disabled. 1 = Interrupt generation due to SO (HcInterruptStatus[0]) Enabled.

Host Controller Interrupt Disable Register (HcInterruptDisable)

Register	Offset	R/W	Description	Reset Value
HcInterruptDisable	USBH_BA+0x014	R/W	Host Controller Interrupt Disable Register	0x0000_0000

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

Bits	Description	
[31]	**MIE**	**Master Interrupt Disable Bit** Global interrupt disable. Writing '1' to disable all interrupts. Write Operation: 0 = No effect. 1 = Disable RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) and SO (HcInterruptStatus[0]) interrupts. Read Operation: 0 = Interrupt generation due to RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) or SO (HcInterruptStatus[0]) Disabled even if the corresponding bit in HcInterruptEnable is high. 1 = Interrupt generation due to RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) or SO (HcInterruptStatus[0]) Enabled if the corresponding bit in HcInterruptEnable is high.
[30:7]	**Reserved**	Reserved.
[6]	**RHSC**	**Root Hub Status Change Disable Bit** Write Operation: 0 = No effect. 1 = Disable RHSC (HcInterruptStatus[6]) interrupt. Read Operation: 0 = Interrupt generation due to RHSC (HcInterruptStatus[6]) Disabled. 1 = Interrupt generation due to RHSC (HcInterruptStatus[6]) Enabled.

[5]	**FNO**	**Frame Number Overflow Disable Bit** Write Operation: 0 = No effect. 1 = Disable FNO (HcInterruptStatus[5]) interrupt. Read Operation: 0 = Interrupt generation due to FNO (HcInterruptStatus[5]) Disabled. 1 = Interrupt generation due to FNO (HcInterruptStatus[5]) Enabled.
[4]	**Reserved**	Reserved.
[3]	**RD**	**Resume Detected Disable Bit** Write Operation: 0 = No effect. 1 = Disable RD (HcInterruptStatus[3]) interrupt. Read Operation: 0 = Interrupt generation due to RD (HcInterruptStatus[3]) Disabled. 1 = Interrupt generation due to RD (HcInterruptStatus[3]) Enabled.
[2]	**SF**	**Start of Frame Disable Bit** Write Operation: 0 = No effect. 1 = Disable SF (HcInterruptStatus[2]) interrupt. Read Operation: 0 = Interrupt generation due to SF (HcInterruptStatus[2]) Disabled. 1 = Interrupt generation due to SF (HcInterruptStatus[2]) Enabled.
[1]	**WDH**	**Write Back Done Head Disable Bit** Write Operation: 0 = No effect. 1 = Disable WDH (HcInterruptStatus[1]) interrupt. Read Operation: 0 = Interrupt generation due to WDH (HcInterruptStatus[1]) Disabled. 1 = Interrupt generation due to WDH (HcInterruptStatus[1]) Enabled.
[0]	**SO**	**Scheduling Overrun Disable Bit** Write Operation: 0 = No effect. 1 = Disable SO (HcInterruptStatus[0]) interrupt. Read Operation: 0 = Interrupt generation due to SO (HcInterruptStatus[0]) Disabled. 1 = Interrupt generation due to SO (HcInterruptStatus[0]) Enabled.

Host Controller Communication Area Register (HcHCCA)

Register	Offset	R/W	Description	Reset Value
HcHCCA	USBH_BA+0x018	R/W	Host Controller Communication Area Register	0x0000_0000

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

Bits	Description	
[31:8]	**HCCA**	**Host Controller Communication Area** Pointer to indicate base address of the Host Controller Communication Area (HCCA).
[7:0]	**Reserved**	Reserved.

Host Controller Period Current ED Register (HcPeriodCurrentED)

Register	Offset	R/W	Description	Reset Value
HcPeriodCurrentED	USBH_BA+0x01C	R/W	Host Controller Period Current ED Register	0x0000_0000

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

Bits	Description	
[31:4]	**PCED**	**Periodic Current ED** Pointer to indicate physical address of the current Isochronous or Interrupt Endpoint Descriptor.
[3:0]	**Reserved**	Reserved.

Host Controller Control Head ED Register (HcControlHeadED)

Register	Offset	R/W	Description	Reset Value
HcControlHeadED	USBH_BA+0x020	R/W	Host Controller Control Head ED Register	0x0000_0000

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

Bits	Description	
[31:4]	**CHED**	**Control Head ED** Pointer to indicate physical address of the first Endpoint Descriptor of the Control list.
[3:0]	**Reserved**	Reserved.

Host Controller Control Current ED Register (HcControlCurrentED)

Register	Offset	R/W	Description	Reset Value
HcControlCurrentED	USBH_BA+0x024	R/W	Host Controller Control Current ED Register	0x0000_0000

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

Bits	Description	
[31:4]	**CCED**	**Control Current Head ED** Pointer to indicate the physical address of the current Endpoint Descriptor of the Control list.
[3:0]	**Reserved**	Reserved.

Host Controller Bulk Head ED Register (HcBulkHeadED)

Register	Offset	R/W	Description	Reset Value
HcBulkHeadED	USBH_BA+0x028	R/W	Host Controller Bulk Head ED Register	0x0000_0000

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

Bits	Description	
[31:4]	**BHED**	**Bulk Head ED** Pointer to indicate the physical address of the first Endpoint Descriptor of the Bulk list.
[3:0]	**Reserved**	Reserved.

Host Controller Bulk Current Head ED Register (HcBulkCurrentED)

Register	Offset	R/W	Description	Reset Value
HcBulkCurrentED	USBH_BA+0x02C	R/W	Host Controller Bulk Current ED Register	0x0000_0000

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

Bits	Description	
[31:4]	**BCED**	**Bulk Current Head ED** Pointer to indicate the physical address of the current endpoint of the Bulk list.
[3:0]	**Reserved**	Reserved.

Host Controller Done Head Register (HcDoneHead)

Register	Offset	R/W	Description	Reset Value
HcDoneHead	USBH_BA+0x030	R/W	Host Controller Done Head Register	0x0000_0000

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

Bits	Description	
[31:4]	**DH**	**Done Head** Pointer to indicate the physical address of the last completed Transfer Descriptor that was added to the Done queue.
[3:0]	**Reserved**	Reserved.

Host Controller Frame Interval Register (HcFmInterval)

Register	Offset	R/W	Description	Reset Value
HcFmInterval	USBH_BA+0x034	R/W	Host Controller Frame Interval Register	0x0000_2EDF

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

Bits	Description	
[31]	**FIT**	**Frame Interval Toggle** This bit is toggled by Host Controller Driver when it loads a new value into FI (HcFmInterval[13:0]). 0 = Host Controller Driver didn't load new value into FI (HcFmInterval[13:0]). 1 = Host Controller Driver loads a new value into FI (HcFmInterval[13:0]).
[30:16]	**FSMPS**	**FS Largest Data Packet** This field specifies a value that is loaded into the Largest Data Packet Counter at the beginning of each frame.
[15:14]	**Reserved**	Reserved.
[13:0]	**FI**	**Frame Interval** This field specifies the length of a frame as (bit times - 1). For 12,000 bit times in a frame, a value of 11,999 is stored here.

Host Controller Frame Remaining Register (HcFmRemaining)

Register	Offset	R/W	Description	Reset Value
HcFmRemaining	USBH_BA+0x038	R	Host Controller Frame Remaining Register	0x0000_0000

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

Bits	Description	
[31]	**FRT**	**Frame Remaining Toggle** This bit is loaded from the FIT (HcFmInterval[31]) whenever FR (HcFmRemaining[13:0]) reaches 0.
[30:14]	**Reserved**	Reserved.
[13:0]	**FR**	**Frame Remaining** When the Host Controller is in the USBOPERATIONAL state, this 14-bit field decrements each 12 MHz clock period. When the count reaches 0, (end of frame) the counter reloads with Frame Interval. In addition, the counter loads when the Host Controller transitions into USBOPERATIONAL.

NUC980 SERIES TECHNICAL REFERENCE MANUAL

Host Controller Frame Number Register (HcFmNumber)

Register	Offset	R/W	Description	Reset Value
HcFmNumber	USBH_BA+0x03C	R	Host Controller Frame 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**
FN							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FN							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**FN**	**Frame Number** This 16-bit incrementing counter field is incremented coincident with the re-load of FR (HcFmRemaining[13:0]). The count rolls over from 'FFFFh' to '0h.'

Host Controller Periodic Start Register (HcPeriodicStart)

Register	Offset	R/W	Description	Reset Value
HcPeriodicStart	USBH_BA+0x040	R/W	Host Controller Periodic Start Register	0x0000_0000

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

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13:0]	**PS**	**Periodic Start** This field contains a value used by the List Processor to determine where in a frame the Periodic List processing must begin.

Host Controller Low-speed Threshold Register (HcLSThreshold)

Register	Offset	R/W	Description	Reset Value
HcLSThreshold	USBH_BA+0x044	R/W	Host Controller Low-speed Threshold Register	0x0000_0628

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

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:0]	**LST**	**Low-speed Threshold** This field contains a value which is compared to the FR (HcFmRemaining[13:0]) field prior to initiating a Low-speed transaction. The transaction is started only if FR (HcFmRemaining[13:0]) >= this field. The value is calculated by Host Controller Driver with the consideration of transmission and setup overhead.

Host Controller Root Hub Descriptor A Register (HcRhDescriptorA)

Register	Offset	R/W	Description	Reset Value
HcRhDescriptorA	USBH_BA+0x048	R/W	Host Controller Root Hub Descriptor A Register	0x0000_0908

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

Bits	Description	
[31:13]	**Reserved**	Reserved.
[12]	**NOCP**	**No over Current Protection** This bit describes how the over current status for the Root Hub ports reported. 0 = Over current status is reported. 1 = Over current status is not reported.
[11]	**OCPM**	**over Current Protection Mode** This bit describes how the over current status for the Root Hub ports reported. This bit is only valid when NOCP (HcRhDescriptorA[12]) is cleared. 0 = Global Over current. 1 = Individual Over current.
[10:9]	**Reserved**	Reserved.
[8]	**PSM**	**Power Switching Mode** This bit is used to specify how the power switching of the Root Hub ports is controlled. 0 = Global Switching. 1 = Individual Switching.
[7:0]	**NDP**	**Number Downstream Ports** USB host control supports two downstream ports and only one port is available in this series of chip.

Host Controller Root Hub Descriptor B Register (HcRhDescriptorB)

Register	Offset	R/W	Description	Reset Value
HcRhDescriptorB	USBH_BA+0x04C	R/W	Host Controller Root Hub Descriptor B Register	0x0000_0000

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

Bits	Description	
[31:16]	**PPCM**	**Port Power Control Mask** Global power switching. This field is only valid if PowerSwitchingMode is set (individual port switching). When set, the port only responds to individual port power switching commands (Set/ClearPortPower). When cleared, the port only responds to global power switching commands (Set/ClearGlobalPower). 0 = Port power controlled by global power switching. 1 = Port power controlled by port power switching. **Note:** PPCM[15:2] and PPCM[0] are reserved.
[15:0]	**Reserved**	Reserved.

Host Controller Root Hub Status Register (HcRhStatus)

Register	Offset	R/W	Description	Reset Value
HcRhStatus	USBH_BA+0x050	R/W	Host Controller Root Hub Status Register	0x0000_0000

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

Bits	Description	
[31]	CRWE	**Clear Remote Wake-up Enable Bit** This bit is used to clear DRWE (HcRhStatus[15]). This bit always read as zero. Write Operation: 0 = No effect. 1 = Clear DRWE (HcRhStatus[15]).
[31:18]	Reserved	Reserved.
[17]	OCIC	**over Current Indicator Change** This bit is set by hardware when a change has occurred in OCI (HcRhStatus[1]). Write 1 to clear this bit to zero. 0 = OCI (HcRhStatus[1]) didn’t change. 1 = OCI (HcRhStatus[1]) change.
[16]	LPSC	**Set Global Power** In global power mode (PSM (HcRhDescriptorA[8]) = 0), this bit is written to one to enable power to all ports. This bit always read as zero. Write Operation: 0 = No effect. 1 = Set global power.

[15]	**DRWE**	**Device Remote Wakeup Enable Bit** This bit controls if port's Connect Status Change as a remote wake-up event. Write Operation: 0 = No effect. 1 = Connect Status Change as a remote wake-up event Enabled. Read Operation: 0 = Connect Status Change as a remote wake-up event Disabled. 1 = Connect Status Change as a remote wake-up event Enabled.
[14:2]	**Reserved**	Reserved.
[1]	**OCI**	**over Current Indicator** This bit reflects the state of the over current status pin. This field is only valid if NOCP (HcRhDesA[12]) and OCPM (HcRhDesA[11]) are cleared. 0 = No over current condition. 1 = Over current condition.
[0]	**LPS**	**Clear Global Power** In global power mode (PSM (HcRhDescriptorA[8]) = 0), this bit is written to one to clear all ports’ power. This bit always read as zero. Write Operation: 0 = No effect. 1 = Clear global power.

Host Controller Root Hub Port Status (HcRhPortStatus [7:0])

Register	Offset	R/W	Description	Reset Value
HcRhPortStatus0	USBH_BA+0x054	R/W	Host Controller Root Hub Port Status [0]	0x0000_0000
HcRhPortStatus1	USBH_BA+0x058	R/W	Host Controller Root Hub Port Status [1]	0x0000_0000
HcRhPortStatus2	USBH_BA+0x05C	R/W	Host Controller Root Hub Port Status [2]	0x0000_0000
HcRhPortStatus3	USBH_BA+0x060	R/W	Host Controller Root Hub Port Status [3]	0x0000_0000
HcRhPortStatus4	USBH_BA+0x064	R/W	Host Controller Root Hub Port Status [4]	0x0000_0000
HcRhPortStatus5	USBH_BA+0x068	R/W	Host Controller Root Hub Port Status [5]	0x0000_0000
HcRhPortStatus6	USBH_BA+0x06C	R/W	Host Controller Root Hub Port Status [6]	0x0000_0000
HcRhPortStatus7	USBH_BA+0x070	R/W	Host Controller Root Hub Port Status [7]	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved			**PRSC**	**OCIC**	**PSSC**	**PESC**	**CSC**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**LSDA**	**PPS**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**PRS**	**POCI**	**PSS**	**PES**	**CCS**

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20]	**PRSC**	**Port Reset Status Change** This bit indicates that the port reset signal has completed. Write 1 to clear this bit to zero. 0 = Port reset is not complete. 1 = Port reset is complete.
[19]	**OCIC**	**Port over Current Indicator Change** This bit is set when POCI (HcRhPortStatus1[3]) changes. Write 1 to clear this bit to zero. 0 = POCI (HcRhPortStatus1[3]) didn’t change. 1 = POCI (HcRhPortStatus1[3]) changes.

[18]	**PSSC**	**Port Suspend Status Change** This bit indicates the completion of the selective resume sequence for the port. Write 1 to clear this bit to zero. 0 = Port resume is not completed. 1 = Port resume completed.
[17]	**PESC**	**Port Enable Status Change** This bit indicates that the port has been disabled (PES (HcRhPortStatus1[1]) cleared) due to a hardware event. Write 1 to clear this bit to zero. 0 = PES (HcRhPortStatus1[1]) didn't change. 1 = PES (HcRhPortStatus1[1]) changed.
[16]	**CSC**	**Connect Status Change** This bit indicates connect or disconnect event has been detected (CCS (HcRhPortStatus1[0]) changed). Write 1 to clear this bit to zero. 0 = No connect/disconnect event (CCS (HcRhPortStatus1[0]) didn't change). 1 = Hardware detection of connect/disconnect event (CCS (HcRhPortStatus1[0]) changed).
[15:10]	**Reserved**	Reserved.
[9]	**LSDA**	**Low Speed Device Attached (Read) or Clear Port Power (Write)** This bit defines the speed (and bud idle) of the attached device. It is only valid when CCS (HcRhPortStatus1[0]) is set. This bit is also used to clear port power. Write Operation: 0 = No effect. 1 = Clear PPS (HcRhPortStatus1[8]). Read Operation: 0 = Full Speed device. 1 = Low-speed device.
[8]	**PPS**	**Port Power Status** This bit reflects the power state of the port regardless of the power switching mode. Write Operation: 0 = No effect. 1 = Port Power Enabled. Read Operation: 0 = Port power is Disabled. 1 = Port power is Enabled.
[7:5]	**Reserved**	Reserved.
[4]	**PRS**	**Port Reset Status** This bit reflects the reset state of the port. Write Operation: 0 = No effect. 1 = Set port reset. Read Operation 0 = Port reset signal is not active. 1 = Port reset signal is active.

[3]	**POCI**	**Port over Current Indicator (Read) or Clear Port Suspend (Write)** This bit reflects the state of the over current status pin dedicated to this port. This field is only valid if NOCP (HcRhDescriptorA[12]) is cleared and OCPM (HcRhDescriptorA[11]) is set. This bit is also used to initiate the selective result sequence for the port. Write Operation: 0 = No effect. 1 = Clear port suspend. Read Operation: 0 = No over current condition. 1 = Over current condition.
[2]	**PSS**	**Port Suspend Status** This bit indicates the port is suspended Write Operation: 0 = No effect. 1 = Set port suspend. Read Operation: 0 = Port is not suspended. 1 = Port is selectively suspended.
[1]	**PES**	**Port Enable Status** Write Operation: 0 = No effect. 1 = Set port enable. Read Operation: 0 = Port Disabled. 1 = Port Enabled.
[0]	**CCS**	**CurrentConnectStatus (Read) or ClearPortEnable Bit (Write)** Write Operation: 0 = No effect. 1 = Clear port enable. Read Operation: 0 = No device connected. 1 = Device connected.

Host Controller PHY Control Register (HcPhyControl)

Register	Offset	R/W	Description	Reset Value
HcPhyControl	USBH_BA+0x200	R/W	Host Controller PHY Control Register	0x0000_0000

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

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27]	**STBYEN**	**USB Transceiver Standby Enable Bit** This bit controls if USB transceiver could enter the standby mode to reduce power consumption. 0 = The USB transceiver would never enter the standby mode. 1 = The USB transceiver will enter standby mode while port is in power off state (port power is inactive).
[26:0]	**Reserved**	Reserved.

NUC980 SERIES TECHNICAL REFERENCE MANUAL

Host Controller Miscellaneous Control Register (HcMiscControl)

Register	Offset	R/W	Description	Reset Value
HcMiscControl	USBH_BA+0x204	R/W	Host Controller Miscellaneous Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							**DPRT1**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**OCAL**	**Reserved**	**ABORT**	**Reserved**

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**DPRT1**	**Disable Port 1** This bit controls if the connection between USB host controller and transceiver of port 1 is disabled. If the connection is disabled, the USB host controller will not recognize any event of USB bus. Set this bit high, the transceiver of port 1 will also be forced into the standby mode no matter what USB host controller operation is. 0 = The connection between USB host controller and transceiver of port 1 Enabled. 1 = The connection between USB host controller and transceiver of port 1 Disabled and the transceiver of port 1 will also be forced into the standby mode.
[15:4]	**Reserved**	Reserved.
[3]	**OCAL**	**over Current Active Low** This bit controls the polarity of over current flag from external power IC. 0 = Over current flag is high active. 1 = Over current flag is low active.
[2]	**Reserved**	Reserved.
[1]	**ABORT**	**AHB Bus ERROR Response** This bit indicates there is an ERROR response received in AHB bus. 0 = No ERROR response received. 1 = ERROR response received.
[0]	**Reserved**	Reserved.

EHCI Version Number Register (EHCVNR)

Register	Offset	R/W	Description	Reset Value
EHCVNR	HSUSBH_BA+0x000	R	EHCI Version Number Register	0x0095_0020

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

Bits	Description	
[31:16]	**VERSION**	**Host Controller Interface Version Number** This is a two-byte register containing a BCD encoding of the EHCI revision number supported by this host controller. The most significant byte of this register represents a major revision and the least significant byte is the minor revision.
[15:8]	**Reserved**	Reserved.
[7:0]	**CRLEN**	**Capability Registers Length** This register is used as an offset to add to register base to find the beginning of the Operational Register Space.

EHCI Structural Parameters Register (EHCSPR)

Register	Offset	R/W	Description	Reset Value
EHCSPR	HSUSBH_BA+0x004	R	EHCI Structural Parameters Register	0x0000_0012

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:12]	**N_CC**	**Number of Companion Controller** This field indicates the number of companion controllers associated with this USB 2.0 host controller. A zero in this field indicates there are no companion host controllers. Port-ownership hand-off is not supported. Only high-speed devices are supported on the host controller root ports. A value larger than zero in this field indicates there are companion USB 1.1 host controller(s). Port-ownership hand-offs are supported. High, Full- and Low-speed devices are supported on the host controller root ports.
[11:8]	**N_PCC**	**Number of Ports Per Companion Controller** This field indicates the number of ports supported per companion host controller. It is used to indicate the port routing configuration to system software. For example, if N_PORTS has a value of 6 and N_CC has a value of 2 then N_PCC could have a value of 3. The convention is that the first N_PCC ports are assumed to be routed to companion controller 1, the next N_PCC ports to companion controller 2, etc. In the previous example, the N_PCC could have been 4, where the first 4 are routed to companion controller 1 and the last two are routed to companion controller 2. The number in this field must be consistent with N_PORTS and N_CC.
[7:5]	**Reserved**	Reserved.
[4]	**PPC**	**Port Power Control** This field indicates whether the host controller implementation includes port power control. A one in this bit indicates the ports have port power switches. A zero in this bit indicates the port do not have port power stitches. The value of this field affects the functionality of the Port Power field in each port status and control register.
[3:0]	**N_PORTS**	**Number of Physical Downstream Ports** This field specifies the number of physical downstream ports implemented on this host controller. The value of this field determines how many port registers are addressable in the Operational Register Space (see Table 2-8). Valid values are in the range of 1H to FH. A zero in this field is undefined.

EHCI Capability Parameters Register (EHCCPR)

Register	Offset	R/W	Description	Reset Value
EHCCPR	HSUSBH_BA+0x008	R	EHCI Capability Parameters Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
EECP							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IST				Reserved	ASPC	PFLF	AC64

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**EECP**	**EHCI Extended Capabilities Pointer (EECP)** 0 = No extended capabilities are implemented.
[7:4]	**IST**	**Isochronous Scheduling Threshold** This field indicates, relative to the current position of the executing host controller, where software can reliably update the isochronous schedule. When bit [7] is zero, the value of the least significant 3 bits indicates the number of micro-frames a host controller can hold a set of isochronous data structures (one or more) before flushing the state.
[3]	**Reserved**	Reserved.
[2]	**ASPC**	**Asynchronous Schedule Park Capability** 0 = This EHCI host controller doesn’t support park feature of high-speed queue heads in the Asynchronous Schedule.
[1]	**PFLF**	**Programmable Frame List Flag** 0 = System software must use a frame list length of 1024 elements with this EHCI host controller.
[0]	**AC64**	**64-bit Addressing Capability** 0 = Data structure using 32-bit address memory pointers.

USB Command Register (UCMDR)

Register	Offset	R/W	Description	Reset Value
UCMDR	HSUSBH_BA+0x020	R/W	USB Command Register	0x0008_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**ITC**	**Interrupt Threshold Control (R/W)** This field is used by system software to select the maximum rate at which the host controller will issue interrupts. The only valid values are defined below. If software writes an invalid value to this register, the results are undefined. Value Maximum Interrupt Interval 0x00 = Reserved. 0x01 = 1 micro-frame. 0x02 = 2 micro-frames. 0x04 = 4 micro-frames. 0x08 = 8 micro-frames (default, equates to 1 ms). 0x10 = 16 micro-frames (2 ms). 0x20 = 32 micro-frames (4 ms). 0x40 = 64 micro-frames (8 ms). Any other value in this register yields undefined results. Software modifications to this bit while HCHalted bit is equal to zero results in undefined behavior.
[15:7]	**Reserved**	Reserved.

[6]	**IAAD**	**Interrupt on Asynchronous Advance Doorbell (R/W)** This bit is used as a doorbell by software to tell the host controller to issue an interrupt the next time it advances asynchronous schedule. Software must write a 1 to this bit to ring the doorbell. When the host controller has evicted all appropriate cached schedule state, it sets the Interrupt on Asynchronous Advance status bit in the USTSR register. If the Interrupt on Asynchronous Advance Enable bit in the UIENR register is a one then the host controller will assert an interrupt at the next interrupt threshold. The host controller sets this bit to a zero after it has set the Interrupt on Asynchronous Advance status bit in the USTSR register to a one. Software should not write a one to this bit when the asynchronous schedule is disabled. Doing so will yield undefined results.
[5]	**ASEN**	**Asynchronous Schedule Enable (R/W)** This bit controls whether the host controller skips processing the Asynchronous Schedule. Values mean: 0 = Do not process the Asynchronous Schedule. 1 = Use the UCALAR register to access the Asynchronous Schedule.
[4]	**PSEN**	**Periodic Schedule Enable (R/W)** This bit controls whether the host controller skips processing the Periodic Schedule. Values mean: 0 = Do not process the Periodic Schedule. 1 = Use the UPFLBAR register to access the Periodic Schedule.
[3:2]	**FLSZ**	**Frame List Size (R/W or RO)** This field is R/W only if Programmable Frame List Flag in the EHCCPAR registers is set to a one. This field specifies the size of the frame list. The size the frame list controls which bits in the Frame Index Register should be used for the Frame List Current index. Values mean: 00 = 1024 elements (4096 bytes) Default value. 01 = 512 elements (2048 bytes). 10 = 256 elements (1024 bytes) – for resource-constrained environment. 11 = Reserved.
[1]	**HCRST**	**Host Controller Reset (HCRST) (R/W)** This control bit is used by software to reset the host controller. The effects of this on Root Hub registers are similar to a Chip Hardware Reset. When software writes a one to this bit, the Host Controller resets its internal pipelines, timers, counters, state machines, etc. to their initial value. Any transaction currently in progress on USB is immediately terminated. A USB reset is not driven on downstream ports. All operational registers, including port registers and port state machines are set to their initial values. Port ownership reverts to the companion host controller(s), with the side effects. Software must reinitialize the host controller in order to return the host controller to an operational state. This bit is set to zero by the Host Controller when the reset process is complete. Software cannot terminate the reset process early by writing a zero to this register. Software should not set this bit to a one when HCHalted(USTSR[12]) is a zero. Attempting to reset an actively running host controller will result in undefined behavior.

NUC980 SERIES TECHNICAL REFERENCE MANUAL

[0]	**RUN**	**Run/Stop (R/W)** When set to a 1, the Host Controller proceeds with execution of the schedule. The Host Controller continues execution as long as this bit is set to a 1. When this bit is set to 0, the Host Controller completes the current and any actively pipelined transactions on the USB and then halts. The Host Controller must halt within 16 micro-frames after software clears the Run bit. HCHalted(USTSR[12]) indicates when the Host Controller has finished its pending pipelined transactions and has entered the stopped state. Software must not write a one to this field unless the host controller is in the Halted state (i.e. HCHalted(USTSR[12]) is a one). Doing so will yield undefined results. 0 = Stop. 1 = Run.

USB Status Register (USTSR)

Register	Offset	R/W	Description	Reset Value
USTSR	HSUSBH_BA+0x024	R/W	USB Status 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**
ASS	**PSS**	**RECLA**	**HCHalted**	**Reserved**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**IAA**	**HSERR**	**FLR**	**PCD**	**UERRINT**	**USBINT**

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**ASS**	**Asynchronous Schedule Status (Read Only)** The bit reports the current real status of the Asynchronous Schedule. If this bit is a zero then the status of them Asynchronous Schedule is disabled. If this bit is a one then the status of the Asynchronous Schedule is enabled. The Host Controller is not required to immediately disable or enable the Asynchronous Schedule when software transitions ASEN(UCMDR[5]) bit. When this bit and ASEN(UCMDR[5]) are the same value, the Asynchronous Schedule is either enabled (1) or disabled (0).
[14]	**PSS**	**Periodic Schedule Status (Read Only)** The bit reports the current real status of the Periodic Schedule. If this bit is a zero then the status of the Periodic Schedule is disabled. If this bit is a one then the status of the Periodic Schedule is enabled. The Host Controller is not required to immediately disable or enable the Periodic Schedule when software transitions PSEN(UCMDR[4]) bit. When this bit and PSEN(UCMDR[4]) bit are the same value, the Periodic Schedule is either enabled (1) or disabled (0).
[13]	**RECLA**	**Reclamation (Read Only)** This is a read-only status bit, which is used to detect an empty asynchronous schedule.
[12]	**HCHalted**	**HCHalted (Read Only)** This bit is a zero whenever the Run/Stop bit is a one. The Host Controller sets this bit to one after it has stopped executing as a result of the Run/Stop bit being set to 0, either by software or by the Host Controller hardware (e.g. internal error).
[11:6]	**Reserved**	Reserved.
[5]	**IAA**	**Interrupt on Asynchronous Advance (R/WC)** System software can force the host controller to issue an interrupt the next time the host controller advances the asynchronous schedule by writing a one to the Interrupt on Asynchronous Advance Doorbell bit in the UCMDR register. This status bit indicates the assertion of that interrupt source.

[4]	**HSERR**	**Host System Error (R/WC)** The Host Controller sets this bit to 1 when a serious error occurs during a host system access involving the Host Controller module.
[3]	**FLR**	**Frame List Rollover (R/WC)** The Host Controller sets this bit to a one when the Frame List Index rolls over from its maximum value to zero. The exact value at which the rollover occurs depends on the frame list size. For example, if the frame list size (as programmed in the Frame List Size field of the UCMDR register) is 1024, the Frame Index Register rolls over every time UFINDR[13] toggles. Similarly, if the size is 512, the Host Controller sets this bit to a one every time UFINDR[12] toggles.
[2]	**PCD**	**Port Change Detect (R/WC)** The Host Controller sets this bit to a one when any port for which the Port Owner bit is set to zero has a change bit transition from a zero to a one or a Force Port Resume bit transition from a zero to a one as a result of a J-K transition detected on a suspended port. This bit will also be set as a result of the Connect Status Change being set to a one after system software has relinquished ownership of a connected port by writing a one to a port's Port Owner bit. This bit is allowed to be maintained in the Auxiliary power well. Alternatively, it is also acceptable that on a D3 to D0 transition of the EHCI HC device, this bit is loaded with the OR of all of the PORTSC change bits (including: Force port resume, over-current change, enable/disable change and connect status change).
[1]	**UERRINT**	**USB Error Interrupt (USBERRINT) (R/WC)** The Host Controller sets this bit to 1 when completion of a USB transaction results in an error condition (e.g., error counter underflow). If the TD on which the error interrupt occurred also had its IOC bit set, both this bit and USBINT bit are set.
[0]	**USBINT**	**USB Interrupt (USBINT) (R/WC)** The Host Controller sets this bit to 1 on the completion of a USB transaction, which results in the retirement of a Transfer Descriptor that had its IOC bit set. The Host Controller also sets this bit to 1 when a short packet is detected (actual number of bytes received was less than the expected number of bytes).

USB Interrupt Enable Register (UIENR)

Register	Offset	R/W	Description	Reset Value
UIENR	HSUSBH_BA+0x028	R/W	USB Interrupt Enable Register	0x0000_0000

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

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5]	**IAAEN**	**Interrupt on Asynchronous Advance Enable or Disable Bit** When this bit is a one, and the Interrupt on Asynchronous Advance bit IAA(USTSR[5]) is a one, the host controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the Interrupt on Asynchronous Advance bit. 0 = Interrupt on Asynchronous Advance Disabled. 1 = Interrupt on Asynchronous Advance Enabled.
[4]	**HSERREN**	**Host System Error Enable or Disable Bit** When this bit is a one and Host System Error bit HSERR(USTSR[4]) is a one, the host controller will issue an interrupt. The interrupt is acknowledged by software clearing the HSERR(USTSR[4]) bit. 0 = Host System Error interrupt Disabled. 1 = Host System Error interrupt Enabled.
[3]	**FLREN**	**Frame List Rollover Enable or Disable Bit** When this bit is a one, and the Frame List Rollover bit FLR(USTSR[3]) is a one, the host controller will issue an interrupt. The interrupt is acknowledged by software clearing the Frame List Rollover bit. 0 = Frame List Rollover interrupt Disabled. 1 = Frame List Rollover interrupt Enabled.
[2]	**PCIEN**	**Port Change Interrupt Enable or Disable Bit** When this bit is a one, and the Port Change Detect bit PCD(USTSR[2]) a one, the host controller will issue an interrupt. The interrupt is acknowledged by software clearing PCD(USTSR[2]) bit. 0 = Port Change interrupt Disabled. 1 = Port Change interrupt Enabled.

[1]	**UERRIEN**	**USB Error Interrupt Enable or Disable Bit** When this bit is a one, and the UERRINT(USTSR[1]) bit is a one, the host controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing UERRINT(USTSR[1]) bit. 0 = USB Error interrupt Disabled. 1 = USB Error interrupt Enabled.
[0]	**USBIEN**	**USB Interrupt Enable or Disable Bit** When this bit is a one, and the USBINT(USTSR[0]) bit is a one, the host controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing USBINT(USTSR[0]) bit. 0 = USB interrupt Disabled. 1 = USB interrupt Enabled.

USB Frame Index Register (UFINDR)

Register	Offset	R/W	Description	Reset Value
UFINDR	HSUSBH_BA+0x02C	R/W	USB Frame Index Register	0x0000_0000

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

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13:0]	**FI**	**Frame Index** The value in this register increment at the end of each time frame (e.g. micro-frame). Bits [N:3] are used for the Frame List current index. This means that each location of the frame list is accessed 8 times (frames or micro-frames) before moving to the next index. The following illustrates values of N based on the value of the Frame List Size FLSZ(UCMR[3:2]). **FLSZ (UCMDR[3:2]) Number Elements N** 0x0 1024 12 0x1 512 11 0x2 256 10 0x3 Reserved

USB Periodic Frame List Base Address Register (UPFLBAR)

Register	Offset	R/W	Description	Reset Value
UPFLBAR	HSUSBH_BA+0x034	R/W	USB Periodic Frame List Base Address Register	0x0000_0000

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

Bits	Description	
[31:12]	**BADDR**	**Base Address** These bits correspond to memory address signals [31:12], respectively.
[11:0]	**Reserved**	Reserved.

USB Current Asynchronous List Address Register (UCALAR)

Register	Offset	R/W	Description	Reset Value
UCALAR	HSUSBH_BA+0x038	R/W	USB Current Asynchronous List Address Register	0x0000_0000

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

Bits	Description	
[31:5]	**LPL**	**Link Pointer Low (LPL)** These bits correspond to memory address signals [31:5], respectively. This field may only reference a Queue Head (QH).
[4:0]	**Reserved**	Reserved.

USB Asynchronous Schedule Sleep Timer Register (UASSTR)

Register	Offset	R/W	Description	Reset Value
UASSTR	HSUSBH_BA+0x03C	R/W	USB Asynchronous Schedule Sleep Timer Register	0x0000_0BD6

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

Bits	Description	
[31:11]	**Reserved**	Reserved.
[11:0]	**ASSTMR**	**Asynchronous Schedule Sleep Timer** This field defines the AsyncSchedSleepTime of EHCI spec. The asynchronous schedule sleep timer is used to control how often the host controller fetches asynchronous schedule list from system memory while the asynchronous schedule is empty. The default value of this timer is 12'hBD6. Because this timer is implemented in UTMI clock (30 MHz) domain, the default sleeping time will be about 100us.

USB Configure Flag Register (UCFGR)

Register	Offset	R/W	Description	Reset Value
UCFGR	HSUSBH_BA+0x060	R/W	USB Configure Flag Register	0x0000_0000

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

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**CF**	**Configure Flag (CF)** Host software sets this bit as the last action in its process of configuring the Host Controller. This bit controls the default port-routing control logic. Bit values and side-effects are listed below. 0 = Port routing control logic default-routes each port to an implementation dependent classic host controller. 1 = Port routing control logic default-routes all ports to this host controller.

USB Port Status and Control Register (UPSCR)

Register	Offset	R/W	Description	Reset Value
UPSCR0	HSUSBH_BA+0x064	R/W	USB Port 0 Status and Control Register	0x0000_2000
UPSCR1	HSUSBH_BA+0x068	R/W	USB Port 1 Status and Control Register	0x0000_2000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved				PTC			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		PO	PP	LSTS		Reserved	PRST
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SUSPEND	FPR	OCC	OCA	PEC	PE	CSC	CCS

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19:16]	**PTC**	**Port Test Control (R/W)** When this field is zero, the port is NOT operating in a test mode. A non-zero value indicates that it is operating in test mode and the specific test mode is indicated by the specific value. The encoding of the test mode bits are (0x6 ~ 0xF are reserved): Bits Test Mode 0x0 = Test mode not enabled. 0x1 = Test J_STATE. 0x2 = Test K_STATE. 0x3 = Test SE0_NAK. 0x4 = Test Packet. 0x5 = Test FORCE_ENABLE.
[15:14]	**Reserved**	Reserved.
[13]	**PO**	**Port Owner (R/W)** This bit unconditionally goes to a 0b when CF(UCFGR[0]) makes a 0 to 1 transition. This bit unconditionally goes to 1 whenever the Configured bit is zero. System software uses this field to release ownership of the port to a selected host controller (in the event that the attached device is not a high-speed device). Software writes a one to this bit when the attached device is not a high-speed device. A one in this bit means that a companion host controller owns and controls the port.
[12]	**PP**	**Port Power (PP)** Host controller has port power control switches. This bit represents the Current setting of the switch (0 = off, 1 = on). When power is not available on a port (i.e. PP equals a 0), the port is nonfunctional and will not report attaches, detaches, etc. When an over-current condition is detected on a powered port and PPC is a one, the PP bit in each affected port may be transitioned by the host controller from a 1 to 0 (removing power from the port).

[11:10]	**LSTS**	**Line Status (Read Only)** These bits reflect the current logical levels of the D+ (bit 11) and D- (bit 10) signal lines. These bits are used for detection of low-speed USB devices prior to the port reset and enable sequence. This field is valid only when the port enable bit is zero and the current connect status bit is set to a one. The encoding of the bits are: Bits[11:10] USB State Interpretation 00 = SE0 Not Low-speed device, perform EHCI reset. 01 = K-state Low-speed device, release ownership of port. 10 = J-state Not Low-speed device, perform EHCI reset. 11 = Undefined Not Low-speed device, perform EHCI reset. This value of this field is undefined if Port Power is zero.
[9]	**Reserved**	Reserved.
[8]	**PRST**	**Port Reset (R/W)** When software writes a one to this bit (from a zero), the bus reset sequence as defined in the USB Specification Revision 2.0 is started. Software writes a zero to this bit to terminate the bus reset sequence. Software must keep this bit at a one long enough to ensure the reset sequence, as specified in the USB Specification Revision 2.0, completes. **Note:** when software writes this bit to a one, it must also write a zero to the Port Enable bit. Note that when software writes a zero to this bit there may be a delay before the bit status changes to a zero. The bit status will not read as a zero until after the reset has completed. If the port is in high-speed mode after reset is complete, the host controller will automatically enable this port (e.g. set the Port Enable bit to a one). A host controller must terminate the reset and stabilize the state of the port within 2 milliseconds of software transitioning this bit from a one to a zero. For example: if the port detects that the attached device is high-speed during reset, then the host controller must have the port in the enabled state within 2ms of software writing this bit to a zero. HCHalted(USTSR[12]) should be a zero before software attempts to use this bit. The host controller may hold Port Reset asserted to a one when HCHalted(USTSR[12]) is a one. This field is zero if Port Power is zero. 0 = Port is not in Reset. 1 = Port is in Reset.
[7]	**SUSPEND**	**Suspend (R/W)** Port Enabled Bit and Suspend bit of this register define the port states as follows: 00 = Port Disable. 01 = Port Disable. 10 = Port Enable. 11 = Port Suspend. When in suspend state, downstream propagation of data is blocked on this port, except for port reset. The blocking occurs at the end of the current transaction, if a transaction was in progress when this bit was written to 1. In the suspend state, the port is sensitive to resume detection. Note that the bit status does not change until the port is suspended and that there may be a delay in suspending a port if there is a transaction currently in progress on the USB. A write of zero to this bit is ignored by the host controller. The host controller will unconditionally set this bit to a zero when: Software sets the Force Port Resume bit to a zero (from a one). Software sets the Port Reset bit to a one (from a zero). If host software sets this bit to a one when the port is not enabled (i.e. Port enabled bit is a zero) the results are undefined. This field is zero if Port Power is zero. 0 = Port not in suspend state. 1 = Port in suspend state.

[6]	**FPR**	**Force Port Resume (R/W)** This functionality defined for manipulating this bit depends on the value of the Suspend bit. For example, if the port is not suspended (Suspend and Enabled bits are a one) and software transitions this bit to a one, then the effects on the bus are undefined. Software sets this bit to a 1 to drive resume signaling. The Host Controller sets this bit to a 1 if a J-to-K transition is detected while the port is in the Suspend state. When this bit transitions to a one because a J-to-K transition is detected, the Port Change Detect bit in the USBSTS register is also set to a one. If software sets this bit to a one, the host controller must not set the Port Change Detect bit. Note that when the EHCI controller owns the port, the resume sequence follows the defined sequence documented in the USB Specification Revision 2.0. The resume signaling (Full-speed 'K') is driven on the port as long as this bit remains a one. Software must appropriately time the Resume and set this bit to a zero when the appropriate amount of time has elapsed. Writing a zero (from one) causes the port to return to high-speed mode (forcing the bus below the port into a high-speed idle). This bit will remain a one until the port has switched to the high-speed idle. The host controller must complete this transition within 2 milliseconds of software setting this bit to a zero. This field is zero if Port Power is zero. 0 = No resume (K-state) detected/driven on port. 1 = Resume detected/driven on port.
[5]	**OCC**	**Over-current Change (R/WC)** 1 = This bit gets set to a one when there is a change to Over-current Active. Software clears this bit by writing a one to this bit position.
[4]	**OCA**	**Over-current Active (Read Only)** This bit will automatically transition from a one to a zero when the over current condition is removed. 0 = This port does not have an over-current condition. 1 = This port currently has an over-current condition.
[3]	**PEC**	**Port Enable/Disable Change (R/WC)** For the root hub, this bit gets set to a one only when a port is disabled due to the appropriate conditions existing at the EOF2 point (See Chapter 11 of the USB Specification for the definition of a Port Error). Software clears this bit by writing a 1 to it. This field is zero if Port Power is zero. 0 = No change. 1 = Port enabled/disabled status has changed.
[2]	**PE**	**Port Enabled/Disabled (R/W)** Ports can only be enabled by the host controller as a part of the reset and enable. Software cannot enable a port by writing a one to this field. The host controller will only set this bit to a one when the reset sequence determines that the attached device is a high-speed device. Ports can be disabled by either a fault condition (disconnect event or other fault condition) or by host software. Note that the bit status does not change until the port state actually changes. There may be a delay in disabling or enabling a port due to other host controller and bus events. When the port is disabled (0b) downstream propagation of data is blocked on this port, except for reset. This field is zero if Port Power is zero. 0 = Port Disabled. 1 = Port Enabled.

[1]	**CSC**	**Connect Status Change (R/W)** Indicates a change has occurred in the port's Current Connect Status. The host controller sets this bit for all changes to the port device connect status, even if system software has not cleared an existing connect status change. For example, the insertion status changes twice before system software has cleared the changed condition, hub hardware will be "setting" an already-set bit (i.e., the bit will remain set).Software sets this bit to 0 by writing a 1 to it. This field is zero if Port Power is zero. 0 = No change. 1 = Change in Current Connect Status.
[0]	**CCS**	**Current Connect Status (Ready Only)** This value reflects the current state of the port, and may not correspond directly to the event that caused the Connect Status Change bit (Bit 1) to be set. This field is zero if Port Power is zero. 0 = No device is present. 1 = Device is present on port.

USB PHY 0 Control Register (USBPCR0)

Register	Offset	R/W	Description	Reset Value
USBPCR0	HSUSBH_BA+0x0C4	R/W	USB PHY 0 Control Register	0x0000_0060

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

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11]	**CLKVALID**	**UTMI Clock Valid** This bit is a flag to indicate if the UTMI clock from USB 2.0 PHY is ready. S/W program must prevent to write other control registers before this UTMI clock valid flag is active. 0 = UTMI clock is not valid. 1 = UTMI clock is valid.
[10:9]	**Reserved**	Reserved.
[8]	**SUSPEND**	**Suspend Assertion** This bit controls the suspend mode of USB PHY 0. While PHY was suspended, all circuits of PHY were powered down and outputs are tri-state. This bit is 1'b0 in default. This means the USB PHY 0 is suspended in default. It is necessary to set this bit 1'b1 to make USB PHY 0 leave suspend mode before doing configuration of USB host. 0 = USB PHY 0 was suspended. 1 = USB PHY 0 was not suspended.
[7:4]	**Reserved**	Reserved.
[3]	**LPWRMOD**	**USB PHY Low Power Mode Enable Bit** This bit controls the low power mode enable of USB PHY 0. 0 = USB PHY 0 low power mode Disabled. 1 = USB PHY 0 low power mode Enabled.
[2:0]	**Reserved**	Reserved.

USB PHY 1 Control Register (USBPCR1)

Register	Offset	R/W	Description	Reset Value
USBPCR1	HSUSBH_BA+0x0C8	R/W	USB PHY 1 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							**SUSPEND**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**SUSPEND**	**Suspend Assertion** This bit controls the suspend mode of USB PHY 1. While PHY was suspended, all circuits of PHY were powered down and outputs are tri-state. This bit is 1'b0 in default. This means the USB PHY 0 is suspended in default. It is necessary to set this bit 1'b1 to make USB PHY 0 leave suspend mode before doing configuration of USB host. 0 = USB PHY 1 was suspended. 1 = USB PHY 1 was not suspended.
[7:0]	**Reserved**	Reserved.

6.23 Controller Area Network (CAN)

6.23.1 Overview

The C_CAN consists of the CAN Core, Message RAM, Message Handler, Control Registers and Module Interface. The CAN Core performs communication according to the CAN protocol version 2.0 part A and B. The bit rate can be programmed to values up to 1MBit/s. For the connection to the physical layer, additional transceiver hardware is required.

For communication on a CAN network, individual Message Objects are configured. The Message Objects and Identifier Masks for acceptance filtering of received messages are stored in the Message RAM. All functions concerning the handling of messages are implemented in the Message Handler. These functions include acceptance filtering, the transfer of messages between the CAN Core and the Message RAM, and the handling of transmission requests as well as the generation of the module interrupt.

The register set of the C_CAN can be accessed directly by the software through the module interface. These registers are used to control/configure the CAN Core and the Message Handler and to access the Message RAM.

6.23.2 Features

- Supports CAN protocol version 2.0 part A and B
- Bit rates up to 1 MBit/s
- 32 Message Objects
- Each Message Object has its own identifier mask
- Programmable FIFO mode (concatenation of Message Objects)
- Maskable interrupt
- Disabled Automatic Re-transmission mode for Time Triggered CAN applications
- Programmable loop-back mode for self-test operation
- 16-bit module interfaces to the AMBA APB bus
- Supports wake-up function

6.23.3 Block Diagram

The C_CAN interfaces with the AMBA APB bus. Figure 6.23-1 shows the block diagram of the C_CAN.

- CAN Core
  - CAN Protocol Controller and Rx/Tx Shift Register for serial/parallel conversion of messages.
- Message RAM
  - Stores Message Objects and Identifier Masks
- Registers
  - All registers used to control and to configure the C_CAN.
  - Message Handler
  - State Machine that controls the data transfer between the Rx/Tx Shift Register of the CAN Core and the Message RAM as well as the generation of interrupts as

programmed in the Control and Configuration Registers.

- Module Interface
  - C_CAN interfaces to the AMBA APB 16-bit bus from CPU.

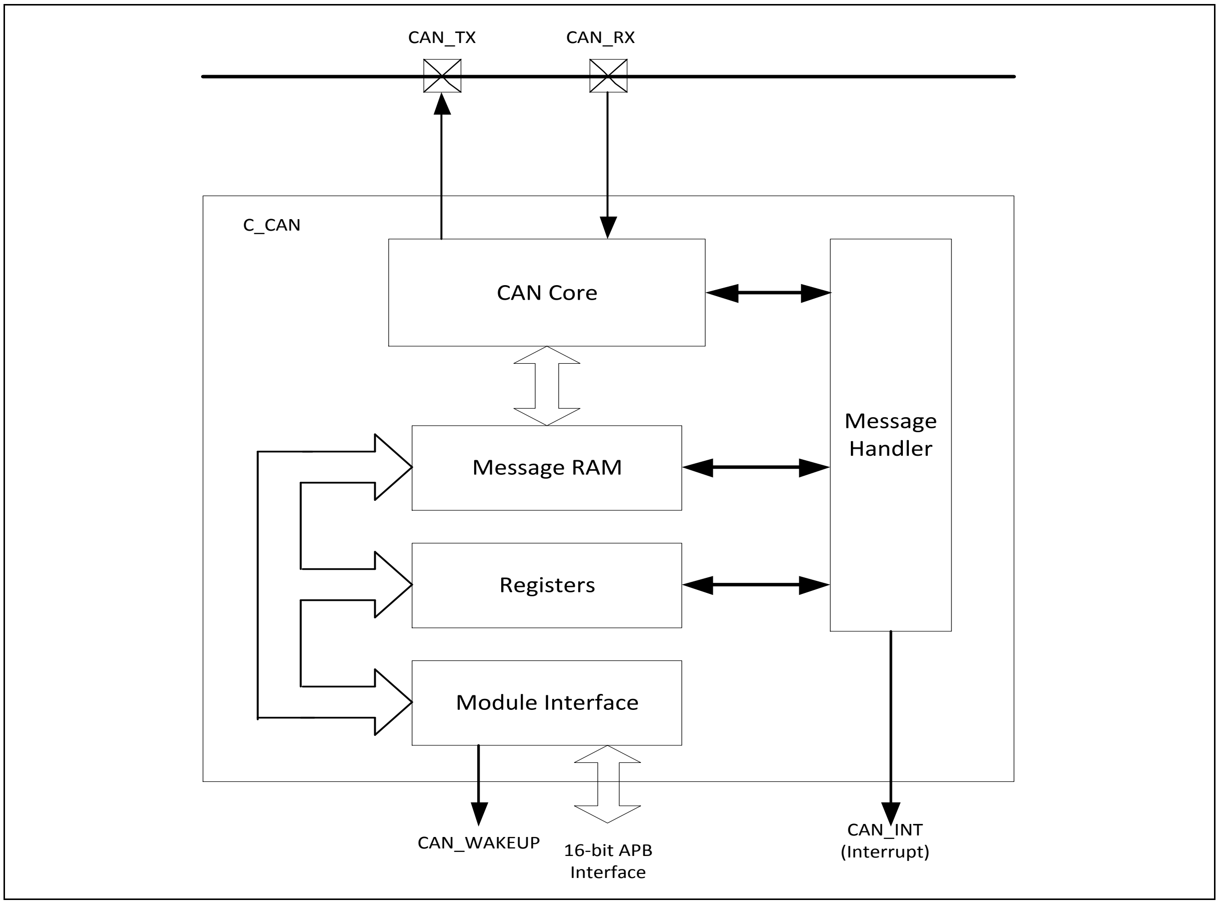


Figure 6.23-1 CAN Peripheral Block Diagram

6.23.4 Basic Configuration

6.23.4.1 CAN0 Basic Configuration

- Clock source Configuration
  - Enable CAN0 clock in CAN0CKEN (APBCLK1[8]).
- Reset Configuration
  - Reset CAN0 controller in CAN0RST (IPRSTC1[8]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
CAN0	CAN0_RXD	PE.0	MFP2
		PD.6, PG.11	MFP4
		PC.3	MFP7
	CAN0_TXD	PE.1	MFP2

		PD.7, PG.12	MFP4
		PC.4	MFP7

6.23.4.2 *CAN1 Basic Configuration*

- Clock source Configuration
  - Enable CAN1 clock in CAN1CKEN (APBCLK1[9]).
- Reset Configuration
  - Reset CAN1 controller in CAN1RST (IPRSTC1[9]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
CAN1	CAN1_RXD	PE.2	MFP2
		PD.14, PG.13	MFP4
		PA.13	MFP5
	CAN1_TXD	PE.3	MFP2
		PD.15, PG.14	MFP4
		PA.14	MFP5

6.23.4.3 *CAN2 Basic Configuration*

- Clock source Configuration
  - Enable CAN2 clock in CAN2CKEN (APBCLK1[10]).
- Reset Configuration
  - Reset CAN2 controller in CAN2RST (IPRSTC1[10]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
CAN2	CAN2_RXD	PE.4	MFP2
		PB.8	MFP3
		PB.1, PD.12	MFP4
		PA.15	MFP5
	CAN2_TXD	PE.5	MFP2
		PC.0	MFP3
		PB.3, PD.13	MFP4
		PG.10	MFP5

6.23.4.4 *CAN3 Basic Configuration*

- Clock source Configuration
  - Enable CAN3 clock in CAN3CKEN (APBCLK1[11]).

- Reset Configuration
  - Reset CAN3 controller in CAN3RST (IPRSTC1[11]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
CAN3	CAN3_RXD	PE.6, PE.10	MFP2
		PA.0	MFP7
	CAN3_TXD	PE.7, PE.12	MFP2
		PA.1	MFP7

6.23.4.5 Functional Description

6.23.4.6 Software Initialization

The software initialization is started by setting the Init bit (CAN_CON[0]), either by a software or a hardware reset, or by going to bus-off state.

While the Init bit is set, all messages transfer to and from the CAN bus are stopped and the status of the CAN_TX output pin is recessive (HIGH). The Error Management Logic (EML) counters are unchanged. Setting the Init bit does not change any configuration register.

To initialize the CAN Controller, software has to set up the Bit Timing Register and each Message Object. If a Message Object is not required, the corresponding MsgVal bit (CAN_IFn_ARB2[15]) should be cleared. Otherwise, the entire Message Object has to be initialized.

Access to the Bit Timing Register and to the Baud Rate Prescaler Extension Register for configuring bit timing is enabled when both the Init and CCE (CAN_CON[6]) bits are set.

Resetting the Init bit (by software only) finishes the software initialization. Later, the Bit Stream Processor (BSP) (see Section 6.23.6.15: Configuring the Bit Timing) synchronizes itself to the data transfer on the CAN bus by waiting for the occurrence of a sequence of 11 consecutive recessive bits (= Bus Idle) before it can take part in bus activities and start the message transfer.

The initialization of the Message Objects is independent of Init and can be done on the fly, but the Message Objects should all be configured to particular identifiers or set to not valid before the BSP starts the message transfer.

To change the configuration of a Message Object during normal operation, the software has to start by resetting the corresponding MsgVal bit. When the configuration is completed, MsgVal bit is set again.

6.23.4.7 CAN Message Transfer

Once the C_CAN is initialized and Init bit (CAN_CON[0]) is reset to zero, the C_CAN Core synchronizes itself to the CAN bus and starts the message transfer.

Received messages are stored in their appropriate Message Objects if they pass the Message Handler's acceptance filtering. The whole message including all arbitration bits, DLC (CAN_IFn_MCON[3:0]) and eight data bytes (CAN_IFn_DAT_A1/2; CAN_IFn_DAT_B1/2) are stored in the Message Object. If the Identifier Mask is used, the arbitration bits which are masked to "don't care" may be overwritten in the Message Object.

Software can read or write each message any time through the Interface Registers and the Message Handler guarantees data consistency in case of concurrent accesses.

Messages to be transmitted are updated by the application software. If a permanent Message Object (arbitration and control bits are set during configuration) exists for the message, only the data bytes are updated and the TxRqst bit (CAN_IFn_MCON[8]) with NewDat bit (CAN_IFn_MCON[15]) are set to start the transmission. If several transmit messages are assigned to the same Message Object (when the number of Message Objects is not sufficient), the whole Message Object has to be configured before the transmission of this message is requested.

The transmission of any number of Message Objects may be requested at the same time. Message objects are transmitted subsequently according to their internal priority. Messages may be updated or set to not valid any time, even when their requested transmission is still pending. The old data will be discarded when a message is updated before its pending transmission has started.

Depending on the configuration of the Message Object, the transmission of a message may be requested autonomously by the reception of a remote frame with a matching identifier.

Disabled Automatic Retransmission

In accordance with the CAN Specification (see ISO11898, 6.3.3 Recovery Management), the C_CAN provides means for automatic retransmission of frames that have lost arbitration or have been disturbed by errors during transmission. The frame transmission service will not be confirmed to the user before the transmission is successfully completed. This means that, by default, automatic retransmission is enabled. It can be disabled to enable the C_CAN to work within a Time Triggered CAN (TTCAN, see ISO11898-1) environment.

The Disabled Automatic Retransmission mode is enabled by setting the Disable Automatic Retransmission (DAR bit (CAN_CON[5])) to one. In this operation mode, the programmer has to consider the different behavior of bits TxRqst (CAN_IFn_MCON[8]) and NewDat (CAN_IFn_MCON[15]) of the Message Buffers:

- When a transmission starts, bit TxRqst of the respective Message Buffer is cleared, while bit NewDat remains set.
- When the transmission completed successfully, bit NewDat is cleared.
- When a transmission fails (lost arbitration or error), bit NewDat remains set.
- To restart the transmission, the software should set the bit TxRqst again.

6.23.5 Test Mode

Test Mode is entered by setting the Test bit (CAN_CON[7]). In Test Mode, bits Tx1 (CAN_TEST[6]), Tx0 (CAN_TEST[5]), LBack (CAN_TEST[4]), Silent (CAN_TEST[3]) and Basic (CAN_TEST[2]) are writeable. Bit Rx (CAN_TEST[7]) monitors the state of the CAN_RX pin and therefore is only readable. All Test Register functions are disabled when the Test bit is cleared.

6.23.5.1 Silent Mode

The CAN Core can be set in Silent Mode by programming the Silent bit (CAN_TEST[3]) to one. In Silent Mode, the C_CAN is able to receive valid data frames and valid remote frames, but it sends only recessive bits on the CAN bus and it cannot start a transmission. If the CAN Core is required to send a dominant bit (ACK bit, Error Frames), the bit is rerouted internally so that the CAN Core monitors this dominant bit, although the CAN bus may remain in recessive state. The Silent Mode can be used to analyze the traffic on a CAN bus without affecting it by the transmission of dominant bits. Figure 6.23-2 CAN Core in Silent Mode shows the connection of signals CAN_TX and CAN_RX to the CAN Core in Silent Mode.

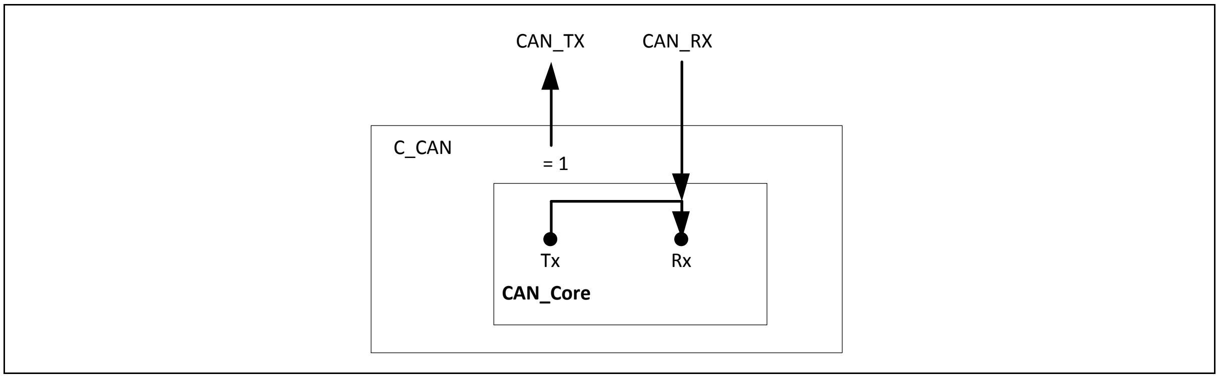


Figure 6.23-2 CAN Core in Silent Mode

6.23.5.2 Loop Back Mode

The CAN Core can be set in Loop Back Mode by programming the Test Register bit LBack (CAN_TEST[4]) to one. In Loop Back Mode, the CAN Core treats its own transmitted messages as received messages and stores them in a Receive Buffer (if they pass acceptance filtering). Figure 6.23-3 shows the connection of signals, CAN_TX and CAN_RX, to the CAN Core in Loop Back Mode.

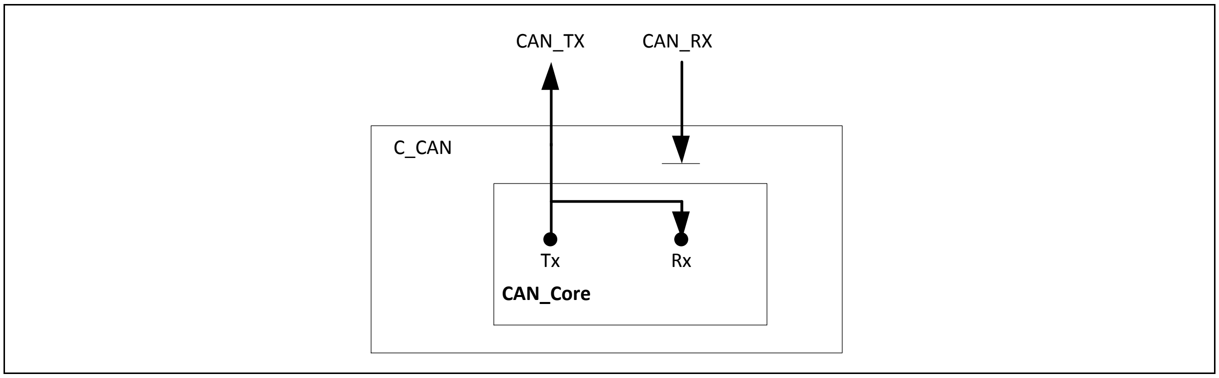


Figure 6.23-3 CAN Core in Loop Back Mode

This mode is provided for self-test functions. To be independent from external stimulation, the CAN Core ignores acknowledge errors (recessive bit sampled in the acknowledge slot of a data/ remote frame) in Loop Back Mode. In this mode, the CAN Core performs an internal feedback from its Tx output to its Rx input. The actual value of the CAN_RX input pin is disregarded by the CAN Core. The transmitted messages can be monitored on the CAN_TX pin.

6.23.5.3 Loop Back Combined with Silent Mode

It is also possible to combine Loop Back Mode and Silent Mode by programming bits LBack (CAN_TEST[4]) and Silent (CAN_TEST[3]) to one at the same time. This mode can be used for a “Hot Selftest”, which means that C_CAN can be tested without affecting a running CAN system connected to the CAN_TX and CAN_RX pins. In this mode, the CAN_RX pin is disconnected from the CAN Core and the CAN_TX pin is held recessive. Figure 6.23-4 shows the connection of signals CAN_TX and CAN_RX to the CAN Core in case of the combination of Loop Back Mode with Silent Mode.

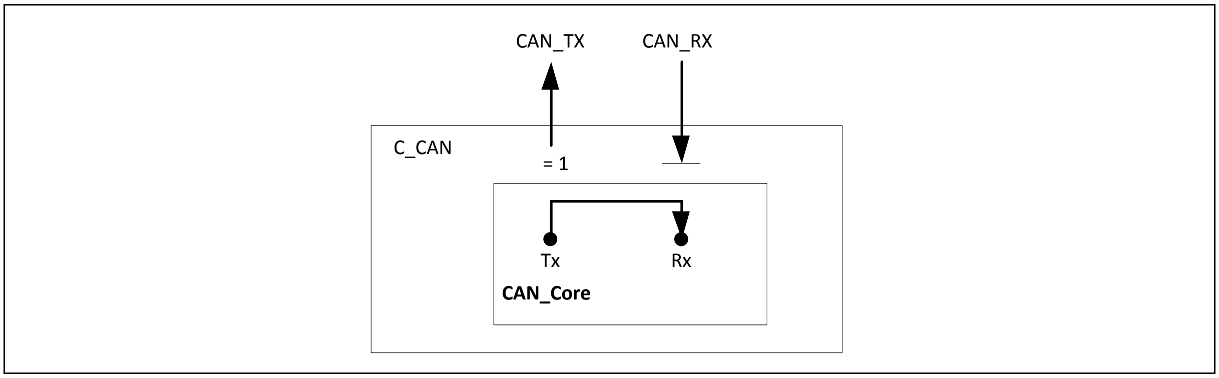


Figure 6.23-4 CAN Core in Loop Back Mode Combined with Silent Mode

6.23.5.4 Basic Mode

The CAN Core can be set in Basic Mode by programming the Basic bit (CAN_TEST[2]) to one. In this mode, the C_CAN runs without the Message RAM.

The IF1 Registers are used as Transmit Buffer. The transmission of the contents of the IF1 Registers is requested by writing the Busy bit (CAN_IFn_CREQ[15]) of the IF1 Command Request Register to one. The IF1 Registers are locked while the Busy bit is set. The Busy bit indicates that the transmission is pending.

As soon the CAN bus is idle, the IF1 Registers are loaded into the shift register of the CAN Core and the transmission is started. When the transmission has been completed, the Busy bit is reset and the locked IF1 Registers are released.

A pending transmission can be aborted at any time by resetting the Busy bit in the IF1 Command Request Register while the IF1 Registers are locked. If the software has reset the Busy bit, a possible retransmission in case of lost arbitration or in case of an error is disabled.

The IF2 Registers are used as a Receive Buffer. After the reception of a message the contents of the shift register is stored into the IF2 Registers, without any acceptance filtering.

Additionally, the actual contents of the shift register can be monitored during the message transfer. Each time a read Message Object is initiated by writing the Busy bit of the IF2 Command Request Register to one, the contents of the shift register are stored into the IF2 Registers.

In Basic Mode, the evaluation of all Message Object related control and status bits and the control bits of the IFn Command Mask Registers are turned off. The message number of the Command request registers is not evaluated. The NewDat (CAN_IFn_MCON[15]) and MsgLst (CAN_IFn_MCON[14]) bits retain their function, DLC3-0 indicates the received DLC (CAN_IFn_MCON[3:0]), and the other control bits are read as ‘0’.

6.23.5.5 Software Control of CAN_TX Pin

Four output functions are available for the CAN transmit pin, CAN_TX. In addition to its default function (serial data output), the CAN transmit pin can drive the CAN Sample Point signal to monitor CAN_Core’s bit timing and it can drive constant dominant or recessive values. The latter two functions, combined with the readable CAN receive pin CAN_RX, can be used to check the physical layer of the CAN bus.

The output mode for the CAN_TX pin is selected by programming the Tx1 (CAN_TEST[6]) and Tx0 (CAN_TEST[5]) bits.

The three test functions of the CAN_TX pin interfere with all CAN protocol functions. CAN_TX must be left in its default function when CAN message transfer or any of the test modes (Loop Back Mode, Silent Mode or Basic Mode) are selected.

6.23.6 CAN Communications

6.23.6.1 Managing Message Objects

The configuration of the Message Objects in the Message RAM (with the exception of the bits MsgVal, NewDat, IntPnd and TxRqst) will not be affected by resetting the chip. All the Message Objects must be initialized by the application software or they must be “not valid” (MsgVal bit = ‘0’) and the bit timing must be configured before the application software clears the Init bit (CAN_CON[0]).

The configuration of a Message Object is done by programming Mask, Arbitration, Control and Data fields of one of the two interface registers to the desired values. By writing to the corresponding IFn Command Request Register, the IFn Message Buffer Registers are loaded into the addressed Message Object in the Message RAM.

When the Init bit is cleared, the CAN Protocol Controller state machine of the CAN_Core and the state machine of the Message Handler control the internal data flow of the C_CAN. Received messages that pass the acceptance filtering are stored into the Message RAM, messages with pending transmission request are loaded into the CAN_Core’s Shift Register and are transmitted through the CAN bus.

The application software reads received messages and updates messages to be transmitted through the IFn Interface Registers. Depending on the configuration, the application software is interrupted on certain CAN message and CAN error events.

6.23.6.2 Message Handler State Machine

The Message Handler controls the data transfer between the Rx/Tx Shift Register of the CAN Core, the Message RAM and the IFn Registers.

The Message Handler FSM controls the following functions:

- Data Transfer from IFn Registers to the Message RAM
- Data Transfer from Message RAM to the IFn Registers
- Data Transfer from Shift Register to the Message RAM
- Data Transfer from Message RAM to Shift Register
- Data Transfer from Shift Register to the Acceptance Filtering unit
- Scanning of Message RAM for a matching Message Object
- Handling of TxRqst flags
- Handling of interrupts.

6.23.6.3 Data Transfer from/to Message RAM

When the application software initiates a data transfer between the IFn Registers and Message RAM, the Message Handler sets the Busy bit (CAN_IFn_CREQ[15]) to ‘1’. After the transfer has completed, the Busy bit is again cleared (see Figure 6.23-5).

The respective Command Mask Register specifies whether a complete Message Object or only parts of it will be transferred. Due to the structure of the Message RAM, it is not possible to write single bits/bytes of one Message Object. It is always necessary to write a complete Message Object into the Message RAM. Therefore, the data transfer from the IFn Registers to the Message RAM requires a read-modify-write cycle. First, those parts of the Message Object that are not to be changed are read from the Message RAM and then the complete contents of the Message Buffer Registers are written into the Message Object.

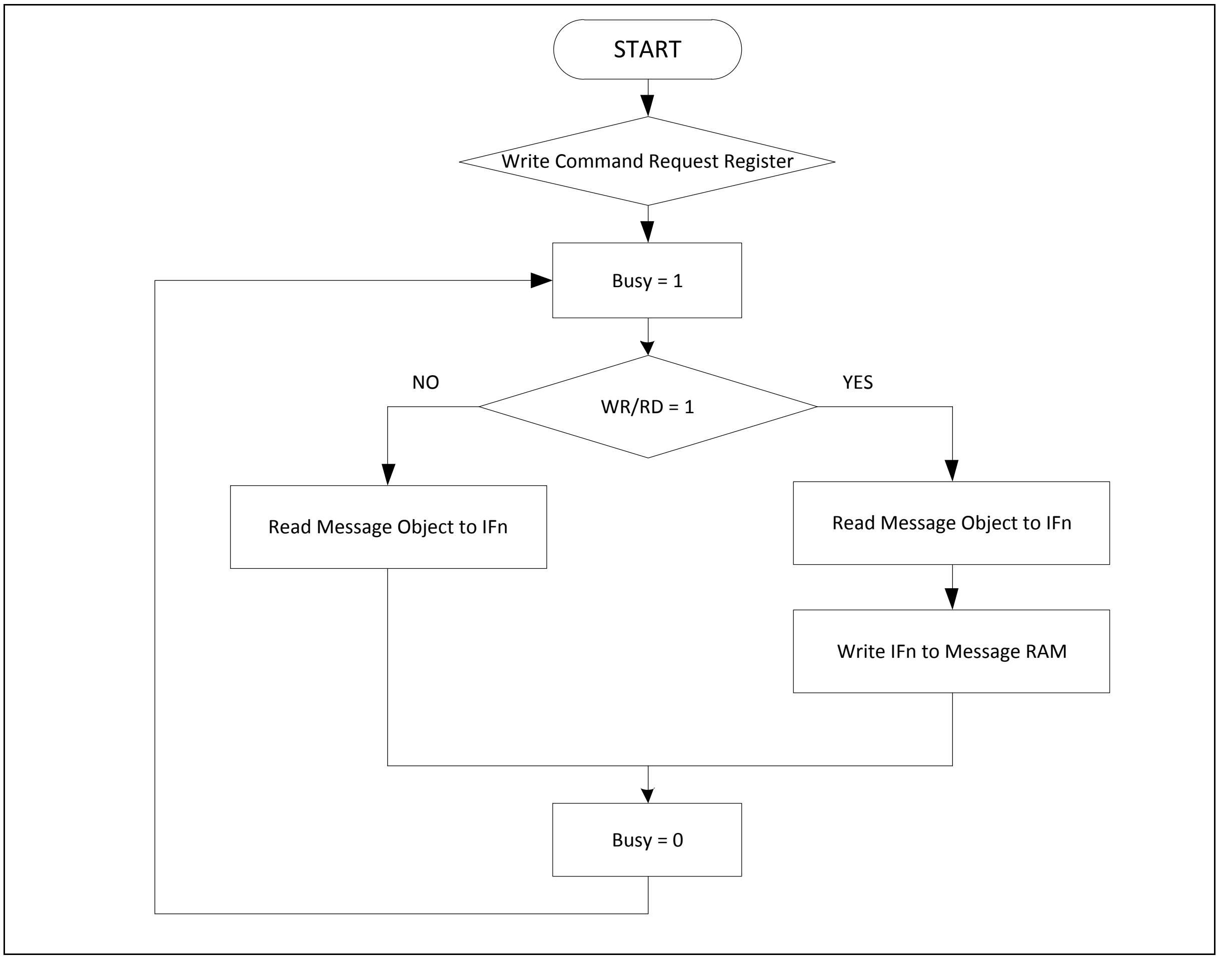

START
Write Command Request Register
Busy = 1
NO
WR/RD = 1
YES
Read Message Object to IFn
Read Message Object to IFn
Write IFn to Message RAM
Busy = 0

Figure 6.23-5 Data transfer between IFn Registers and Message

After a partial write of a Message Object, the Message Buffer Registers that are not selected in the Command Mask Register will set the actual contents of the selected Message Object.

After a partial read of a Message Object, the Message Buffer Registers that are not selected in the Command Mask Register will be left unchanged.

6.23.6.4 Message Transmission

If the shift register of the CAN Core cell is ready for loading and if there is no data transfer between the IFn Registers and Message RAM, the MsgVal bit (CAN_IFn_ARB2[15]) and TxRqst bits (CAN_TXREQ1/2) are evaluated. The valid Message Object with the highest priority pending transmission request is loaded into the shift register by the Message Handler and the transmission is started. The NewDat (CAN_IFn_MCON[15]) bit of the Message Object is reset.

After a successful transmission and also if no new data was written to the Message Object (NewDat = '0') since the start of the transmission, the TxRqst bit of the Message Control register (CAN_IFn_MCON[8]) will be reset. If TxIE bit (CAN_IFn_MCON[11]) is set, IntPnd bit (CAN_IFn_MCON[13]) of the Interrupt Identifier register will be set after a successful transmission. If the C_CAN has lost the arbitration or if an error occurred during the transmission, the message will be retransmitted as soon as the CAN bus is free again. Meanwhile, if the transmission of a message with higher priority has been requested, the messages will be transmitted in the order of their priority.

6.23.6.5 Acceptance Filtering of Received Messages

When the arbitration and control field (Identifier + IDE + RTR + DLC) of an incoming message is completely shifted into the Rx/Tx Shift Register of the CAN Core, the Message Handler FSM starts the scanning of the Message RAM for a matching valid Message Object.

To scan the Message RAM for a matching Message Object, the Acceptance Filtering unit is loaded with the arbitration bits from the CAN Core shift register. The arbitration and mask fields (including MsgVal (CAN_IFn_ARB2[15]), UMask (CAN_IFn_MCON[12]), NewDat (CAN_IFn_MCON[15]) and EoB (CAN_IFn_MCON[7])) of Message Object 1 are then loaded into the Acceptance Filtering unit and compared with the arbitration field from the shift register. This is repeated with each following Message Object until a matching Message Object is found or until the end of the Message RAM is reached.

If a match occurs, the scan is stopped and the Message Handler FSM proceeds depending on the type of frame (Data Frame or Remote Frame) received.

Reception of Data Frame

The Message Handler FSM stores the message from the CAN Core shift register into the respective Message Object in the Message RAM. Not only the data bytes, but all arbitration bits and the Data Length Code are stored into the corresponding Message Object. This is done to keep the data bytes connected with the identifier even if arbitration mask registers are used.

The NewDat bit (CAN_IFn_MCON[15]) is set to indicate that new data (not yet seen by the software) has been received. The application software should reset NewDat bit when the Message Object has been read. If at the time of reception, the NewDat bit was already set, MsgLst (CAN_IFn_MCON[14]) is set to indicate that the previous data (supposedly not seen by the software) is lost. If the RxIE bit (CAN_IFn_MCON[10]) is set, the IntPnd bit (CAN_IFn_MCON[13]) is set, causing the Interrupt Register to point to this Message Object.

The TxRqst bit (CAN_IFn_MCON[8]) of this Message Object is reset to prevent the transmission of a Remote Frame, while the requested Data Frame has just been received.

Reception of Remote Frame

When a Remote Frame is received, three different configurations of the matching Message Object have to be considered:

1. Dir (CAN_IFn_ARB2[13]) = ‘1’ (direction = transmit), RmtEn (CAN_IFn_MCON[9]) = ‘1’ and UMask (CAN_IFn_MCON[12]) = ‘1’ or ’0’
2. At the reception of a matching Remote Frame, the TxRqst bit of this Message Object is set. The rest of the Message Object remains unchanged.
3. Dir = ‘1’ (direction = transmit), RmtEn = ‘0’ and UMask = ’0’
4. At the reception of a matching Remote Frame, the TxRqst bit of this Message Object remains unchanged; the Remote Frame is ignored.
5. Dir = ‘1’ (direction = transmit), RmtEn = ‘0’ and UMask = ’1’
6. At the reception of a matching Remote Frame, the TxRqst bit of this Message Object is reset. The arbitration and control field (Identifier + IDE + RTR + DLC) from the shift register is stored in the Message Object of the Message RAM and the NewDat bit (CAN_IFn_MCON[15]) of this Message Object is set. The data field of the Message Object remains unchanged; the Remote Frame is treated similar to a received Data Frame.

6.23.6.6 Receive/Transmit Priority

The receive/transmit priority for the Message Objects is attached to the message number. Message Object 1 has the highest priority, while Message Object 32 has the lowest priority. If more than one transmission request is pending, they are serviced due to the priority of the corresponding Message Object

6.23.6.7 Configuring a Transmit Object

Table 6.23-1 shows how a Transmit Object should be initialized.

Ms	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
1	appl.	appl.	appl.	1	1	0	0	0	appl.	0	appl.	0

Table 6.23-1 Initialization of Transmit Object

Note: appl. = application software.

The Arbitration Register values (ID28-0 (CAN_IFn_ARB1/2) and Xtd bit (CAN_IFn_ARB2[14])) are provided by the application. They define the identifier and type of the outgoing message. If an 11-bit Identifier ("Standard Frame") is used, it is programmed to ID28 - ID18. The ID17 - ID0 can then be disregarded.

If the TxIE bit (CAN_IFn_MCON[11]) is set, the IntPnd bit (CAN_IFn_MCON[13]) will be set after a successful transmission of the Message Object.

If the RmtEn bit (CAN_IFn_MCON[9]) is set, a matching received Remote Frame will cause the TxRqst bit (CAN_IFn_MCON[8]) to be set; the Remote Frame will autonomously be answered by a Data Frame.

The Data Register values (DLC3-0 (CAN_IFn_MCON[3:0]), Data(0)-(7)) are provided by the application, TxRqst and RmtEn may not be set before the data is valid.

The Mask Registers (Msk28-0, UMask, MXtd and MDir bits) may be used (UMask (CAN_IFn_MCON[12]) = '1') to allow groups of Remote Frames with similar identifiers to set the TxRqst bit. The Dir bit (CAN_IFn_ARB2[13]) should not be masked.

6.23.6.8 Updating a Transmit Object

The software may update the data bytes of a Transmit Object any time through the IFn Interface registers, neither MsgVal bit (CAN_IFn_ARB2[15]) nor TxRqst (CAN_IFn_MCON[8]) have to be reset before the update.

Even if only a part of the data bytes are to be updated, all four bytes of the corresponding IFn Data A Register or IFn Data B Register have to be valid before the contents of that register are transferred to the Message Object. Either the application software has to write all four bytes into the IFn Data Register or the Message Object is transferred to the IFn Data Register before the software writes the new data bytes.

When only the (eight) data bytes are updated, first 0x0087 is written to the Command Mask Register and then the number of the Message Object is written to the Command Request Register, concurrently updating the data bytes and setting TxRqst.

To prevent the reset of TxRqst at the end of a transmission that may already be in progress while the data is updated, NewDat (CAN_IFn_MCON[15]) has to be set together with TxRqst.

When NewDat is set together with TxRqst, NewDat will be reset as soon as the new transmission has started.

6.23.6.9 Configuring a Receive Object

Table 6.23-2 shows how a Receive Object should be initialized.

MsgVal	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
1	appl.	appl.	appl.	1	0	0	0	appl.	0	0	0	0

Table 6.23-2 Initialization of Receive Object

The Arbitration Registers values (ID28-0 (CAN_IFn_ARB1/2) and Xtd bit (CAN_IFn_ARB2[14])) are provided by the application. They define the identifier and type of accepted received messages. If an 11-bit Identifier (“Standard Frame”) is used, it is programmed to ID28 - ID18. Then ID17 - ID0 can be disregarded. When a Data Frame with an 11-bit Identifier is received, ID17 - ID0 will be set to ‘0’.

If the RxIE bit (CAN_IFn_MCON[10]) is set, the IntPnd bit (CAN_IFn_MCON[13]) will be set when a received Data Frame is accepted and stored in the Message Object.

The Data Length Code (DLC3-0 (CAN_IFn_MCON[3:0])) is provided by the application. When the Message Handler stores a Data Frame in the Message Object, it will store the received Data Length Code and eight data bytes. If the Data Length Code is less than 8, the remaining bytes of the Message Object will be overwritten by unspecified values.

The Mask Registers (Msk28-0, UMask, MXtd and MDir bits) may be used (UMask (CAN_IFn_MCON[12]) = ’1’) to allow groups of Data Frames with similar identifiers to be accepted. The Dir bit (CAN_IFn_ARB2[13]) should not be masked in typical applications.

6.23.6.10 Handling Received Messages

The application software may read a received message any time through the IFn Interface registers. The data consistency is guaranteed by the Message Handler state machine.

Typically, the software will write first 0x007F to the Command Mask Register and then the number of the Message Object to the Command Request Register. This combination will transfer the whole received message from the Message RAM into the Message Buffer Register. Additionally, the bits NewDat (CAN_IFn_MCON[15]) and IntPnd (CAN_IFn_MCON[13]) are cleared in the Message RAM (not in the Message Buffer).

If the Message Object uses masks for acceptance filtering, the arbitration bits show which of the matching messages have been received.

The actual value of NewDat shows whether a new message has been received since the last time this Message Object was read. The actual value of MsgLst (CAN_IFn_MCON[14]) shows whether more than one message has been received since the last time this Message Object was read. MsgLst will not be automatically reset.

By means of a Remote Frame, the software may request another CAN node to provide new data for a receive object. Setting the TxRqst bit (CAN_IFn_MCON[8]) of a receive object will cause the transmission of a Remote Frame with the receive object’s identifier. This Remote Frame triggers the other CAN node to start the transmission of the matching Data Frame. If the matching Data Frame is received before the Remote Frame could be transmitted, the TxRqst bit is automatically reset.

6.23.6.11 Configuring a FIFO Buffer

With the exception of the EoB bit (CAN_IFn_MCON[7]), the configuration of Receive Objects belonging to a FIFO Buffer is the same as the configuration of a (single) Receive Object, see Section 6.5.7.9: Configuring a Receive Object.

To concatenate two or more Message Objects into a FIFO Buffer, the identifiers and masks (if used) of these Message Objects have to be programmed to matching values. Due to the implicit priority of the Message Objects, the Message Object with the lowest number will be the first Message Object of the

FIFO Buffer. The EoB bit of all Message Objects of a FIFO Buffer except the last have to be programmed to zero. The EoB bit of the last Message Object of a FIFO Buffer is set to one, configuring it as the End of the Block.

6.23.6.12 Receiving Messages with FIFO Buffers

Received messages with identifiers matching to a FIFO Buffer are stored into a Message Object of this FIFO Buffer starting with the Message Object with the lowest message number.

When a message is stored into a Message Object of a FIFO Buffer, the NewDat bit (CAN_IFn_MCON[15]) of this Message Object is set. By setting NewDat while EoB (CAN_IFn_MCON[7]) is zero, the Message Object is locked for further write access by the Message Handler until the application software has written the NewDat bit back to zero.

Messages are stored into a FIFO Buffer until the last Message Object of this FIFO Buffer is reached. If none of the preceding Message Objects is released by writing NewDat to zero, all further messages for this FIFO Buffer will be written into the last Message Object of the FIFO Buffer and therefore overwrite the previous messages.

6.23.6.13 Reading from a FIFO Buffer

When the application software transfers the contents of a Message Object to the IFn Message Buffer register by writing its number to the IFn Command Request Register, the corresponding Command Mask Register should be programmed in such a way that bits NewDat (CAN_IFn_MCON[15]) and IntPnd (CAN_IFn_MCON[13]) are reset to zero (TxRqst/NewDat (CAN_IFn_CMASK[2]) = ‘1’ and ClrIntPnd (CAN_IFn_CMASK[3]) = ‘1’). The values of these bits in the Message Control Register always reflect the status before resetting the bits.

To assure the correct function of a FIFO Buffer, the application software should read the Message Objects starting at the FIFO Object with the lowest message number.

Figure 6.23-6 shows how a set of Message Objects which are concatenated to a FIFO Buffer can be handled by the application software.

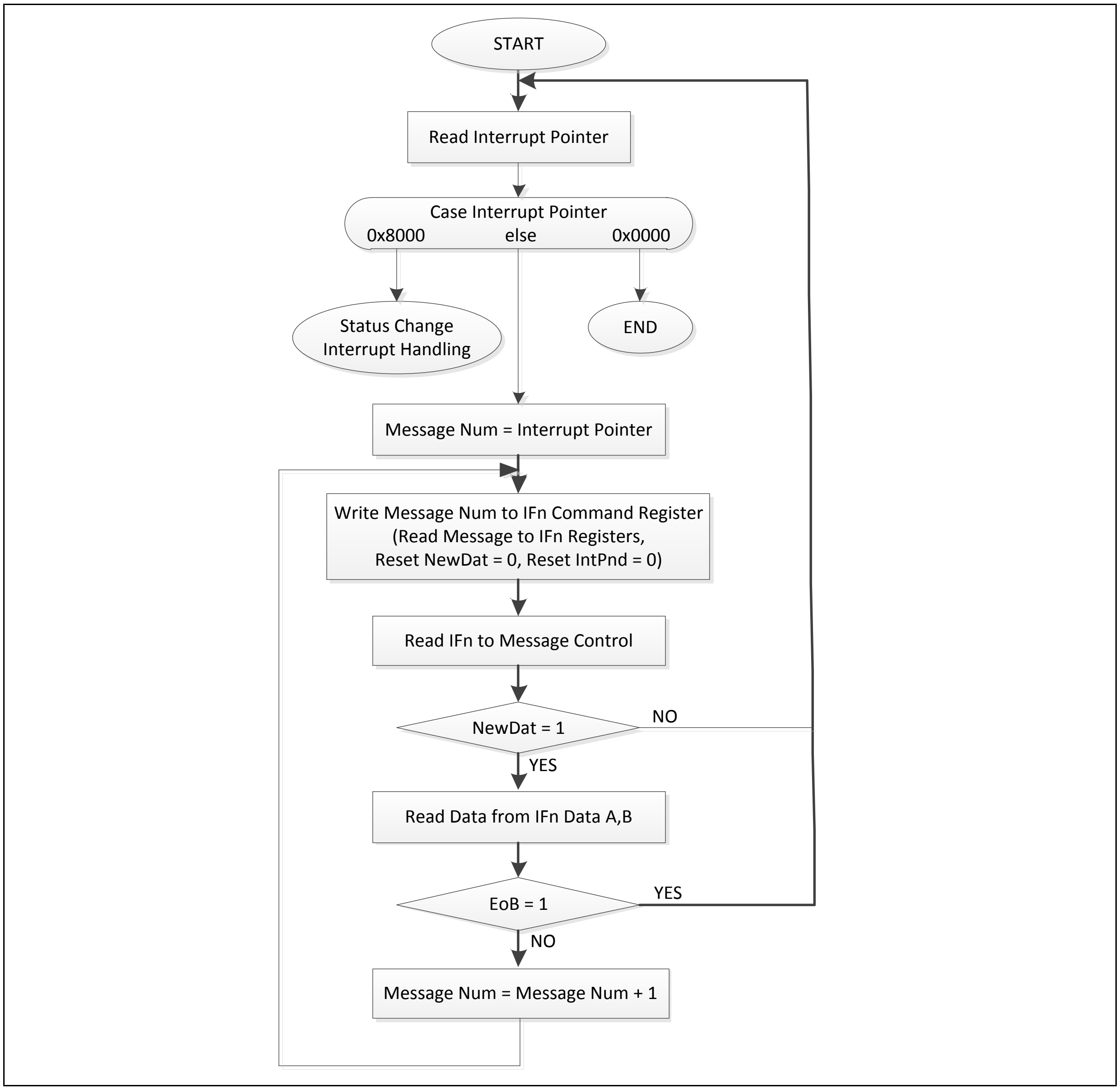

START
Read Interrupt Pointer
Case Interrupt Pointer
0x8000
else
0x0000
Status Change Interrupt Handling
END
Message Num = Interrupt Pointer
Write Message Num to IFn Command Register (Read Message to IFn Registers, Reset NewDat = 0, Reset IntPnd = 0)
Read IFn to Message Control
NewDat = 1
NO
YES
Read Data from IFn Data A,B
EoB = 1
YES
NO
Message Num = Message Num + 1

Figure 6.23-6 Application Software Handling of a FIFO Buffer

6.23.6.14 Handling Interrupts

If several interrupts are pending, the CAN Interrupt Register will point to the pending interrupt with the highest priority, disregarding their chronological order. An interrupt remains pending until the application software has cleared it.

The Status Interrupt has the highest priority. Among the message interrupts, interrupt priority of the Message Object decreases with increasing message number.

A message interrupt is cleared by clearing the IntPnd bit (CAN_IFn_MCON[13]) of the Message Object. The Status Interrupt is cleared by reading the Status Register.

The interrupt identifier, IntId, in the Interrupt Register, indicates the cause of the interrupt. When no interrupt is pending, the register will hold the value zero. If the value of the Interrupt Register is different from zero, then there is an interrupt pending and, if IE (CAN_CON[1]) is set, the CAN_INT interrupt signal is active. The interrupt remains active until the Interrupt Register is back to value zero (the cause of the interrupt is reset) or until IE is reset.

The value 0x8000 indicates that an interrupt is pending because the CAN Core has updated (not necessarily changed) the Status Register (Error Interrupt or Status Interrupt). This interrupt has the highest priority. The application software can update (reset) the status bits RxOk (CAN_STATUS[4]), TxOk (CAN_STATUS[3]) and LEC (CAN_STATUS[2:0]), but a write access of the software to the Status Register can never generate or reset an interrupt.

All other values indicate that the source of the interrupt is one of the Message Objects. IntId points to the pending message interrupt with the highest interrupt priority.

The application software controls whether a change of the Status Register may cause an interrupt (bits EIE (CAN_CON[3]) and SIE (CAN_CON[2])) and whether the interrupt line becomes active when the Interrupt Register is different from zero (bit IE in the CAN Control Register). The Interrupt Register will be updated even when IE is reset.

The application software has two possibilities to follow the source of a message interrupt. First, it can follow the IntId in the Interrupt Register and second it can poll the Interrupt Pending Register.

An interrupt service routine that is reading the message that is the source of the interrupt may read the message and reset the Message Object's IntPnd at the same time (bit ClrIntPnd (CAN_IFn_CMASK[3])). When IntPnd is cleared, the Interrupt Register will point to the next Message Object with a pending interrupt.

6.23.6.15 Configuring the Bit Timing

Even if minor errors in the configuration of the CAN bit timing do not result in immediate failure, the performance of a CAN network can be reduced significantly.

In many cases, the CAN bit synchronization will amend a faulty configuration of the CAN bit timing to such a degree that only occasionally an error frame is generated. However, in the case of arbitration, when two or more CAN nodes simultaneously try to transmit a frame, a misplaced sample point may cause one of the transmitters to become error passive.

The analysis of such sporadic errors requires a detailed knowledge of the CAN bit synchronization inside a CAN node and interaction of the CAN nodes on the CAN bus.

6.23.6.16 Bit Time and Bit Rate

CAN supports bit rates in the range of lower than 1 Kbit/s up to 1000 Kbit/s. Each member of the CAN network has its own clock generator, usually a quartz oscillator. The timing parameter of the bit time (i.e. the reciprocal of the bit rate) can be configured individually for each CAN node, creating a common bit rate even though the oscillator periods of the CAN nodes (fosc) may be different.

The frequencies of these oscillators are not absolutely stable, small variations are caused by changes in temperature or voltage and by deteriorating components. As long as the variations remain inside a specific oscillator tolerance range (df), the CAN nodes are able to compensate for the different bit rates by re-synchronizing to the bit stream.

According to the CAN specification, the bit time is divided into four segments (see Figure 6.23-7). The Synchronization Segment, the Propagation Time Segment, the Phase Buffer Segment 1 and the Phase Buffer Segment 2. Each segment consists of a specific, programmable number of time quanta (Table 6.23-3). The length of the time quantum (tq), which is the basic time unit of the bit time, is defined by the CAN controller's APB clock fAPB and the BRP bit (CAN_BTIME[5:0]) : tq = BRP / fAPB.

The Synchronization Segment, Sync_Seg, is that part of the bit time where edges of the CAN bus level are expected to occur. The distance between an edge that occurs outside of Sync_Seg, and the Sync_Seg is called the phase error of that edge. The Propagation Time Segment, Prop_Seg, is intended to compensate for the physical delay time within the CAN network. The Phase Buffer Segments Phase_Seg1 and Phase_Seg2 surround the Sample Point. The (Re-)Synchronization Jump Width (SJW) defines how far a re-synchronization may move the Sample Point inside the limits defined by the Phase Buffer Segments to compensate for edge phase errors.

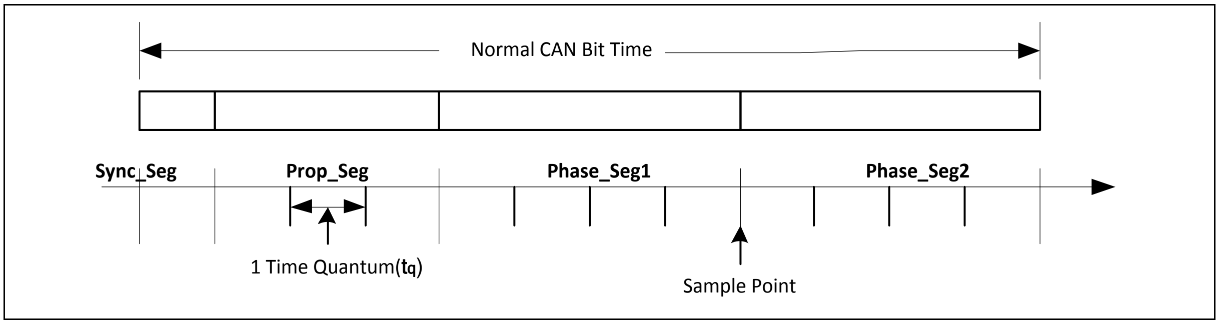


Figure 6.23-7 Bit Timing

Parameter	Range	Remark
BRP	[1..32]	Defines the length of the time quantum t_q
Sync_Seg	1 t_q	Fixed length, synchronization of bus input to APB clock
Prop_Seg	[1..8] t_q	Compensates for the physical delay time
Phase_Seg1	[1..8] t_q	Which may be lengthened temporarily by synchronization
Phase_Seg2	[1..8] t_q	Which may be shortened temporarily by synchronization
SJW	[1..4] t_q	Which may not be longer than either Phase Buffer Segment
This table describes the minimum programmable ranges required by the CAN protocol		

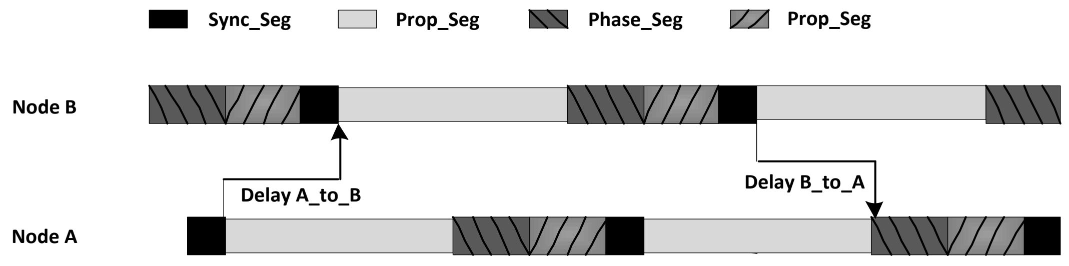
Table 6.23-3 CAN Bit Time Parameters

A given bit rate may be met by different bit time configurations, but for the proper function of the CAN network the physical delay time and the oscillator's tolerance range have to be considered.

6.23.6.17 Propagation Time Segment

This part of the bit time is used to compensate physical delay time within the network. These delay time consist of the signal propagation time on the bus and the internal delay time of the CAN nodes.

Any CAN node synchronized to the bit stream on the CAN bus will be out of phase with the transmitter of that bit stream, caused by the signal propagation time between the two nodes. The CAN protocol's non-destructive bitwise arbitration and the dominant acknowledge bit provided by receivers of CAN messages requires that a CAN node transmitting a bit stream must also be able to receive dominant bits transmitted by other CAN nodes that are synchronized to that bit stream. The example in Figure 6.23-8 shows the phase shift and propagation time between two CAN nodes.



Delay A_to_B >= node output delay (A) + bus line delay (A-> B) + node input delay (B)

Prop_seg >= Delay A_to_B + Delay B_to_A

Prop_Seg >= 2*[max(node output delay + bus line dealy + node input delay)]

Figure 6.23-8 Propagation Time Segment

In this example, both nodes A and B are transmitters, performing an arbitration for the CAN bus. Node A has sent its Start of Frame bit less than one bit time earlier than node B, therefore node B has synchronized itself to the received edge from recessive to dominant. Since node B has received this edge delay (A_to_B) after it has been transmitted, B's bit timing segments are shifted with respect to A. Node B sends an identifier with higher priority and so it will win the arbitration at a specific identifier bit when it transmits a dominant bit while node A transmits a recessive bit. The dominant bit transmitted by node B will arrive at node A after the delay (B_to_A).

Due to oscillator tolerances, the actual position of node A's Sample Point can be anywhere inside the nominal range of node A's Phase Buffer Segments, so the bit transmitted by node B must arrive at node A before the start of Phase_Seg1. This condition defines the length of Prop_Seg.

If the edge from recessive to dominant transmitted by node B arrives at node A after the start of Phase_Seg1, it can happen that node A samples a recessive bit instead of a dominant bit, resulting in a bit error and the destruction of the current frame by an error flag.

The error occurs only when two nodes arbitrate for the CAN bus that have oscillators of opposite ends of the tolerance range and that are separated by a long bus line. This is an example of a minor error in the bit timing configuration (Prop_Seg is too short) that causes sporadic bus errors.

Some CAN implementations provide an optional 3 Sample Mode but the C_CAN does not. In this mode, the CAN bus input signal passes a digital low-pass filter, using three samples and a majority logic to determine the valid bit value. This results in an additional input delay of 1 tq, requiring a longer Prop_Seg.

6.23.6.18 Phase Buffer Segments and Synchronization

The Phase Buffer Segments (Phase_Seg1 and Phase_Seg2) and the Synchronization Jump Width (SJW) are used to compensate for the oscillator tolerance. The Phase Buffer Segments may be lengthened or shortened by synchronization.

Synchronizations occur on edges from recessive to dominant, their purpose is to control the distance between edges and Sample Points.

Edges are detected by sampling the actual bus level in each time quantum and comparing it with the bus level at the previous Sample Point. A synchronization may be done only if a recessive bit was sampled at the previous Sample Point and if the bus level at the actual time quantum is dominant.

An edge is synchronous if it occurs inside of Sync_Seg, otherwise the distance between edge and the end of Sync_Seg is the edge phase error, measured in time quanta. If the edge occurs before Sync_Seg, the phase error is negative, else it is positive.

Two types of synchronization exist, Hard Synchronization and Re-synchronization.

A Hard Synchronization is done once at the start of a frame and inside a frame only when Re-synchronizations occur.

Hard Synchronization

After a hard synchronization, the bit time is restarted with the end of Sync_Seg, regardless of the edge phase error. Thus hard synchronization forces the edge, which has caused the hard synchronization to lie within the synchronization segment of the restarted bit time.

Bit Re-synchronization

Re-synchronization leads to a shortening or lengthening of the bit time such that the position of the sample point is shifted with regard to the edge.

When the phase error of the edge which causes Re-synchronization is positive, Phase_Seg1 is lengthened. If the magnitude of the phase error is less than SJW, Phase_Seg1 is lengthened by the magnitude of the phase error, else it is lengthened by SJW.

When the phase error of the edge, which causes Re-synchronization is negative, Phase_Seg2 is shortened. If the magnitude of the phase error is less than SJW, Phase_Seg2 is shortened by the magnitude of the phase error, else it is shortened by SJW.

When the magnitude of the phase error of the edge is less than or equal to the programmed value of SJW, the results of Hard Synchronization and Re-synchronization are the same. If the magnitude of the phase error is larger than SJW, the Re-synchronization cannot compensate the phase error completely, an error (phase error - SJW) remains.

Only one synchronization may be done between two Sample Points. The Synchronizations maintain a minimum distance between edges and Sample Points, giving the bus level time to stabilize and filtering out spikes that are shorter than (Prop_Seg + Phase_Seg1).

Apart from noise spikes, most synchronizations are caused by arbitration. All nodes synchronize “hard” on the edge transmitted by the “leading” transceiver that started transmitting first, but due to propagation delay time, they cannot become ideally synchronized. The “leading” transmitter does not necessarily win the arbitration, therefore the receivers have to synchronize themselves to different transmitters that subsequently “take the lead” and that are differently synchronized to the previously “leading” transmitter. The same happens at the acknowledge field, where the transmitter and some of the receivers will have to synchronize to that receiver that “takes the lead” in the transmission of the dominant acknowledge bit.

Synchronizations after the end of the arbitration will be caused by oscillator tolerance, when the differences in the oscillator’s clock periods of transmitter and receivers sum up during the time between synchronizations (at most ten bits). These summarized differences may not be longer than the SJW, limiting the oscillator’s tolerance range.

The examples in Figure 6.23-9 show how the Phase Buffer Segments are used to compensate for phase errors. There are three drawings of each two consecutive bit timings. The upper drawing shows the synchronization on a “late” edge, the lower drawing shows the synchronization on an “early” edge, and the middle drawing is the reference without synchronization.

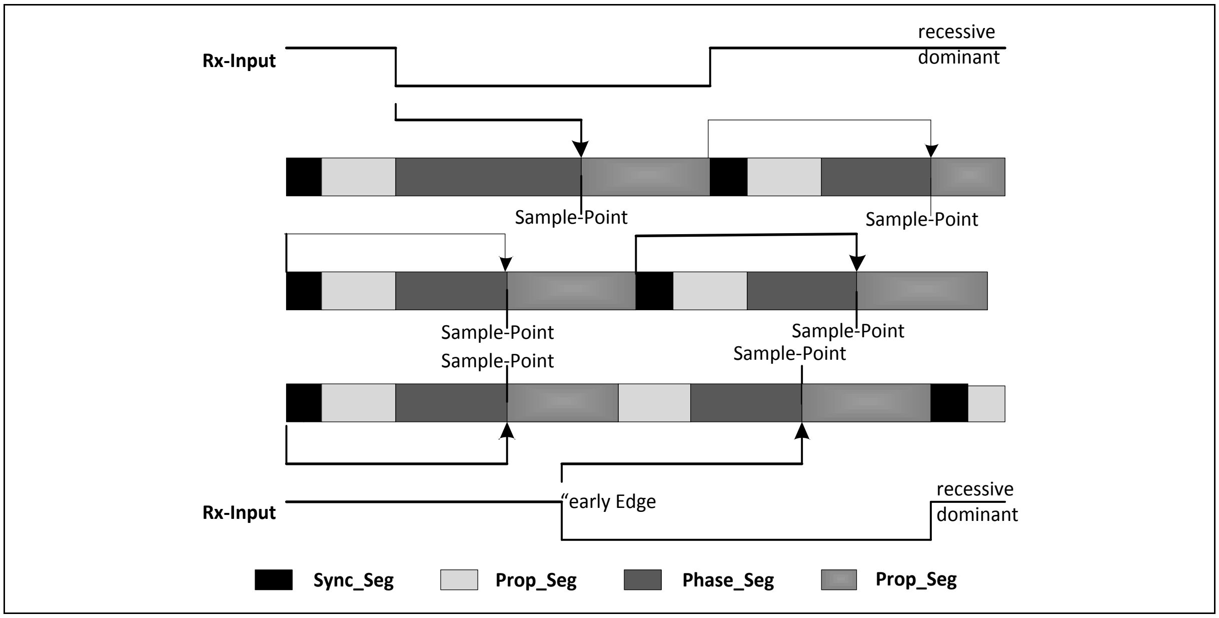


Figure 6.23-9 Synchronization on “Late” and “Early” Edges

In the first example, an edge from recessive to dominant occurs at the end of Prop_Seg. The edge is “late” since it occurs after the Sync_Seg. Reacting to the “late” edge, Phase_Seg1 is lengthened so that the distance from the edge to the Sample Point is the same as it would have been from the Sync_Seg to the Sample Point if no edge had occurred. The phase error of this “late” edge is less than SJW, so it is fully compensated and the edge from dominant to recessive at the end of the bit, which is one nominal bit time long, occurs in the Sync_Seg.

In the second example an edge from recessive to dominant occurs during Phase_Seg2. The edge is “early” since it occurs before a Sync_Seg. Reacting to the “early” edge, Phase_Seg2 is shortened and Sync_Seg is omitted, so that the distance from the edge to the Sample Point is the same as it would have been from an Sync_Seg to the Sample Point if no edge had occurred. As in the previous example, the magnitude of this “early” edge’s phase error is less than SJW, so it is fully compensated.

The Phase Buffer Segments are lengthened or shortened temporarily only; at the next bit time, the segments return to their nominal programmed values.

In these examples, the bit timing is seen from the point of view of the CAN implementation’s state machine, where the bit time starts and ends at the Sample Points. The state machine omits Sync_Seg when synchronising on an “early” edge because it cannot subsequently redefine that time quantum of Phase_Seg2 where the edge occurs to be the Sync_Seg.

The examples in Figure 6.23-11 show how short dominant noise spikes are filtered by synchronisations. In both examples the spike starts at the end of Prop_Seg and has the length of (Prop_Seg + Phase_Seg1).

In the first example, the Synchronization Jump Width is greater than or equal to the phase error of the spike’s edge from recessive to dominant. Therefore the Sample Point is shifted after the end of the spike; a recessive bus level is sampled.

In the second example, SJW is shorter than the phase error, so the Sample Point cannot be shifted far enough; the dominant spike is sampled as actual bus level.

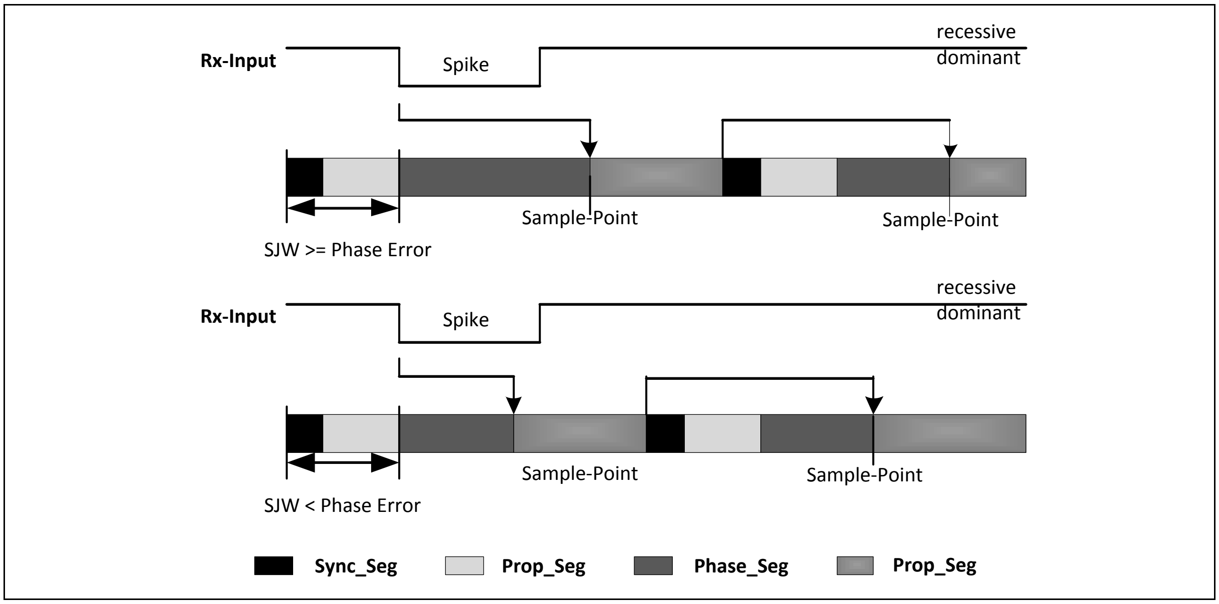


Figure 6.23-10 Filtering of Short Dominant Spikes

6.23.6.19 Oscillator Tolerance Range

The oscillator tolerance range was increased when the CAN protocol was developed from version 1.1 to version 1.2 (version 1.0 was never implemented in silicon). The option to synchronize on edges from dominant to recessive became obsolete, only edges from recessive to dominant are considered for synchronization. The protocol update to version 2.0 (A and B) had no influence on the oscillator tolerance.

The tolerance range d*f* for an oscillator frequency fosc around the nominal frequency fnom is:

(1 – d*f*). fnom <= fosc <= (1 + d*f*) . fnom

It depends on the proportions of Phase_Seg1, Phase_Seg2, SJW and the bit time. The maximum tolerance df is the defined by two conditions (both shall be met):

$$I: df <= \frac{\min(\mathrm{Phase\_Seg1}, \mathrm{Phase\_Seg2})}{2 * (13 * \mathrm{bit\_time} - Phase\_Seg2}$$

$$II: df <= \frac{\mathrm{SJW}}{20 * \mathrm{bit\_tim}}$$

Note: These conditions base on the APB cock = *f* osc.

It has to be considered that SJW may not be larger than the smaller of the Phase Buffer Segments and that the Propagation Time Segment limits that part of the bit time that may be used for the Phase Buffer Segments.

The combination Prop_Seg = 1 and Phase_Seg1 = Phase_Seg2 = SJW = 4 allows the largest possible oscillator tolerance of 1.58%. This combination with a Propagation Time Segment of only 10% of the bit time is not suitable for short bit times; it can be used for bit rates of up to 125 Kbit/s (bit time = 8us) with a bus length of 40 m.

6.23.6.20 Configuring the CAN Protocol Controller

In most CAN implementations and also in the C_CAN, the bit timing configuration is programmed in two register bytes. The sum of Prop_Seg and Phase_Seg1 (as TSEG1 (CAN_BTIME[11:8])) is combined with Phase_Seg2 (as TSEG2 (CAN_BTIME[14:12])) in one byte, SJW (CAN_BTIME[7:6]) and BRP (CAN_BTIME[5:0]) are combined in the other byte.

In these bit timing registers, the four components TSEG1, TSEG2, SJW and BRP have to be programmed to a numerical value that is one less than its functional value. Therefore, instead of values in the range of [1..n], values in the range of [0..n-1] are programmed. That way, e.g. SJW (functional range of [1..4]) is represented by only two bits.

Therefore the length of the bit time is (programmed values) [TSEG1 + TSEG2 + 3] tq or (functional values) [Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] tq.

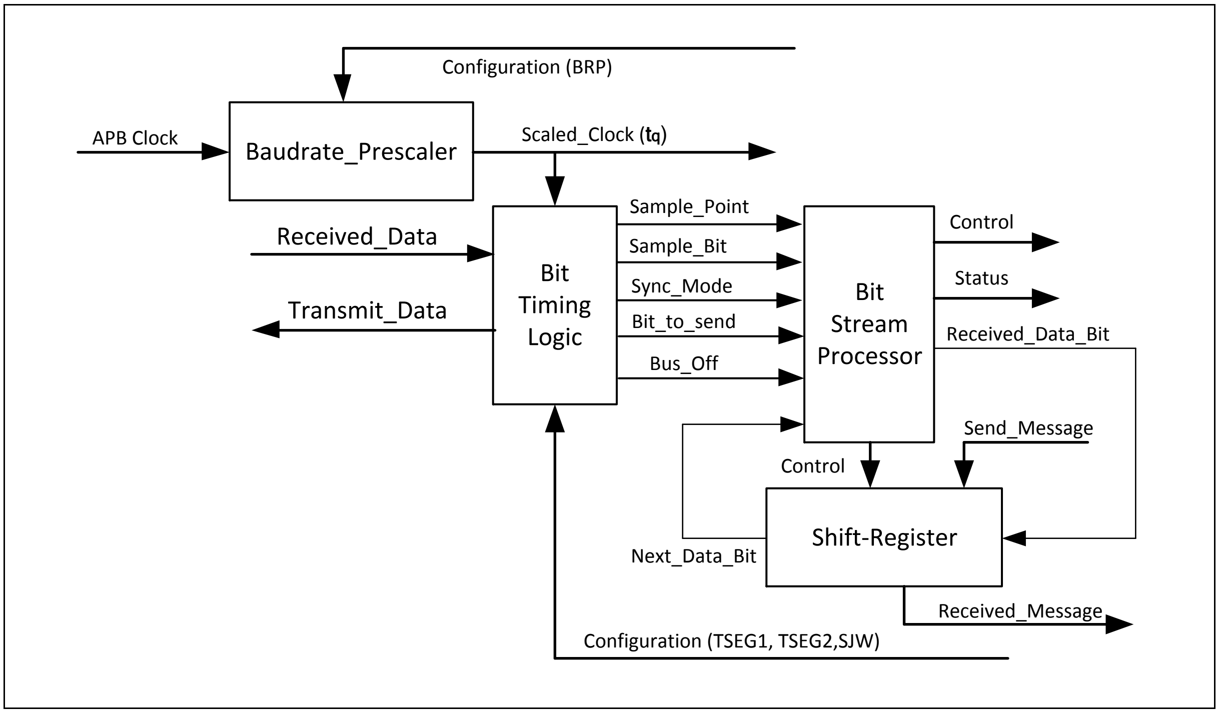


Figure 6.23-11 Structure of the CAN Core's CAN Protocol Controller

The data in the bit timing registers is the configuration input of the CAN protocol controller. The Baud Rate Prescaler (configured by BRP) defines the length of the time quantum, the basic time unit of the bit time; the Bit Timing Logic (configured by TSEG1, TSEG2 and SJW) defines the number of time quanta in the bit time.

The processing of the bit time, the calculation of the position of the Sample Point, and occasional synchronizations are controlled by the BTL (Bit Timing Logic) state machine, which is evaluated once each time quantum. The rest of the CAN protocol controller, the BSP (Bit Stream Processor) state machine is evaluated once each bit time, at the Sample Point.

The Shift Register sends the messages serially and parallelizes received messages. It's loading and shifting is controlled by the BSP.

The BSP translates messages into frames and vice versa. It generates and discards the enclosing fixed format bits, inserts and extracts stuff bits, calculates and checks the CRC code, performs the error management, and decides which type of synchronization is to be used. It is evaluated at the Sample Point and processes the sampled bus input bit. The time that is needed to calculate the next bit to be sent after the Sample point (e.g. data bit, CRC (Cyclic Redundancy Check) bit, stuff bit, error flag or idle) is called the Information Processing Time (IPT).

The IPT is application specific but may not be longer than 2 tq; the IPT for the C_CAN is 0 tq. Its length is the lower limit of the programmed length of Phase_Seg2. In case of a synchronization, Phase_Seg2 may be shortened to a value less than IPT, which does not affect bus timing.

6.23.6.21 Calculating Bit Timing Parameters

Usually, the calculation of the bit timing configuration starts with a desired bit rate or bit time. The resulting bit time (1/bit rate) must be an integer multiple of the APB clock period.

The bit time may consist of 4 to 25 time quanta, the length of the time quantum tq is defined by the Baud Rate Prescaler with tq = (Baud Rate Prescaler)/fapb_clk. Several combinations may lead to the desired bit time, allowing iterations of the following steps.

First part of the bit time to be defined is the Prop_Seg. Its length depends on the delay time measured in the APB clock. A maximum bus length as well as a maximum node delay has to be defined for expandible CAN bus systems. The resulting time for Prop_Seg is converted into time quanta (rounded up to the nearest integer multiple of tq).

The Sync_Seg is 1 tq long (fixed), leaving (bit time – Prop_Seg – 1) tq for the two Phase Buffer Segments. If the number of remaining tq is even, the Phase Buffer Segments have the same length, Phase_Seg2 = Phase_Seg1, else Phase_Seg2 = Phase_Seg1 + 1.

The minimum nominal length of Phase_Seg2 has to be regarded as well. Phase_Seg2 may not be shorter than the IPT of the CAN controller, which, depending on the actual implementation, is in the range of [0..2] tq.

The length of the Synchronization Jump Width is set to its maximum value, which is the minimum of 4 and Phase_Seg1.

The oscillator tolerance range necessary for the resulting configuration is calculated by the formulas given in Section “Oscillator Tolerance Range”.

If more than one configuration is possible, that configuration allowing the highest oscillator tolerance range should be chosen.

CAN nodes with different system clocks require different configurations to come to the same bit rate. The calculation of the propagation time in the CAN network, based on the nodes with the longest delay time, is done once for the whole network.

The oscillator tolerance range of the CAN systems is limited by that node with the lowest tolerance range.

The calculation may shows that bus length or bit rate have to be decreased or that the stability of the oscillator frequency has to be increased in order to find a protocol compliant configuration of the CAN bit timing. The resulting configuration is written into the Bit Timing Register: (Phase_Seg2-1) & (Phase_Seg1+Prop_Seg-1) & (SynchronisationJumpWidth-1) & (Prescaler-1)

Example for Bit Timing at High Baud Rate

In this example, the frequency of APB_CLK is 10 MHz, BRP (CAN_BTIME[5:0]) is 0, and the bit rate is 1 MBit/s.

t_q	100 ns	$= t_{APB\_CLK}$
delay of bus driver	50ns	
delay of receiver circuit	30ns	
delay of bus line (40m)	220ns	
t_{Prop}	600ns	$= 6 \cdot t_q$
t_{SJW}	100ns	$= 1 \cdot t_q$
t_{TSeg1}	700ns	$= t_{Prop} + t_{SJW}$
t_{TSeg2}	200ns	= Information Processing Time + $1 \cdot t_q$
$t_{Sync\text{-}Seg}$	100ns	$= 1 \cdot t_q$

bit time 1000ns $= t_{\text{Sync-Seg}} + t_{\text{TSeg1}} + t_{\text{TSeg2}}$

tolerance for APB_CLK 0.39% $= \dfrac{Min(PB1, PB2)}{2 \times 13 \times (bit\ \ time - PB2))}$

$$= \frac{0.1us}{2 \times 13 \times (1us - 0.2us))}$$

In this example, the concatenated bit time parameters are $(2\text{-}1)_3$ & $(7\text{-}1)_4$ & $(1\text{-}1)_2$ & $(1\text{-}1)_6$, and the Bit Timing Register is programmed to 0x1600.

Note:

PB1/2: indicate the phase buffer segment 1/2

The subscript of $(2\text{-}1)_3$ indicates the number of bits in the corresponding bit of Bit Timing Register.

6.23.6.22 CAN Interface Reset State

After the hardware reset, the C_CAN registers hold the reset values which are given in the register description in 6.23.7.

Additionally the bus-off state is reset and the output CAN_TX is set to recessive (HIGH). The value 0x0001 (Init = ‘1’) in the CAN Control Register enables the software initialization. The C_CAN does not influence the CAN bus until the application software resets the Init bit (CAN_CON[0]) to ‘0’.

The data stored in the Message RAM is not affected by a hardware reset. After powered on, the contents of the Message RAM are undefined.

<u>Example for Bit Timing at Low Baud Rate</u>

In this example, the frequency of APB_CLK is 2 MHz, BRP (CAN_BTIME[5:0]) is 1, and the bit rate is 100 Kbit/s.

t_q 1 us $= 2 \cdot t_{\text{APB\_CLK}}$

delay of bus driver 200ns

delay of receiver circuit 80ns

delay of bus line (40m) 220ns

t_{Prop} 1us $= 1 \cdot t_q$

t_{SJW} 4us $= 4 \cdot t_q$

t_{TSeg1} 5us $= t_{\text{Prop}} + t_{\text{SJW}}$

t_{TSeg2} 4us = Information Processing Time + $3 \cdot t_q$

$t_{\text{Sync-Seg}}$ 1us $= 1 \cdot t_q$

bit time 10us $= t_{\text{Sync-Seg}} + t_{\text{TSeg1}} + t_{\text{TSeg2}}$

tolerance for APB_CLK 1.58 % $= \dfrac{Min(PB1, PB2)}{2 \times 13 \times (bit\ \ time - PB2))}$

$$= \frac{4us}{2 \times (13 \times (10us - 4us))}$$

In this example, the concatenated bit time parameters are $(4\text{-}1)_3$ & $(5\text{-}1)_4$ & $(4\text{-}1)_2$ & $(2\text{-}1)_6$, and the Bit Timing Register is programmed to 0x34C1.

6.23.7 Register Map

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

Register	Offset	R/W	Description	Reset Value
CAN Base Address: **CANx_BA = 0xB00A_0000 + 0x0000_1000*x** **x=0,1,2,3**				
CAN_CON	CANx_BA+0x00	R/W	CAN Control Register	0x0000_0001
CAN_STATUS	CANx_BA+0x04	R/W	CAN Status Register	0x0000_0000
CAN_ERR	CANx_BA+0x08	R	CAN Error Counter Register	0x0000_0000
CAN_BTIME	CANx_BA+0x0C	R/W	Bit Timing Register	0x0000_2301
CAN_IIDR	CANx_BA+0x10	R	Interrupt Identifier Register	0x0000_0000
CAN_TEST	CANx_BA+0x14	R/W	Test Register (Register Map Note 1)	0x0000_0080
CAN_BRPE	CANx_BA+0x18	R/W	Baud Rate Prescaler Extension Register	0x0000_0000
CAN_IFn_CREQ **n = 1,2**	CANx_BA+0x20 + (0x60 *(n-1))	R/W	IFn (Register Map Note 2) Command Request Registers	0x0000_0001
CAN_IFn_CMASK **n=1,2**	CANx_BA+0x24 + (0x60 *(n-1))	R/W	IFn Command Mask Registers	0x0000_0000
CAN_IFn_MASK1 **n=1,2**	CANx_BA+0x28 + (0x60 *(n-1))	R/W	IFn Mask 1 Registers	0x0000_FFFF
CAN_IFn_MASK2 **n=1,2**	CANx_BA+0x2C + (0x60 *(n-1))	R/W	IFn Mask 2 Registers	0x0000_FFFF
CAN_IFn_ARB1 **n=1,2**	CANx_BA+0x30 + (0x60 *(n-1))	R/W	IFn Arbitration 1 Registers	0x0000_0000
CAN_IFn_ARB2 **n=1,2**	CANx_BA+0x34 + (0x60 *(n-1))	R/W	IFn Arbitration 2 Registers	0x0000_0000
CAN_IFn_MCON **n=1,2**	CANx_BA+0x38 + (0x60 *(n-1))	R/W	IFn Message Control Registers	0x0000_0000
CAN_IFn_DAT_A1 **n=1,2**	CANx_BA+0x3C + (0x60 *(n-1))	R/W	IFn Data A1 Registers (Register Map Note 3)	0x0000_0000
CAN_IFn_DAT_A2 **n=1,2**	CANx_BA+0x40 + (0x60 *(n-1))	R/W	IFn Data A2 Registers (Register Map Note 3)	0x0000_0000
CAN_IFn_DAT_B1 **n=1,2**	CANx_BA+0x44 + (0x60 *(n-1))	R/W	IFn Data B1 Registers (Register Map Note 3)	0x0000_0000
CAN_IFn_DAT_B2 **n=1,2**	CANx_BA+0x48 + (0x60 *(n-1))	R/W	IFn Data B2 Registers (Register Map Note 3)	0x0000_0000
CAN_TXREQ1	CANx_BA+0x100	R	Transmission Request Register 1	0x0000_0000

CAN_TXREQ2	CANx_BA+0x104	R	Transmission Request Register 2	0x0000_0000
CAN_NDAT1	CANx_BA+0x120	R	New Data Register 1	0x0000_0000
CAN_NDAT2	CANx_BA+0x124	R	New Data Register 2	0x0000_0000
CAN_IPND1	CANx_BA+0x140	R	Interrupt Pending Register 1	0x0000_0000
CAN_IPND2	CANx_BA+0x144	R	Interrupt Pending Register 2	0x0000_0000
CAN_MVLD1	CANx_BA+0x160	R	Message Valid Register 1	0x0000_0000
CAN_MVLD2	CANx_BA+0x164	R	Message Valid Register 2	0x0000_0000
CAN_WU_EN	CANx_BA+0x168	R/W	Wake-up Enable Control Register	0x0000_0000
CAN_WU_STATUS	CANx_BA+0x16C	R/W	Wake-up Status Register	0x0000_0000

Note:

1. 0x00 & 0br0000000, where r signifies the actual value of the CAN_RX
2. IFn: The two sets of Message Interface Registers – IF1 and IF2, have identical function
3. An/Bn: The two sets of data registers – A1, A2 and B1, B2.
4. CAN_BA, where x = 0 or 1.

CAN Register Map for Each Bit Function

Addr Offset	Register Name	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
00h	CAN_CON	Reserved								Test	CCE	DAR	Res	EIE	SIE	IE	Init
04h	CAN_STATUS	Reserved								BOff	EWarn	EPass	RxOk	TxOk	LEC		
08h	CAN_ERR	RP	REC6-0							TEC7-0							
0Ch	CAN_BTIME	Res	TSeg2			TSeg1				SJW		BRP					
10h	CAN_IIDR	IntId15-8								IntId7-0							
14h	CAN_TEST	Reserved								Rx	Tx1	Tx0	LBack	Silent	Basic	Reserved	
18h	CAN_BRPE	Reserved												BRPE			
20h	CAN_IF1_CREQ	Busy	Reserved									Message Number					
24h	CAN_IF1_CMASK	Reserved								WR/RD	Mask	Arb	Control	ClrIntPnd	TxRqst/	Data A	Data B
28h	CAN_IF1_MASK1	Msk15-0															
2Ch	CAN_IF1_MASK2	MXtd	MDir	Res	Msk28-16												
30h	CAN_IF1_ARB1	ID15-0															
34h	CAN_IF1_ARB2	MsgVal	Xtd	Dir	ID28-16												

Addr Offset	Register Name	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
38h	CAN_IF1_MCON	NewDat	MsgLst	IntPnd	UMask	TxIE	RxIE	RmtEn	TxRqst	EoB	Reserved			DLC3-0			
3Ch	CAN_IF1_DAT_A1	Data(1)								Data(0)							
40h	CAN_IF1_DAT_A2	Data(3)								Data(2)							
44h	CAN_IF1_DAT_B1	Data(5)								Data(4)							
48h	CAN_IF1_DAT_B2	Data(7)								Data(6)							
80h	CAN_IF2_CREQ	Busy	Reserved									Message Number					
84h	CAN_IF2_CMASK	Reserved								WR/RD	Mask	Arb	Control	ClrIntPnd	TxRqst/	Data A	Data B
88h	CAN_IF2_MASK1	Msk15-0															
8Ch	CAN_IF2_MASK2	MXtd	MDir	Res.	Msk28-16												
90h	CAN_IF2_ARB1	ID15-0															
94h	CAN_IF2_ARB2	MsgVal	Xtd	Dir	ID28-16												
98h	CAN_IF2_MCON	NewDat	MsgLst	IntPnd	UMask	TxIE	RxIE	RmtEn	TxRqst	EoB	Reserved			DLC3-0			
9Ch	CAN_IF2_DAT_A1	Data(1)								Data(0)							

Addr Offset	Register Name	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A0h	CAN_IF2_DAT_A2	Data(3)								Data(2)							
A4h	CAN_IF2_DAT_B1	Data(5)								Data(4)							
A8h	CAN_IF2_DAT_B2	Data(7)								Data(6)							
100h	CAN_TXREQ1	TxRqst16-1															
104h	CAN_TXREQ2	TxRqst32-17															
120h	CAN_NDAT1	NewDat16-1															
124h	CAN_NDAT2	NewDat32-17															
140h	CAN_IPND1	IntPnd16-1															
144h	CAN_IPND2	IntPnd32-17															
160h	CAN_MVLD1	MsgVal16-1															
164h	CAN_MVLD2	MsgVal32-17															
168h	CAN_WU_EN	Reserved															WAKUP_EN
16Ch	CAN_WU_STATUS	Reserved															WAKUP_STS
170h	CAN_RAM_CEN	Reserved															RAM_CEN
Others	Reserved	Reserved															

Table 6.23-4 CAN Register Map for Each Bit Function

Note: Reserved bits are read as 0’ except for IFn Mask 2 Register where they are read as ’1’.

Res. = Reserved

6.23.8 Register Description

The C_CAN allocates an address space of 256 bytes. The registers are organized as 16-bit registers.

The two sets of interface registers (IF1 and IF2) control the software access to the Message RAM. They buffer the data to be transferred to and from the RAM, avoiding conflicts between software accesses and message reception/transmission.

CAN Control Register (CAN_CON)

Register	Offset	R/W	Description	Reset Value
CAN_CON	CANx_BA+0x00	R/W	CAN 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
Test	CCE	DAR	Reserved	EIE	SIE	IE	Init

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**Test**	**Test Mode Enable Bit** 0 = Normal Operation. 1 = Test Mode.
[6]	**CCE**	**Configuration Change Enable Bit** 0 = No write access to the Bit Timing Register. 1 = Write access to the Bit Timing Register (CAN_BTIME) allowed. (while Init bit (CAN_CON[0]) = 1).
[5]	**DAR**	**Automatic Re-transmission Disable Bit** 0 = Automatic Retransmission of disturbed messages Enabled. 1 = Automatic Retransmission Disabled.
[4]	**Reserved**	Reserved.
[3]	**EIE**	**Error Interrupt Enable Bit** 0 = Disabled - No Error Status Interrupt will be generated. 1 = Enabled - A change in the bits BOff (CAN_STATUS[7]) or EWarn (CAN_STATUS[6]) in the Status Register will generate an interrupt.
[2]	**SIE**	**Status Change Interrupt Enable Bit** 0 = Disabled - No Status Change Interrupt will be generated. 1 = Enabled - An interrupt will be generated when a message transfer is successfully completed or a CAN bus error is detected.
[1]	**IE**	**Module Interrupt Enable Bit** 0 = Funcrion interrupt Disabled. 1 = Funcrion interrupt Enabled.
[0]	**Init**	**Init Initialization** 0 = Normal Operation.

		1 = Initialization is started.

Note: The bus-off recovery sequence (see CAN Specification Rev. 2.0) cannot be shortened by setting or resetting the Init bit (CAN_CON[0]). If the device goes in the bus-off state, it will set Init of its own accord, stopping all bus activities. Once Init has been cleared by the CPU, the device will then wait for 129 occurrences of Bus Idle (129 * 11 consecutive recessive bits) before resuming normal operations. At the end of the bus-off recovery sequence, the Error Management Counters will be reset.

During the waiting time after resetting Init, each time a sequence of 11 recessive bits has been monitored, a Bit0Error code is written to the Status Register, enabling the CPU to readily check up whether the CAN bus is stuck at dominant or continuously disturbed and to monitor the proceeding of the bus-off recovery sequence.

CAN Status Register (CAN_STATUS)

Register	Offset	R/W	Description	Reset Value
CAN_STATUS	CANx_BA+0x04	R/W	CAN 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**
BOff	**EWarn**	**EPass**	**RxOK**	**TxOK**	**LEC**		

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**BOff**	**Bus-off Status (Read Only)** 0 = The CAN module is not in bus-off state. 1 = The CAN module is in bus-off state.
[6]	**EWarn**	**Error Warning Status (Read Only)** 0 = Both error counters are below the error warning limit of 96. 1 = At least one of the error counters in the EML has reached the error warning limit of 96.
[5]	**EPass**	**Error Passive (Read Only)** 0 = The CAN Core is error active. 1 = The CAN Core is in the error passive state as defined in the CAN Specification.
[4]	**RxOK**	**Received a Message Successfully** 0 = No message has been successfully received since this bit was last reset by the CPU. This bit is never reset by the CAN Core. 1 = A message has been successfully received since this bit was last reset by the CPU (independent of the result of acceptance filtering).
[3]	**TxOK**	**Transmitted a Message Successfully** 0 = Since this bit was reset by the CPU, no message has been successfully transmitted. This bit is never reset by the CAN Core. 1 = Since this bit was last reset by the CPU, a message has been successfully (error free and acknowledged by at least one other node) transmitted.
[2:0]	**LEC**	**Last Error Code (Type of the Last Error to Occur on the CAN Bus)** The LEC field holds a code, which indicates the type of the last error to occur on the CAN bus. This field will be cleared to '0' when a message has been transferred (reception or transmission) without error. The unused code '7' may be written by the CPU to check for updates. Table 6.23-5 Last Error Codedescribes the error code.

Error Code	Meanings
0	No Error
1	Stuff Error: More than 5 equal bits in a sequence have occurred in a part of a received message where this is not allowed.
2	Form Error: A fixed format part of a received frame has the wrong format.
3	AckError: The message this CAN Core transmitted was not acknowledged by another node.
4	Bit1Error: During the transmission of a message (with the exception of the arbitration field), the device wanted to send a recessive level (bit of logical value '1'), but the monitored bus value was dominant.
5	Bit0Error: During the transmission of a message (or acknowledge bit, or active error flag, or overload flag), though the device wanted to send a dominant level (data or identifier bit logical value '0'), but the monitored Bus value was recessive. During bus-off recovery, this status is set each time a sequence of 11 recessive bits has been monitored. This enables the CPU to monitor the proceedings of the bus-off recovery sequence (indicating the bus is not stuck at dominant or continuously disturbed).
6	CRCError: The CRC check sum was incorrect in the message received, the CRC received for an incoming message does not match with the calculated CRC for the received data.
7	Unused: When the LEC shows the value '7', no CAN bus event was detected since the CPU wrote this value to the LEC.

Table 6.23-5 Last Error Code

Status Interrupts

A Status Interrupt is generated by bits BOff (CAN_STATUS[7]) and EWarn (CAN_STATUS[6]) (Error Interrupt) or by RxOk (CAN_STATUS[4]), TxOk (CAN_STATUS[3]) and LEC (CAN_STATUS[2:0]) (Status Change Interrupt) assumed that the corresponding enable bits in the CAN Control Register are set. A change of bit EPass (CAN_STATUS[5]) or a write to RxOk, TxOk or LEC will never generate a Status Interrupt.

Reading the Status Register will clear the Status Interrupt value (8000h) in the Interrupt Register, if it is pending.

CAN Error Counter Register (CAN_ERR)

Register	Offset	R/W	Description	Reset Value
CAN_ERR	CANx_BA+0x08	R	CAN Error 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**
RP	REC						
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TEC							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**RP**	**Receive Error Passive** 0 = The Receive Error Counter is below the error passive level. 1 = The Receive Error Counter has reached the error passive level as defined in the CAN Specification.
[14:8]	**REC**	**Receive Error Counter** Actual state of the Receive Error Counter. Values between 0 and 127.
[7:0]	**TEC**	**Transmit Error Counter** Actual state of the Transmit Error Counter. Values between 0 and 255.

Bit Timing Register (CAN_BTIME)

Register	Offset	R/W	Description	Reset Value
CAN_BTIME	CANx_BA+0x0C	R/W	Bit Timing Register	0x0000_2301

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	**TSeg2**			**TSeg1**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SJW		**BRP**					

Bits	Description	
[31:15]	**Reserved**	Reserved.
[14:12]	**TSeg2**	**Time Segment After Sample Point** 0x0-0x7: Valid values for TSeg2 are [0…7]. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used.
[11:8]	**TSeg1**	**Time Segment Before the Sample Point Minus Sync_Seg** 0x01-0x0F: valid values for TSeg1 are [1…15]. The actual interpretation by the hardware of this value is such that one more than the value programmed is used.
[7:6]	**SJW**	**(Re)Synchronization Jump Width** 0x0-0x3: Valid programmed values are [0…3]. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used.
[5:0]	**BRP**	**Baud Rate Prescaler** 0x01-0x3F: The value by which the oscillator frequency is divided for generating the bit time quanta. The bit time is built up from a multiple of this quanta. Valid values for the Baud Rate Prescaler are [0…63]. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used.

Note: With a module clock APB_CLK of 8 MHz, the reset value of 0x2301 configures the C_CAN for a bit rate of 500 Kbit/s. The registers are only writable if bits CCE (CAN_CON[6]) and Init (CAN_CON[0]) are set.

Interrupt Identify Register (CAN_IIDR)

Register	Offset	R/W	Description	Reset Value
CAN_IIDR	CANx_BA+0x10	R	Interrupt Identifier Register	0x0000_0000

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

Bits	Description	
[15:0]	**IntId**	**Interrupt Identifier (Indicates the Source of the Interrupt)** If several interrupts are pending, the CAN Interrupt Register will point to the pending interrupt with the highest priority, disregarding their chronological order. An interrupt remains pending until the application software has cleared it. If IntId is different from 0x0000 and IE (CAN_CON[1]) is set, the IRQ interrupt signal to the EIC is active. The interrupt remains active until IntId is back to value 0x0000 (the cause of the interrupt is reset) or until IE is reset. The Status Interrupt has the highest priority. Among the message interrupts, the Message Object' s interrupt priority decreases with increasing message number. A message interrupt is cleared by clearing the Message Object's IntPnd bit (CAN_IFn_MCON[13]). The Status Interrupt is cleared by reading the Status Register.

IntId Value	Meanings
0x0000	No Interrupt is Pending
0x0001-0x0020	Number of Message Object which caused the interrupt.
0x0021-0x7FFF	Unused
0x8000	Status Interrupt
0x8001-0xFFFF	Unused

Table 6.23-6 Source of Interrupts

Test Register (CAN_TEST)

Register	Offset	R/W	Description	Reset Value
CAN_TEST	CANx_BA+0x14	R/W	Test Register (Register Map Note 1)	0x0000_0080

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**Rx**	**Monitors the Actual Value of CAN_RX Pin (Read Only) *(1)** 0 = The CAN bus is dominant (CAN_RX = ‘0’). 1 = The CAN bus is recessive (CAN_RX = ‘1’).
[6:5]	**Tx**	**Tx[1:0]: Control of CAN_TX Pin** 00 = Reset value, CAN_TX pin is controlled by the CAN Core. 01 = Sample Point can be monitored at CAN_TX pin. 10 = CAN_TX pin drives a dominant (‘0’) value. 11 = CAN_TX pin drives a recessive (‘1’) value.
[4]	**LBack**	**Loop Back Mode Enable Bit** 0 = Loop Back Mode Disabled. 1 = Loop Back Mode Enabled.
[3]	**Silent**	**Silent Mode** 0 = Normal operation. 1 = The module is in Silent Mode.
[2]	**Basic**	**Basic Mode** 0 = Basic Mode Disabled. 1= IF1 Registers used as Tx Buffer, IF2 Registers used as Rx Buffer.
[1:0]	**Reserved**	Reserved.

Reset value: 0000 0000 R000 0000 b (R:current value of RX pin)

Note: Write access to the Test Register is enabled by setting the Test bit (CAN_CON[7]). The different test functions may be combined, but Tx[1-0] “00” (CAN_TEST[6:5]) disturbs message transfer.

Baud Rate Prescaler Extension REGISTER (CAN_BRPE)

Register	Offset	R/W	Description	Reset Value
CAN_BRPE	CANx_BA+0x18	R/W	Baud Rate Prescaler Extension Register	0x0000_0000

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

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3:0]	**BRPE**	**BRPE: Baud Rate Prescaler Extension** 0x00-0x0F: By programming BRPE, the Baud Rate Prescaler can be extended to values up to 1023. The actual interpretation by the hardware is that one more than the value programmed by BRPE (MSBs) and BTIME (LSBs) is used.

Message Interface Register Sets

There are two sets of Interface Registers, which are used to control the CPU access to the Message RAM. The Interface Registers avoid conflict between the CPU accesses to the Message RAM and CAN message reception and transmission by buffering the data to be transferred. A complete Message Object or parts of the Message Object may be transferred between the Message RAM and the IFn Message Buffer registers in one single transfer.

The function of the two interface register sets is identical except for the Basic test mode. They can be used the way one set of registers is used for data transfer to the Message RAM while the other set of registers is used for the data transfer from the Message RAM, allowing both processes to be interrupted by each other. Table 6.23-7 provides an overview of the two Interface Register sets.

Each set of Interface Registers consists of Message Buffer Registers controlled by their own Command Registers. The Command Mask Register specifies the direction of the data transfer and which parts of a Message Object will be transferred. The Command Request Register is used to select a Message Object in the Message RAM as target or source for the transfer and to start the action specified in the Command Mask Register.

Address	IF1 Register Set	Address	IF2 Register Set
CAN_BA+0x20	IF1 Command Request	CAN_BA+0x80	IF2 Command Request
CAN_BA+0x24	IF1 Command Mask	CAN_BA+0x84	IF2 Command Mask
CAN_BA+0x28	IF1 Mask 1	CAN_BA+0x88	IF2 Mask 1
CAN_BA+0x2C	IF1 Mask 2	CAN_BA+0x8C	IF2 Mask 2
CAN_BA+0x30	IF1 Arbitration 1	CAN_BA+0x90	IF2 Arbitration 1
CAN_BA+0x34	IF1 Arbitration 2	CAN_BA+0x94	IF2 Arbitration 2
CAN_BA+0x38	IF1 Message Control	CAN_BA+0x98	IF2 Message Control
CAN_BA+0x3C	IF1 Data A 1	CAN_BA+0x9C	IF2 Data A 1
CAN_BA+0x40	IF1 Data A 2	CAN_BA+0xA0	IF2 Data A 2
CAN_BA+0x44	IF1 Data B 1	CAN_BA+0xA4	IF2 Data B 1
CAN_BA+0x48	IF1 Data B 2	CAN_BA+0xA8	IF2 Data B 2

Table 6.23-7 IF1 and IF2 Message Interface Register

IFn Command Request Register (CAN_IFn_CREQ)

Register	Offset	R/W	Description	Reset Value
CAN_IFn_CREQ	CANx_BA+0x20 + (0x60 *(n-1))	R/W	IFn (Register Map Note 2) Command Request Registers	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**
Busy	**Reserved**						
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**Message Number**					

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**Busy**	**Busy Flag** 0 = Read/write action has finished. 1 = Writing to the IFn Command Request Register is in progress. This bit can only be read by the software.
[14:6]	**Reserved**	Reserved.
[5:0]	**Message Number**	**Message Number** 0x01-0x20: Valid Message Number, the Message Object in the Message RAM is selected for data transfer. 0x00: Not a valid Message Number, interpreted as 0x20. 0x21-0x3F: Not a valid Message Number, interpreted as 0x01-0x1F.

A message transfer is started as soon as the application software has written the message number to the Command Request Register. With this write operation, the Busy bit (CAN_IFn_CREQ[15]) is automatically set to notify the CPU that a transfer is in progress. After a waiting time of 3 to 6 APB_CLK periods, the transfer between the Interface Register and the Message RAM is completed. The Busy bit is cleared.

Note: When a Message Number that is not valid is written into the Command Request Register, the Message Number will be transformed into a valid value and that Message Object will be transferred.

NUC980 SERIES TECHNICAL REFERENCE MANUAL

IFn Command Mask Register (CAN_IFn_CMASK)

The control bits of the IFn Command Mask Register specify the transfer direction and select which of the IFn Message Buffer Registers are source or target of the data transfer.

Register	Offset	R/W	Description	Reset Value
CAN_IFn_CMASK	CANx_BA+0x24 + (0x60 *(n-1))	R/W	IFn Command Mask 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							
7	6	5	4	3	2	1	0
WR/RD	Mask	Arb	Control	ClrIntPnd	TxRqst/NewDat	DAT_A	DAT_B

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**WR/RD**	**Write / Read Mode** 0 = Read: Transfer data from the Message Object addressed by the Command Request Register into the selected Message Buffer Registers. 1 = Write: Transfer data from the selected Message Buffer Registers to the Message Object addressed by the Command Request Register.
[6]	**Mask**	**Access Mask Bits** Write Operation: 0 = Mask bits unchanged. 1 = Transfer Identifier Mask + MDir + MXtd to Message Object. Read Operation: 0 = Mask bits unchanged. 1 = Transfer Identifier Mask + MDir + MXtd to IFn Message Buffer Register.
[5]	**Arb**	**Access Arbitration Bits** Write Operation: 0 = Arbitration bits unchanged. 1 = Transfer Identifier + Dir (CAN_IFn_ARB2[13]) + Xtd (CAN_IFn_ARB2[14]) + MsgVal (CAN_IFn_ARB2[15]) to Message Object. Read Operation: 0 = Arbitration bits unchanged. 1 = Transfer Identifier + Dir + Xtd + MsgVal to IFn Message Buffer Register.
[4]	**Control**	**Control Access Control Bits** Write Operation: 0 = Control Bits unchanged.

		1 = Transfer Control Bits to Message Object. Read Operation: 0 = Control Bits unchanged. 1 = Transfer Control Bits to IFn Message Buffer Register.
[3]	**ClrIntPnd**	**Clear Interrupt Pending Bit** Write Operation: When writing to a Message Object, this bit is ignored. Read Operation: 0 = IntPnd bit (CAN_IFn_MCON[13]) remains unchanged. 1 = Clear IntPnd bit in the Message Object.
[2]	**TxRqst/NewDat**	**Access Transmission Request Bit When Write Operation** 0 = TxRqst bit unchanged. 1 = Set TxRqst bit. **Note:** If a transmission is requested by programming bit TxRqst/NewDat in the IFn Command Mask Register, bit TxRqst in the IFn Message Control Register will be ignored. Access New Data Bit when Read Operation. 0 = NewDat bit remains unchanged. 1 = Clear NewDat bit in the Message Object. **Note:** A read access to a Message Object can be combined with the reset of the control bits IntPnd and NewDat. The values of these bits transferred to the IFn Message Control Register always reflect the status before resetting these bits.
[1]	**DAT_A**	**Access Data Bytes [3:0]** Write Operation: 0 = Data Bytes [3:0] unchanged. 1 = Transfer Data Bytes [3:0] to Message Object. Read Operation: 0 = Data Bytes [3:0] unchanged. 1 = Transfer Data Bytes [3:0] to IFn Message Buffer Register.
[0]	**DAT_B**	**Access Data Bytes [7:4]** Write Operation: 0 = Data Bytes [7:4] unchanged. 1 = Transfer Data Bytes [7:4] to Message Object. Read Operation: 0 = Data Bytes [7:4] unchanged. 1 = Transfer Data Bytes [7:4] to IFn Message Buffer Register.

IFn Mask 1 Register (CAN_IFn_MASK1)

Register	Offset	R/W	Description	Reset Value
CAN_IFn_MASK1	CANx_BA+0x28 + (0x60 *(n-1))	R/W	IFn Mask 1 Registers	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**
Msk							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Msk							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**Msk**	**Identifier Mask 15-0** 0 = The corresponding bit in the identifier of the message object cannot inhibit the match in the acceptance filtering. 1 = The corresponding identifier bit is used for acceptance filtering.

IFn Mask 2 Register (CAN_IFn_MASK2)

Register	Offset	R/W	Description	Reset Value
CAN_IFn_MASK2	CANx_BA+0x2C + (0x60 *(n-1))	R/W	IFn Mask 2 Registers	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
MXtd	MDir	Reserved	Msk				
7	6	5	4	3	2	1	0
Msk							

Bits	Description	
[31:16]	Reserved	Reserved.
[15]	MXtd	**Mask Extended Identifier** 0 = The extended identifier bit (IDE) has no effect on the acceptance filtering. 1 = The extended identifier bit (IDE) is used for acceptance filtering. **Note:** When 11-bit (“standard”) Identifiers are used for a Message Object, the identifiers of received Data Frames are written into bits ID28 to ID18 (CAN_IFn_ARB2[12:2]). For acceptance filtering, only these bits together with mask bits Msk28 to Msk18 (CAN_IFn_MASK2[12:2]) are considered.
[14]	MDir	**Mask Message Direction** 0 = The message direction bit (Dir (CAN_IFn_ARB2[13])) has no effect on the acceptance filtering. 1 = The message direction bit (Dir) is used for acceptance filtering.
[13]	Reserved	Reserved.
[12:0]	Msk	**Identifier Mask 28-16** 0 = The corresponding bit in the identifier of the message object cannot inhibit the match in the acceptance filtering. 1 = The corresponding identifier bit is used for acceptance filtering.

IFn Arbitration 1 Register (CAN_IFn_ARB1)

Register	Offset	R/W	Description	Reset Value
CAN_IFn_ARB1	CANx_BA+0x30 + (0x60 *(n-1))	R/W	IFn Arbitration 1 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**
ID							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ID							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**ID**	**Message Identifier 15-0** ID28 - ID0, 29-bit Identifier (“Extended Frame”). ID28 - ID18, 11-bit Identifier (“Standard Frame”)

IFn Arbitration 2 Register (CAN_IFn_ARB2)

Register	Offset	R/W	Description	Reset Value
CAN_IFn_ARB2	CANx_BA+0x34 + (0x60 *(n-1))	R/W	IFn Arbitration 2 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**
MsgVal	**Xtd**	**Dir**	**ID**				
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ID							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**MsgVal**	**Message Valid** 0 = The Message Object is ignored by the Message Handler. 1 = The Message Object is configured and should be considered by the Message Handler. **Note:** The application software must reset the MsgVal bit of all unused Messages Objects during the initialization before it resets bit Init (CAN_CON[0]). This bit must also be reset before the identifier Id28-0 (CAN_IFn_ARB1/2), the control bits Xtd (CAN_IFn_ARB2[14]), Dir (CAN_IFn_ARB2[13]), or the Data Length Code DLC3-0 (CAN_IFn_MCON[3:0]) are modified, or if the Messages Object is no longer required.
[14]	**Xtd**	**Extended Identifier** 0 = The 11-bit (“standard”) Identifier will be used for this Message Object. 1 = The 29-bit (“extended”) Identifier will be used for this Message Object.
[13]	**Dir**	**Message Direction** 0 = Direction is receive. On TxRqst, a Remote Frame with the identifier of this Message Object is transmitted. On reception of a Data Frame with matching identifier, that message is stored in this Message Object. 1 = Direction is transmit. On TxRqst, the respective Message Object is transmitted as a Data Frame. On reception of a Remote Frame with matching identifier, the TxRqst bit (CAN_IFn_CMASK[2]) of this Message Object is set (if RmtEn (CAN_IFn_MCON[9]) = one).
[12:0]	**ID**	**Message Identifier 28-16** ID28 - ID0, 29-bit Identifier (“Extended Frame”). ID28 - ID18, 11-bit Identifier (“Standard Frame”)

IFn Message Control Register (CAN_IFn_MCON)

Register	Offset	R/W	Description	Reset Value
CAN_IFn_MCON	CANx_BA+0x38 + (0x60 *(n-1))	R/W	IFn Message Control 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**
NewDat	**MsgLst**	**IntPnd**	**UMask**	**TxIE**	**RxIE**	**RmtEn**	**TxRqst**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EoB	**Reserved**			**DLC**			

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**NewDat**	**New Data** 0 = No new data has been written into the data portion of this Message Object by the Message Handler since last time this flag was cleared by the application software. 1 = The Message Handler or the application software has written new data into the data portion of this Message Object.
[14]	**MsgLst**	**Message Lost** 0 = No message lost since last time this bit was reset by the CPU. 1 = The Message Handler stored a new message into this object when NewDat was still set, the CPU has lost a message. **Note:** Only valid for Message Objects with direction = receive.
[13]	**IntPnd**	**Interrupt Pending** 0 = This message object is not the source of an interrupt. 1 = This message object is the source of an interrupt. The Interrupt Identifier in the Interrupt Register will point to this message object if there is no other interrupt source with higher priority.
[12]	**UMask**	**Use Acceptance Mask** 0 = Mask ignored. 1 = Use Mask (Msk28-0, MXtd, and MDir) for acceptance filtering. **Note:** If the UMask bit is set to one, the Message Object's mask bits have to be programmed during initialization of the Message Object before MsgVal bit (CAN_IFn_ARB2[15]) is set to one.
[11]	**TxIE**	**Transmit Interrupt Enable Bit** 0 = IntPnd (CAN_IFn_MCON[13]) will be left unchanged after the successful transmission of a frame. 1 = IntPnd will be set after a successful transmission of a frame.
[10]	**RxIE**	**Receive Interrupt Enable Bit** 0 = IntPnd (CAN_IFn_MCON[13]) will be left unchanged after a successful reception of a frame.

		1 = IntPnd will be set after a successful reception of a frame.
[9]	**RmtEn**	**Remote Enable Bit** 0 = At the reception of a Remote Frame, TxRqst (CAN_IFn_MCON[8]) is left unchanged. 1 = At the reception of a Remote Frame, TxRqst is set.
[8]	**TxRqst**	**Transmit Request** 0 = This Message Object is not waiting for transmission. 1 = The transmission of this Message Object is requested and is not yet done.
[7]	**EoB**	**End of Buffer** 0 = Message Object belongs to a FIFO Buffer and is not the last Message Object of that FIFO Buffer. 1 = Single Message Object or last Message Object of a FIFO Buffer. **Note:** This bit is used to concatenate two or more Message Objects (up to 32) to build a FIFO Buffer. For single Message Objects (not belonging to a FIFO Buffer), this bit must always be set to one.
[6:4]	**Reserved**	Reserved.
[3:0]	**DLC**	**Data Length Code** 0-8: Data Frame has 0-8 data bytes. 9-15: Data Frame has 8 data bytes **Note:** The Data Length Code of a Message Object must be defined the same as in all the corresponding objects with the same identifier at other nodes. When the Message Handler stores a data frame, it will write the DLC to the value given by the received message. Data(0): 1st data byte of a CAN Data Frame Data(1): 2nd data byte of a CAN Data Frame Data(2): 3rd data byte of a CAN Data Frame Data(3): 4th data byte of a CAN Data Frame Data(4): 5th data byte of a CAN Data Frame Data(5): 6th data byte of a CAN Data Frame Data(6): 7th data byte of a CAN Data Frame Data(7): 8th data byte of a CAN Data Frame **Note:** The Data(0) byte is the first data byte shifted into the shift register of the CAN Core during a reception while the Data(7) byte is the last. When the Message Handler stores a Data Frame, it will write all the eight data bytes into a Message Object. If the Data Length Code is less than 8, the remaining bytes of the Message Object will be overwritten by unspecified values.

IFn Data A1 Register (CAN_IFn_DAT_A1)

Register	Offset	R/W	Description	Reset Value
CAN_IFn_DAT_A1	CANx_BA+0x3C + (0x60 *(n-1))	R/W	IFn Data A1 Registers (Register Map Note 3)	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**Data(1)**	**Data Byte 1** 2nd data byte of a CAN Data Frame
[7:0]	**Data(0)**	**Data Byte 0** 1st data byte of a CAN Data Frame

IFn Data A2 Register (CAN_IFn_DAT_A2)

Register	Offset	R/W	Description	Reset Value
CAN_IFn_DAT_A2	CANx_BA+0x40 + (0x60 *(n-1))	R/W	IFn Data A2 Registers (Register Map Note 3)	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**Data(3)**	**Data Byte 3** 4th data byte of CAN Data Frame
[7:0]	**Data(2)**	**Data Byte 2** 3rd data byte of CAN Data Frame

IFn Data B1 Register (CAN_IFn_DAT_B1)

Register	Offset	R/W	Description	Reset Value
CAN_IFn_DAT_B1	CANx_BA+0x44 + (0x60 *(n-1))	R/W	IFn Data B1 Registers (Register Map Note 3)	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**Data(5)**	**Data Byte 5** 6th data byte of CAN Data Frame
[7:0]	**Data(4)**	**Data Byte 4** 5th data byte of CAN Data Frame

IFn Data B2 Register (CAN_IFn_DAT_B2)

Register	Offset	R/W	Description	Reset Value
CAN_IFn_DAT_B2	CANx_BA+0x48 + (0x60 *(n-1))	R/W	IFn Data B2 Registers (Register Map Note 3)	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**Data(7)**	**Data Byte 7** 8th data byte of CAN Data Frame.
[7:0]	**Data(6)**	**Data Byte 6** 7th data byte of CAN Data Frame.

In a CAN Data Frame, Data [0] is the first, Data [7] is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first.

Message Object in the Message Memory

There are 32 Message Objects in the Message RAM. To avoid conflicts between application software access to the Message RAM and CAN message reception and transmission, the CPU cannot directly access the Message Objects, these accesses are handled through the IFn Interface Registers. Table 6.23-8 provides an overview of the structures of a Message Object.

Message Object												
UMask	Msk [28:0]	MXtd	MDir	EoB	NewDat		MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
MsgVal	ID [28:0]	Xtd	Dir	DLC [3:0]	Data(0)	Data(1)	Data(2)	Data(3)	Data(4)	Data(5)	Data(6)	Data(7)

Table 6.23-8 Structure of a Message Object in the Message Memory

The Arbitration Registers ID28-0 (CAN_IFn_ARB1/2), Xtd (CAN_IFn_ARB2[14]) and Dir (CAN_IFn_ARB2[13]) are used to define the identifier and type of outgoing messages and are used (together with the mask registers Msk28-0 (CAN_IFn_MASK1/2), MXtd (CAN_IFn_MASK2[15]) and MDir (CAN_IFn_MASK2[14])) for acceptance filtering of incoming messages. A received message is stored in the valid Message Object with matching identifier and Direction = receive (Data Frame) or Direction = transmit (Remote Frame). Extended frames can be stored only in Message Objects with Xtd = one, standard frames in Message Objects with Xtd = zero. If a received message (Data Frame or Remote Frame) matches with more than one valid Message Object, it is stored into that with the lowest message number.

Message Handler Registers

All Message Handler registers are read only. Their contents (TxRqst (CAN_IFn_MCON[8]), NewDat (CAN_IFn_MCON[15]), IntPnd (CAN_IFn_MCON[13]) and MsgVal (CAN_IFn_ARB2[15]) bits of each Message Object and the Interrupt Identifier) are status information provided by the Message Handler FSM.

Transmission Request Register 1 (CAN_TXREQ1)

These registers hold the TxRqst bits of the 32 Message Objects. By reading the TxRqst bits, the software can check which Message Object in a Transmission Request is pending. The TxRqst bit of a specific Message Object can be set/reset by the application software through the IFn Message Interface Registers or by the Message Handler after reception of a Remote Frame or after a successful transmission.

Register	Offset	R/W	Description	Reset Value
CAN_TXREQ1	CANx_BA+0x100	R	Transmission Request 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
TxRqst16-9							
7	6	5	4	3	2	1	0
TxRqst8-1							

Bits	Description	
[31:16]	Reserved	Reserved.
[15:0]	TxRqst16-1	**Transmission Request Bits 16-1 (of All Message Objects) (Read Only)** 0 = This Message Object is not waiting for transmission. 1 = The transmission of this Message Object is requested and is not yet done.

Transmission Request Register 2 (CAN_TXREQ2)

Register	Offset	R/W	Description	Reset Value
CAN_TXREQ2	CANx_BA+0x104	R	Transmission Request 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**
TxRqst32-25							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TxRqst24-17							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**TxRqst32-17**	**Transmission Request Bits 32-17 (of All Message Objects) (Read Only)** 0 = This Message Object is not waiting for transmission. 1 = The transmission of this Message Object is requested and is not yet done.

New Data Register 1 (CAN_NDAT1)

These registers hold the NewDat bits of the 32 Message Objects. By reading out the NewDat bits, the software can check for which Message Object the data portion was updated. The NewDat bit of a specific Message Object can be set/reset by the software through the IFn Message Interface Registers or by the Message Handler after reception of a Data Frame or after a successful transmission.

Register	Offset	R/W	Description	Reset Value
CAN_NDAT1	CANx_BA+0x120	R	New Data 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**
NewData16-9							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NewData8-1							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**NewData16-1**	**New Data Bits 16-1 (of All Message Objects)** 0 = No new data has been written into the data portion of this Message Object by the Message Handler since the last time this flag was cleared by the application software. 1 = The Message Handler or the application software has written new data into the data portion of this Message Object.

New Data Register 2 (CAN_NDAT2)

Register	Offset	R/W	Description	Reset Value
CAN_NDAT2	CANx_BA+0x124	R	New Data 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**
NewData32-25							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NewData24-17							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**NewData32-17**	**New Data Bits 32-17 (of All Message Objects)** 0 = No new data has been written into the data portion of this Message Object by the Message Handler since the last time this flag was cleared by the application software. 1 = The Message Handler or the application software has written new data into the data portion of this Message Object.

Interrupt Pending Register 1 (CAN_IPND1)

These registers contain the IntPnd bits of the 32 Message Objects. By reading the IntPnd bits, the software can check for which Message Object an interrupt is pending. The IntPnd bit of a specific Message Object can be set/reset by the application software through the IFn Message Interface Registers or by the Message Handler after reception or after a successful transmission of a frame. This will also affect the value of IntId in the Interrupt Register.

Register	Offset	R/W	Description	Reset Value
CAN_IPND1	CANx_BA+0x140	R	Interrupt Pending 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**
IntPnd16-9							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IntPnd8-1							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**IntPnd16-1**	**Interrupt Pending Bits 16-1 (of All Message Objects)** 0 = This message object is not the source of an interrupt. 1 = This message object is the source of an interrupt.

Interrupt Pending Register 2 (CAN_IPND2)

Register	Offset	R/W	Description	Reset Value
CAN_IPND2	CANx_BA+0x144	R	Interrupt Pending 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**
IntPnd32-25							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IntPnd24-17							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**IntPnd32-17**	**Interrupt Pending Bits 32-17 (of All Message Objects)** 0 = This message object is not the source of an interrupt. 1 = This message object is the source of an interrupt.

Message Valid Register 1 (CAN_MVLD1)

These registers hold the MsgVal bits of the 32 Message Objects. By reading the MsgVal bits, the application software can check which Message Object is valid. The MsgVal bit of a specific Message Object can be set/reset by the application software via the IFn Message Interface Registers.

Register	Offset	R/W	Description	Reset Value
CAN_MVLD1	CANx_BA+0x160	R	Message Valid 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**
MsgVal16- 9							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MsgVal8-1							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**MsgVal16-1**	**Message Valid Bits 16-1 (of All Message Objects) (Read Only)** 0 = This Message Object is ignored by the Message Handler. 1 = This Message Object is configured and should be considered by the Message Handler. **Note:** CAN_MVLD1[0] means Message object No.1 is valid or not. If CAN_MVLD1[0] is set, message object No.1 is configured.

Message Valid Register 2 (CAN_MVLD2)

Register	Offset	R/W	Description	Reset Value
CAN_MVLD2	CANx_BA+0x164	R	Message Valid 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**
MsgVal32-25							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MsgVal24-17							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**MsgVal32-17**	**Message Valid Bits 32-17 (of All Message Objects) (Read Only)** 0 = This Message Object is ignored by the Message Handler. 1 = This Message Object is configured and should be considered by the Message Handler. **Note:** CAN_MVLD2[15] means Message object No.32 is valid or not. If CAN_MVLD2[15] is set, message object No.32 is configured.

Wake-up Enable Control Register (CAN_WU_EN)

Register	Offset	R/W	Description	Reset Value
CAN_WU_EN	CANx_BA+0x168	R/W	Wake-up 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							**WAKUP_EN**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**WAKUP_EN**	**Wake-up Enable Bit** 0 = The wake-up function Disabled. 1 = The wake-up function Enabled. **Note:** User can wake up system when there is a falling edge in the CAN_Rx pin.

Wake-up Status Register (CAN_WU_STATUS)

Register	Offset	R/W	Description	Reset Value
CAN_WU_STATUS	CANx_BA+0x16C	R/W	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							**WAKUP_STS**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**WAKUP_STS**	**Wake-up Status** 0 = No wake-up event occurred. 1 = Wake-up event occurred. **Note:** This bit can be cleared by writing ‘0’ to it.

6.24 Flash Memory Interface (FMI)

6.24.1 Overview

The Flash Memory Interface (FMI) in this chip has DMA unit and FMI unit. The DMA unit provides a DMA (Direct Memory Access) function for FMI to exchange data between system memory (ex. SDRAM) and shared buffer (128 bytes), and the FMI unit control the interface of SD0/eMMC0 or NAND Flash. The interface controller can support SD0/eMMC0 and NAND-type Flash and the FMI is cooperated with DMAC to provide a fast data transfer between system memory and cards.

6.24.2 Features

- Supports single DMA channel and address in non-word boundary.
- Supports hardware Scatter-Gather function.
- Supports 128Bytes shared buffer for data exchange between system memory and Flash device. (Separate into two 64 bytes ping-pong FIFO).
- Supports SD0/eMMC0 Flash device.
- Supports SLC and MLC NAND type Flash.
- Adjustable NAND page sizes. (2048B+spare area, 4096B+spare area and 8192B+spare area).
- Supports up to 8bit/12bit/24bit hardware ECC calculation circuit to protect data communication.
- Supports programmable NAND timing cycle.

6.24.3 Block Diagram

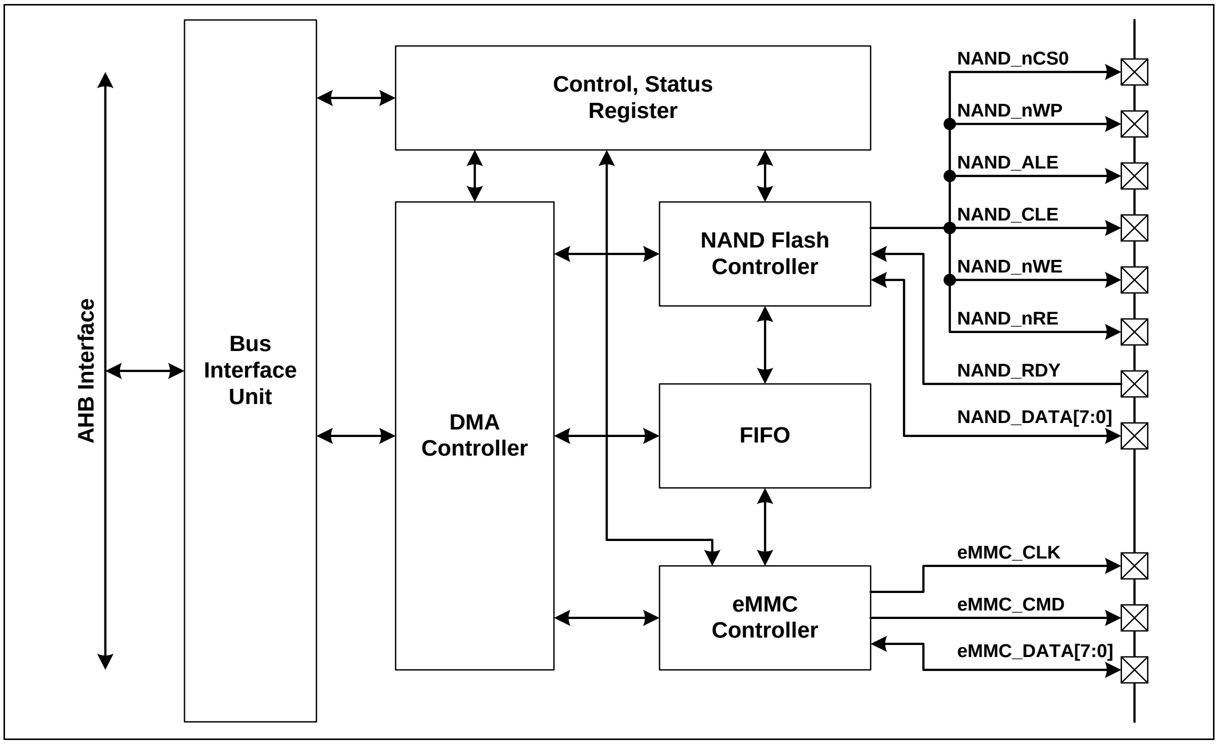


Figure 6.24-1 FMI Block Diagram

6.24.4 Basic Configuration

6.24.4.1 SD0 Basic Configuration

- Clock source Configuration
  - Select the source clock divide number for clock divider of APLL and UPLL on eMMC_SDIV (CLK_DIVCTL3[2:0]).
  - Select the source of SD0 engine clock on eMMC_S (CLK_DIVCTL3[4:3]).
  - Select the clock divider number of SD0 engine clock on eMMC_N (CLK_DIVCTL3[15:8]).
  - Enable SD0 engine clock in eMMC (CLK_HCLKEN[22]).
  - Set FMI (CLK_HCLKEN[20]) high to enable clock while set eMMC (CLK_HCLKEN[22]) high to enable clock for SD0 controller operation.
- Reset Configuration
  - Reset SD0 controller in FMIRST (SYS_AHBIPRST[20]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
SD0	SD0_CMD	PC.5	MFP6
	SD0_CLK	PC.6	MFP6
	SD0_DAT0	PC.7	MFP6
	SD0_DAT1	PC.8	MFP6
	SD0_DAT2	PC.9	MFP6
	SD0_DAT3	PC.10	MFP6
	SD0_nCD	PB.8, PC.12	MFP6

Table 6.24-1 SD0 Pin Configuration

6.24.4.2 eMMC0 Basic Configuration

- Clock source Configuration
  - Select the source clock divide number for clock divider of APLL and UPLL on eMMC_SDIV (CLK_DIVCTL3[2:0]).
  - Select the source of eMMC0 engine clock on eMMC_S (CLK_DIVCTL3[4:3]).
  - Select the clock divider number of eMMC0 engine clock on eMMC_N (CLK_DIVCTL3[15:8]).
  - Enable eMMC0 engine clock in eMMC (CLK_HCLKEN[22]).
  - Set FMI (CLK_HCLKEN[20]) high to enable clock while set eMMC (CLK_HCLKEN[22]) high to enable clock for eMMC0 controller operation.
- Reset Configuration
  - Reset eMMC0 controller in FMIRST (SYS_AHBIPRST[20]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
eMMC0	eMMC0_CMD	PC.5	MFP6
	eMMC0_CLK	PC.6	MFP6
	eMMC0_DATA0	PC.7	MFP6
	eMMC0_DATA1	PC.8	MFP6
	eMMC0_DATA2	PC.9	MFP6
	eMMC0_DATA3	PC.10	MFP6

Table 6.24-2 eMMC0 Pin Configuration

6.24.4.3 NAND Flash Basic Configuration

- Clock source Configuration
  - Set FMI (CLK_HCLKEN[20]) high to enable clock while set NAND (CLK_HCLKEN[21]) high to enable clock for NAND Flash controller operation.
- Reset Configuration
  - Reset NAND Flash controller in FMIRST (SYS_AHBIPRST[20]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
NAND	NAND_ALE	PC.3	MFP3
	NAND_CLE	PC.4	MFP3
	NAND_DATA0	PC.8	MFP3
	NAND_DATA1	PC.9	MFP3
	NAND_DATA2	PC.10	MFP3
	NAND_DATA3	PC.11	MFP3
	NAND_DATA4	PC.12	MFP3
	NAND_DATA5	PC.13	MFP3
	NAND_DATA6	PC.14	MFP3
	NAND_DATA7	PC.15	MFP3
	NAND_RDY0	PC.7	MFP3
	NAND_nCS0	PC.1	MFP3
	NAND_nRE	PC.6	MFP3
	NAND_nWE	PC.5	MFP3
	NAND_nWP	PC.2	MFP3

Table 6.24-3 NAND Flash Pin Configuration

6.24.5 Functional Description

6.24.5.1 DMA Controller (DMAC)

The DMAC provides a DMA (Direct Memory Access) function for FMI controller to exchange data between system memory (ex. SDRAM) and shared buffer (128 bytes). Arbitration of DMA request between FMI is done by DMAC's bus master. User only simply fills in the starting address and enables DMAC, and then you can let DMAC to handle the data transfer automatically.

There is a 128 bytes shared buffer inside DMAC, separate into two 64 bytes ping-pong FIFO (total 128 bytes). It can provide multi-block transfers using ping-pong mechanism for FMI. When FMI is not busy, these shared buffers can be accessed directly by software.

6.24.5.2 Flash Memory Interface Controller (FMI)

The Flash Memory Interface supports SD0/eMMC0 and NAND-type Flash. FMI is cooperated with DMAC to provide a fast data transfer between system memory and cards. There is a single 128 bytes buffer embedded in DMAC for temporary data storage (separate into two 64 bytes ping-pong FIFO). Due to DMAC only has single channel, that means only one interface can be active at one time.

6.24.5.3 SD0/eMMC0

FMI provides an interface for SD0/eMMC0 Flash device access. This SD0/eMMC0 controller supports 1-bit/4-bit data bus mode for SD0/eMMC0 device.

SD0/eMMC0 controller uses an independent clock source named SD0_CLK /eMMC0_CLK as engine clock. SD0_CLK /eMMC0_CLK can be completely asynchronous with system clock HCLK. In addition, SD0/eMMC0 clock can be changed arbitrarily by software. Note that HCLK should be faster than SD0_CLK /eMMC0_CLK.

This SD0/eMMC0 controller can generate all types of 48-bit command to SD0/eMMC0 device and retrieve all types of response from SD0/eMMC0 device. After response in, the content of response will be stored at FMI_EMMCRESP0 and FMI_EMMCRESP1. SD0/eMMC0 controller will calculate CRC7 and check its correctness for response. If CRC7 is error, CRCIF (FMI_EMMCINTSTS[1]) will be set and CRC7 (FMI_EMMCINTSTS[2]) will be '0'. For response R1b, software should note that after response in, SD0/eMMC0 device will put busy signal on data line SD0_DATA0/eMMC0_DATA0; software has to check this status with clock polling until it became high. For response R3, CRC7 is invalid; but SD0/eMMC0 controller will still calculate CRC7 and get an error result, software should ignore this error and clear CRCIF (FMI_EMMCINTSTS[1]) flag.

This SD0/eMMC0 controller is composed of two state machines – command/response part and data part. For command/response part, the trigger bits are COEN (FMI_EMMCCTL[0]), RIEN (FMI_EMMCCTL[1]), R2EN (FMI_EMMCCTL[4]), CLK74OE (FMI_EMMCCTL[5]) and CLK8OE (FMI_EMMCCTL[6]). If software enables all of these bits, the execution priority will be CLK74OE (FMI_EMMCCTL[5]) > COEN (FMI_EMMCCTL[0]) > RIEN (FMI_EMMCCTL[1]) or R2EN (FMI_EMMCCTL[4]) > CLK8OE(FMI_EMMCCTL[6]). Please note that RIEN (FMI_EMMCCTL[1]) and R2EN (FMI_EMMCCTL[4]) cannot be triggered at the same time.

For data part, there are DIEN (FMI_EMMCCTL[2]) and DOEN (FMI_EMMCCTL[3]) for selection. Software can only trigger DIEN(FMI_EMMCCTL[2]) or DOEN (FMI_EMMCCTL[3]) at the same time. If DIEN (FMI_EMMCCTL[2]) is triggered, SD0/eMMC0 controller waits start bit from data line SD0_DATA0/eMMC0_DATA0 immediately, and then get specified amount data from SD0/eMMC0 device. After data-in, SD0/eMMC0 controller will check CRC16 correctness; if it is error, CRCIF (FMI_EMMCINTSTS[1]) will be set and CRC16 (FMI_EMMCINTSTS[3]) will be '0'. If DOEN (FMI_EMMCCTL[3]) is triggered, SD0/eMMC0 controller will wait until response in is finished, and then send specified amount data to SD0/eMMC0 device. After data-out, SD0/eMMC0 controller will get CRC status from SD0/eMMC0 device and check its correctness; it should be '010', otherwise CRCIF (FMI_EMMCINTSTS[1]) will be set and CRCSTS (FMI_EMMCINTSTS[6:4]) will be the value it received.

If R2EN (FMI_EMMCCTL[4]) is triggered, SD0/eMMC0 controller will receive response R2 (136 bits) from SD0/eMMC0 device, CRC7 and end bit will be dropped. The receiving data will be placed at DMAC's buffer, starting from address offset 0x0.

This SD0/eMMC0 controller also provides multiple block transfer function (change FMI_EMMCBLEN to change the block length). By using this function data transfer throughput accelerated. If CRC7, CRC16 or CRC status is error, SD0/eMMC0 controller will stop transfer and set CRCIF (FMI_EMMCINTSTS[1]). When this situation occurred, it's necessary to set DMARST (FMI_DMACTL [1]) to reset SD0/eMMC0 controller.

There is a hardware time-out mechanism for response in and data in inside of SD0/eMMC0 engine. By specifying a 24-bit time-out value at FMI_EMMCTOUT, SD0/eMMC0 controller will decide when to time-out the transfer.

6.24.5.4 NAND Flash

FMI provides an interface for NAND-type Flash access. It supports 2048bytes/page, 4096bytes/page and 8192bytes/page NAND. This NAND-type Flash controller provides all required signals for NAND Flash, including NAND_RDY0, NAND_nCS0, NAND_CLE, NAND_ALE, NAND_nWE, NAND_nRE and data pins NAND_DATA0 to NAND_DATA7.

The NAND Flash controller provide direct command port, address port and data port to control NAND Flash signals manually. When command port written, NAND Flash controller generate appropriate signal to NAND. When address port written without setting EOA (FMI_NANDADDR[31])] high, NAND Flash controller generate an address cycle to NAND, but do not clear ALE until the last address cycle written through address port with setting EOA (FMI_NANDADDR[31]) high. By using this method, address cycle can be generated by software arbitrarily.

For example, if user wants to write 4 address cycles to NAND, it necessary to write 3 addresses to address port without setting EOA (FMI_NANDADDR[31])] high, and then write the last one address to address port with setting EOA (FMI_NANDADDR[31])] high.

NAND Flash controller also provides a status and an interrupt flag of NAND_RDY0 pin. The interrupt flag will be set only when rising edge encountered on NAND_RDY0 pin.

FMI support four different page size, they are 2048bytes/page, 4096bytes/page and 8192bytes/page. Use PSIZE (FMI_NANDCTL[17:16]) to select the NAND Flash type. Using DMA function for data transfer could increase the performance. For different model of NAND, it's necessary to adjust the timing parameter at FMI_NANDTMCTL to meet specification of NAND Flash device. Adjust timing parameter can also improve data transfer performance.

The NAND Flash controller has a BCH algorithm for error recovery. The BCH algorithm can correct up to 8 bits errors, 12 bits errors or 24 bits errors. By reading ECC_FLD_IF (FMI_NANDINTSTS[2]) to check the error occurrence while by reading FMI_NANDECCES0, FMI_NANDECCES1, FMI_NANDECCES2 and FMI_NANDECCES3 to know how many errors and if those errors are correctable or not. If those errors are correctable, please read FMI_NANDECCEAx and FMI_NANDECCEDx to correct the errors manually.

For 2K/4K/8K Page size NAND Flash with BCH algorithm, T can be t8, t12 or t24. Based on the page size and T setting, FMI generate different size of parity data. The number of byte for parity data in different page size and T setting are listed in Table 6.24-4 shown below. The data arrangement of redundant area is as Figure 6.24-2, Figure 6.24-3 and Figure 6.24-4 shown below.

It's recommended to select appropriate T based on NAND Flash page size and redundant area size.

BCH Algorithm	**Parity (Byte) 2048 Page Size**	**Parity (Byte) 4096 Page Size**	**Parity (Byte) 8192 Page Size**
BCH T8	60	120	240
BCH T12	92	184	368

BCH T24	90	180	360

Table 6.24-4 Number of Parity (Byte) for Each BCH Algorithm

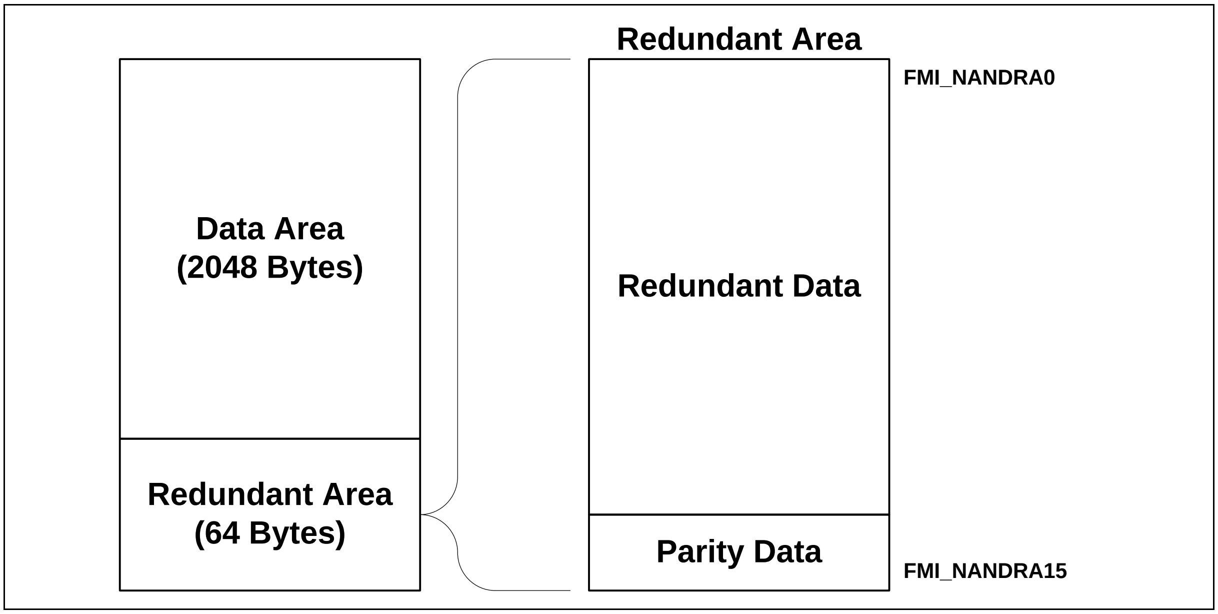


Figure 6.24-2 Data Arrangement for 2 kB Page Size NAND Flash

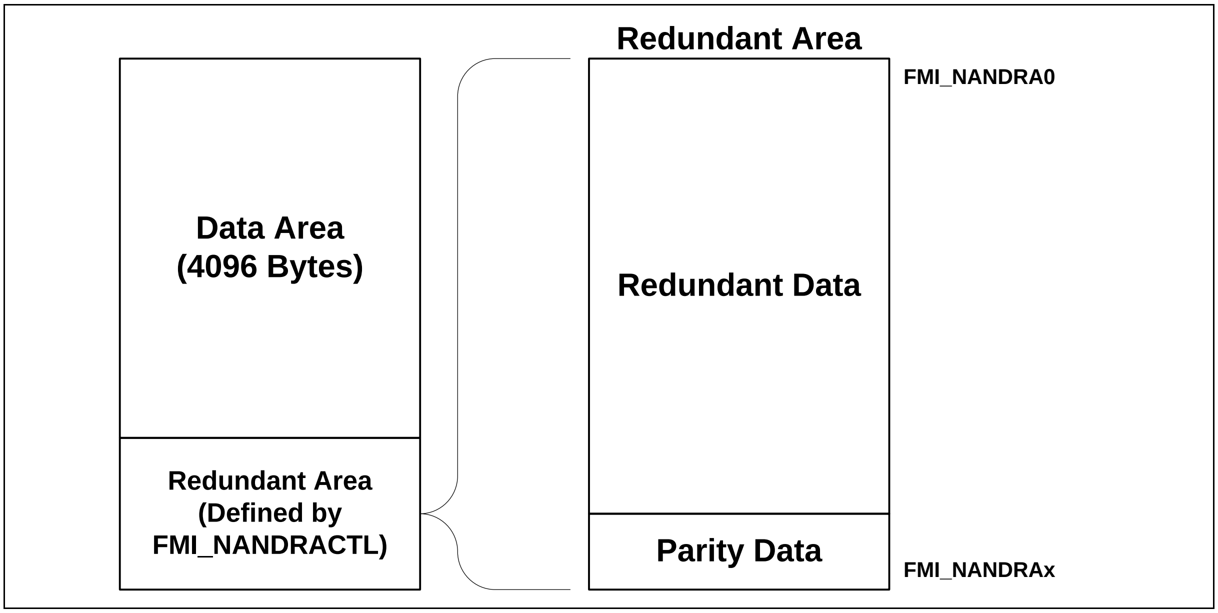


Figure 6.24-3 Data Arrangement for 4 kB Page Size NAND Flash

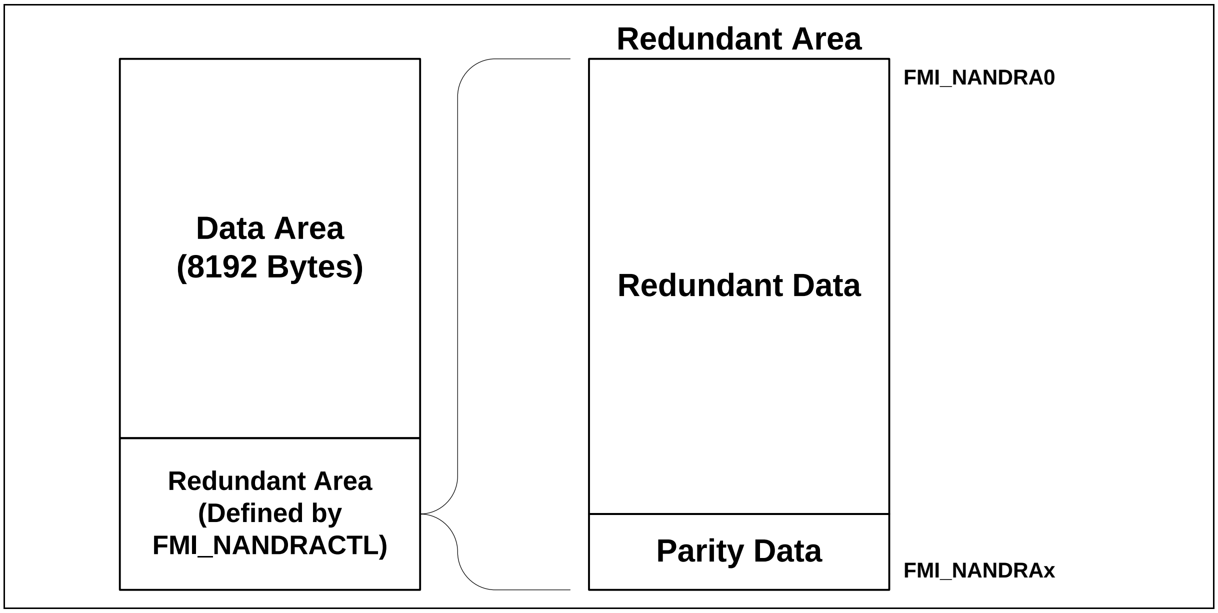


Figure 6.24-4 Data Arrangement for 8 KB Page Size NAND Flash

6.24.5.5 Programming Flow

<u>Here is a simple example programming flow without DMA Scatter-Gather enable.</u>

1. Set DMACEN (FMI_DMACTL[0]) to enable DMA Controller.
2. Fill corresponding starting address in FMI_DMASA for FMI.
3. Trigger SD0/EMMC0 or NAND flash controller to start DMA transfer.
4. Wait until DMA transfer finished.

<u>Here is a simple example programming flow with DMA Scatter-Gather enable.</u>

1. Set DMACEN (FMI_DMACTL[0]) to enable DMA and SGEN (FMI_DMACTL[3]) to enable Scatter-Gather function.
2. Fill corresponding starting address of Physical Address Descriptor (PAD) table in FMI_DMASA for FMI.
3. When bit-0 of FMI_DMASA is 1, the PAD will fetch in out of order, otherwise, it’s fetched in order from PAD. The first time of writing bit-0 with 1 or not is not available for this function. The bits will be available in PAD table.
4. Trigger SD0/EMMC0 or NAND flash controller to start DMA transfer.
5. Wait until DMA transfer finished.

6.24.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
FMI Base Address: **FMI_BA = 0xB001_9000**				
FMI_BUFFERn **n = 0, 1..31**	FMI_BA+0x000+0x4*n	R/W	FMI Embedded Buffer Word n n = 0, 1..31	0x0000_0000
FMI_DMACTL	FMI_BA+0x400	R/W	FMI DMA Control and Status Register	0x0000_0000
FMI_DMASA	FMI_BA+0x408	R/W	FMI DMA Transfer Starting Address Register	0x0000_0000
FMI_DMABCNT	FMI_BA+0x40C	R	FMI DMA Transfer Byte Count Register	0x0000_0000
FMI_DMAINTEN	FMI_BA+0x410	R/W	FMI DMA Interrupt Enable Control Register	0x0000_0001
FMI_DMAINTSTS	FMI_BA+0x414	R/W	FMI DMA Interrupt Status Register	0x0000_0000
FMI_GCTL	FMI_BA+0x800	R/W	FMI Global Control and Status Register	0x0000_0000
FMI_GINTEN	FMI_BA+0x804	R/W	FMI Global Interrupt Control Register	0x0000_0001
FMI_GINTSTS	FMI_BA+0x808	R/W	FMI Global Interrupt Status Register	0x0000_0000
FMI_EMMCCTL	FMI_BA+0x820	R/W	SD0/EMMC0 Control and Status Register	0x0101_0000
FMI_EMMCCMDARG	FMI_BA+0x824	R/W	SD0/EMMC0 Command Argument Register	0x0000_0000
FMI_EMMCINTEN	FMI_BA+0x828	R/W	SD0/EMMC0 Interrupt Control Register	0x0000_0000
FMI_EMMCINTSTS	FMI_BA+0x82C	R/W	SD0/EMMC0 Interrupt Status Register	0x000X_008C
FMI_EMMCRESP0	FMI_BA+0x830	R	SD0/EMMC0 Receiving Response Token Register 0	0x0000_0000
FMI_EMMCRESP1	FMI_BA+0x834	R	SD0/EMMC0 Receiving Response Token Register 1	0x0000_0000
FMI_EMMCBLEN	FMI_BA+0x838	R/W	SD0/EMMC0 Block Length Register	0x0000_01FF
FMI_EMMCTOUT	FMI_BA+0x83C	R/W	SD0/EMMC0 Response/Data-in Time-out Register	0x0000_0000
FMI_NANDCTL	FMI_BA+0x8A0	R/W	NAND Flash Control Register	0x0288_0090
FMI_NANDTMCTL	FMI_BA+0x8A4	R/W	NAND Flash Timing Control Register	0x0001_0105
FMI_NANDINTEN	FMI_BA+0x8A8	R/W	NAND Flash Interrupt Enable Register	0x0000_0000
FMI_NANDINTSTS	FMI_BA+0x8AC	R/W	NAND Flash Interrupt Status Register	0x000X_0000
FMI_NANDCMD	FMI_BA+0x8B0	W	NAND Flash Command Port Register	0xXXXX_XXXX
FMI_NANDADDR	FMI_BA+0x8B4	W	NAND Flash Address Port Register	0xXXXX_XXXX
FMI_NANDDATA	FMI_BA+0x8B8	R/W	NAND Flash Data Port Register	0xXXXX_XXXX
FMI_NANDRACTL	FMI_BA+0x8BC	R/W	NAND Flash Redundant Area Control Register	0x0000_0000
FMI_NANDECTL	FMI_BA+0x8C0	R/W	NAND Flash Extend Control Register	0x0000_0000
FMI_NANDECCES0	FMI_BA+0x8D0	R	NAND Flash ECC Error Status 0 Register	0x0000_0000
FMI_NANDECCES1	FMI_BA+0x8D4	R	NAND Flash ECC Error Status 1 Register	0x0000_0000
FMI_NANDECCES2	FMI_BA+0x8D8	R	NAND Flash ECC Error Status 2 Register	0x0000_0000

FMI_NANDECCES3	FMI_BA+0x8DC	R	NAND Flash ECC Error Status 3 Register	0x0000_0000
FMI_NANDECCEA0	FMI_BA+0x900	R	NAND Flash ECC Error Byte Address 0 Register	0x0000_0000
FMI_NANDECCEA1	FMI_BA+0x904	R	NAND Flash ECC Error Byte Address 1 Register	0x0000_0000
FMI_NANDECCEA2	FMI_BA+0x908	R	NAND Flash ECC Error Byte Address 2 Register	0x0000_0000
FMI_NANDECCEA3	FMI_BA+0x90C	R	NAND Flash ECC Error Byte Address 3 Register	0x0000_0000
FMI_NANDECCEA4	FMI_BA+0x910	R	NAND Flash ECC Error Byte Address 4 Register	0x0000_0000
FMI_NANDECCEA5	FMI_BA+0x914	R	NAND Flash ECC Error Byte Address 5 Register	0x0000_0000
FMI_NANDECCEA6	FMI_BA+0x918	R	NAND Flash ECC Error Byte Address 6 Register	0x0000_0000
FMI_NANDECCEA7	FMI_BA+0x91C	R	NAND Flash ECC Error Byte Address 7 Register	0x0000_0000
FMI_NANDECCEA8	FMI_BA+0x920	R	NAND Flash ECC Error Byte Address 8 Register	0x0000_0000
FMI_NANDECCEA9	FMI_BA+0x924	R	NAND Flash ECC Error Byte Address 9 Register	0x0000_0000
FMI_NANDECCEA10	FMI_BA+0x928	R	NAND Flash ECC Error Byte Address 10 Register	0x0000_0000
FMI_NANDECCEA11	FMI_BA+0x92C	R	NAND Flash ECC Error Byte Address 11 Register	0x0000_0000
FMI_NANDECCED0	FMI_BA+0x960	R	NAND Flash ECC Error Data Register 0	0x8080_8080
FMI_NANDECCED1	FMI_BA+0x964	R	NAND Flash ECC Error Data Register 1	0x8080_8080
FMI_NANDECCED2	FMI_BA+0x968	R	NAND Flash ECC Error Data Register 2	0x8080_8080
FMI_NANDECCED3	FMI_BA+0x96C	R	NAND Flash ECC Error Data Register 3	0x8080_8080
FMI_NANDECCED4	FMI_BA+0x970	R	NAND Flash ECC Error Data Register 4	0x8080_8080
FMI_NANDECCED5	FMI_BA+0x974	R	NAND Flash ECC Error Data Register 5	0x8080_8080
FMI_NANDRAn **n = 0, 1..117**	FMI_BA+0xA00+0x4*n	R/W	NAND Flash Redundant Area Word n n = 0, 1..117	Undefined

6.24.7 Register Description

FMI DMA Control and Status Register (FMI_DMACTL)

Register	**Offset**	**R/W**	**Description**	**Reset Value**
FMI_DMACTL	FMI_BA+0x400	R/W	FMI DMA Control and 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						**DMABUSY**	**Reserved**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**

Reserved	SGEN	Reserved	DMARST	DMACEN

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**DMABUSY**	**FMI DMA Transfer Is in Progress** This bit indicates if FMI is granted and doing DMA transfer or not. 0 = FMI DMA transfer is not in progress. 1 = FMI DMA transfer is in progress.
[8:4]	**Reserved**	Reserved.
[3]	**SGEN**	**Scatter-gather Function for FMI Enable Bit** 0 = Scatter-gather function Disabled (DMA will treat the starting address in DMASA as starting pointer of a single block memory). 1 = Scatter-gather function Enabled (DMA will treat the starting address in DMASA as a starting address of Physical Address Descriptor (PAD) table. The format of these Pads' will be described later).
[2]	**Reserved**	Reserved.
[1]	**DMARST**	**Software Engine Reset** 0 = No effect. 1 = Reset internal state machine and pointers. The contents of control register will not be cleared. This bit will auto be cleared after few clock cycles. **NOTE:** The software reset DMA related registers.
[0]	**DMACEN**	**DMA Controller Engine Enable Bit** 0 = DMA Controller Disabled. 1 = DMA Controller Enabled. **NOTE:** If this bit is cleared, DMA will ignore all requests from SD0/eMMC0 and force bus master into IDLE state. **NOTE:** If target abort is occurred, DMACEN will be cleared.

FMI DMA Transfer Starting Address Register (FMI_DMASA)

Register	Offset	R/W	Description	Reset Value
FMI_DMASA	FMI_BA+0x408	R/W	FMI DMA Transfer Starting Address Register	0x0000_0000

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

Bits	Description	
[31:1]	**DMASA**	**DMA Transfer Starting Address** This field pads 0 as least significant bit indicates a 32-bit starting address of system memory (SRAM) for DMA to retrieve or fill in data. If DMA is not in normal mode, this field will be interpreted as a starting address of Physical Address Descriptor (PAD) table. **Note:** Starting address of the SRAM must be word aligned, for example, 0x0000_0000, 0x0000_0004.
[0]	**ORDER**	**Determined to the PAD Table Fetching Is in Order or Out of Order** 0 = PAD table is fetched in order. 1 = PAD table is fetched out of order. **Note:** the bit 0 is valid in scatter-gather mode when SGEN (FMI_DMACTL[3]) = 1.

FMI DMA Transfer Byte Count Register (FMI_DMABCNT)

Register	Offset	R/W	Description	Reset Value
FMI_DMABCNT	FMI_BA+0x40C	R	FMI DMA Transfer Byte Count Register	0x0000_0000

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

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25:0]	**BCNT**	**DMA Transfer Byte Count (Read Only)** This field indicates the remained byte count of DMA transfer. The value of this field is valid only when FMI is busy; otherwise, it is zero.

FMI DMA Interrupt Enable Control Register (FMI_DMAINTEN)

Register	Offset	R/W	Description	Reset Value
FMI_DMAINTEN	FMI_BA+0x410	R/W	FMI DMA Interrupt Enable Control Register	0x0000_0001

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

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**WEOTIEN**	**Wrong EOT (End of Transfer)Encountered Interrupt Enable Bit** 0 = Interrupt generation Disabled when wrong EOT (End of Transfer) is encountered. 1 = Interrupt generation Enabled when wrong EOT (End of Transfer) is encountered.
[0]	**ABORTIEN**	**DMA Read/Write Target Abort Interrupt Enable Bit** 0 = Target abort interrupt generation Disabled during DMA transfer. 1 = Target abort interrupt generation Enabled during DMA transfer.

FMI DMA Interrupt Status Register (FMI_DMAINTSTS)

Register	Offset	R/W	Description	Reset Value
FMI_DMAINTSTS	FMI_BA+0x414	R/W	FMI DMA 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						**WEOTIF**	**ABORTIF**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**WEOTIF**	**Wrong EOT Encountered Interrupt Flag(Read Only)** When DMA Scatter-Gather function is enabled, and EOT of the descriptor is encountered before DMA transfer finished (that means the total sector count of all PAD is less than the sector count of FMI), this bit will be set. 0 = No EOT encountered before DMA transfer finished. 1 = EOT encountered before DMA transfer finished. **NOTE:** This bit is read only, but can be cleared by writing ‘1’ to it.
[0]	**ABORTIF**	**DMA Read/Write Target Abort Interrupt Flag(Read Only)** 0 = No bus ERROR response received. 1 = Bus ERROR response received. **NOTE:** This bit is read only, but can be cleared by writing ‘1’ to it. **NOTE:** When DMA’s bus master received ERROR response, it means that target abort happened. DMA will stop transfer and respond this event by set ABORTIF high. Then, FMI go to IDLE state. When target abort occurred or WEOTIF is set, it’s necessary to reset FMI’s DMA and related function (SD0/eMMC0 or NAND Flash controller), and then transfer those data again.

FMI Global Control and Status Register (FMI_GCTL)

Register	Offset	R/W	Description	Reset Value
FMI_GCTL	FMI_BA+0x800	R/W	FMI Global Control and 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				**NAND_EN**	**Reserved**	**SDEN**	**GCTLRST**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**NAND_EN**	**NAND Flash Functionality Enable Bit** 0 = NAND Flash functionality of FMI Disabled. 1 = NAND Flash functionality of FMI Enabled.
[2]	**Reserved**	Reserved.
[1]	**SDEN**	**SD0/EMMC0 Functionality Enable Bit** 0 = SD0/eMMC0 functionality of FMI Disabled. 1 = SD0/eMMC0 functionality of FMI Enabled.
[0]	**GCTLRST**	**Software Engine Reset** 0 = Writing 0 to this bit has no effect. 1 = Writing 1 to this bit will reset all FMI engines. The contents of control register will not be cleared. This bit will auto clear after few clock cycles.

NOTE: Only one engine can be enabled at one time, or FMI will work abnormal.

FMI Global Interrupt Control Register (FMI_GINTEN)

Register	Offset	R/W	Description	Reset Value
FMI_GINTEN	FMI_BA+0x804	R/W	FMI Global Interrupt 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							DTAIEN

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**DTAIEN**	**DMA READ/WRITE Target Abort Interrupt Enable Bit** 0 = DMA READ/WRITE target abort interrupt generation Disabled. 1 = DMA READ/WRITE target abort interrupt generation Enabled.

FMI Global Interrupt Status Register (FMI_GINTSTS)

Register	Offset	R/W	Description	Reset Value
FMI_GINTSTS	FMI_BA+0x808	R/W	FMI Global 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							**DTAIF**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**DTAIF**	**DMA READ/WRITE Target Abort Interrupt Flag (Read Only)** This bit indicates DMA received an ERROR response from internal AHB bus during DMA read/write operation. When Target Abort is occurred, please reset all engine. 0 = No bus ERROR response received. 1 = Bus ERROR response received. **NOTE:** This bit is read only, but can be cleared by writing '1' to it.

SD0/EMMC0 Control and Status Register (FMI_EMMCCTL)

Register	Offset	R/W	Description	Reset Value
FMI_EMMCCTL	FMI_BA+0x820	R/W	SD0/EMMC0 Control and Status Register	0x0101_0000

31	30	29	28	27	26	25	24
Reserved				SDNWR			
23	22	21	20	19	18	17	16
BLKCNT							
15	14	13	12	11	10	9	8
DBW	CTLRST	CMDCODE					
7	6	5	4	3	2	1	0
CLKKEEP	CLK8OE	CLK74OE	R2EN	DOEN	DIEN	RIEN	COEN

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:24]	**SDNWR**	**NWR Parameter for Block Write Operation** This value indicates the NWR parameter for data block write operation in SD0/eMMC0 clock counts. The actual clock cycle will be SDNWR+1.
[23:16]	**BLKCNT**	**Block Counts to Be Transferred or Received** This field contains the block counts for data-in and data-out transfer. For READ_MULTIPLE_BLOCK and WRITE_MULTIPLE_BLOCK command, using this function to accelerate data transfer and improve performance. Don't fill 0x0 to this field. **Note:** For READ_MULTIPLE_BLOCK and WRITE_MULTIPLE_BLOCK command, the actual total length is BLKCNT * (FMI_EMMCBLEN[10:0]+1).
[15]	**DBW**	**SD0/EMMC0 Data Bus Width (for 1-bit / 4-bit Selection)** 0 = Data bus width is 1-bit. 1 = Data bus width is 4-bit.
[14]	**CTLRST**	**Software Engine Reset** 0 = Writing 0 to this bit has no effect. 1 = Reset the internal state machine and counters. The contents of control register will not be cleared (but CLK8OE, CLK74OE, R2EN, DOEN, DIEN, RIEN and COEN will be cleared). This bit will be auto cleared after few clock cycles.
[13:8]	**CMDCODE**	**SD0/EMMC0 Command Code** This register contains the SD0/eMMC0 command code (0x00 – 0x3F).
[7]	**CLKKEEP**	**SD0/EMMC0 Clock Enable Control** 0 = SD0/eMMC0 decided when to output clock and when to disable clock output automatically. 1 = SD0/eMMC0 always keeps free running.

[6]	**CLK8OE**	**Generating 8 Clock Cycles Output Enable Bit** 0 = No effect. (Please use CTLRST (FMI_EMMCCTL[1]) to clear this bit.) 1 = Enabled, SD0/eMMC0 will output 8 clock cycles. **NOTE:** When operation is finished, this bit will be cleared automatically, so don't write 0 to this bit (the controller will be abnormal).
[5]	**CLK74OE**	**Initial 74 Clock Cycles Output Enable Bit** 0 = No effect. (Please use CTLRST (FMI_EMMCCTL[1]) to clear this bit.) 1 = Enabled, SD0/eMMC0 will output 74 clock cycles to SD0/eMMC0 device. **NOTE:** When operation is finished, this bit will be cleared automatically, so don't write 0 to this bit (the controller will be abnormal).
[4]	**R2EN**	**Response R2 Input Enable Bit** 0 = No effect. (Please use CTLRST (FMI_EMMCCTL[1]) to clear this bit.) 1 = Enabled, SD0/eMMC0 will wait to receive a response R2 from SD0/eMMC0 device and store the response data into DMAC's Flash buffer (exclude CRC7). **NOTE:** When operation is finished, this bit will be cleared automatically, so don't write 0 to this bit (the controller will be abnormal).
[3]	**DOEN**	**Data Output Enable Bit** 0 = No effect. (Please use CTLRST (FMI_EMMCCTL[1]) to clear this bit.) 1 = Enabled, SD0/eMMC0 will transfer block data and the CRC16 value to SD0/eMMC0 device. **NOTE:** When operation is finished, this bit will be cleared automatically, so don't write 0 to this bit (the controller will be abnormal).
[2]	**DIEN**	**Data Input Enable Bit** 0 = No effect. (Please use CTLRST (FMI_EMMCCTL[1]) to clear this bit.) 1 = Enabled, SD0/eMMC0 will wait to receive block data and the CRC16 value from SD0/eMMC0 device. **NOTE:** When operation is finished, this bit will be cleared automatically, so don't write 0 to this bit (the controller will be abnormal).
[1]	**RIEN**	**Response Input Enable Bit** 0 = No effect. (Please use CTLRST (FMI_DMACTL[1]) to clear this bit.) 1 = Enabled, SD0/eMMC0 will wait to receive a response from SD0/eMMC0 device. **NOTE:** When operation is finished, this bit will be cleared automatically, so don't write 0 to this bit (the controller will be abnormal).
[0]	**COEN**	**Command Output Enable Bit** 0 = No effect. (Please use CTLRST (FMI_EMMCCTL[1]) to clear this bit.) 1 = Enabled, SD0/eMMC0 will output a command to SD0/eMMC0 device. **NOTE:** When operation is finished, this bit will be cleared automatically, so don't write 0 to this bit (the controller will be abnormal).

SD0/EMMC0 Command Argument Register (FMI_EMMCCMDARG)

Register	Offset	R/W	Description	Reset Value
FMI_EMMCCMDARG	FMI_BA+0x824	R/W	SD0/EMMC0 Command Argument Register	0x0000_0000

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

Bits	Description	
[31:0]	**ARGUMENT**	**SD0/EMMC0 Command Argument** This register contains a 32-bit value specifies the argument of SD0/eMMC0 command from host controller to SD0/eMMC0 device. Before trigger COEN (FMI_EMMCCTL[0]), software should fill argument in this field.

SD0/EMMC0 Interrupt Control Register (FMI_EMMCINTEN)

Register	Offset	R/W	Description	Reset Value
FMI_EMMCINTEN	FMI_BA+0x828	R/W	SD0/EMMC0 Interrupt Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	CDSRC	Reserved					
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved	WKIEN	DITOIEN	RTOIEN	Reserved	SD0_IE	Reserved	CDIEN
7	6	5	4	3	2	1	0
Reserved						CRCIEN	BLKDIEN

Bits	Description	
[31]	Reserved	Reserved.
[30]	CDSRC	**SD0/EMMC0 Card Detect Source Selection** 0 = From SD0/eMMC0 card's DAT3 pin. Host need clock to got data on pin DAT3. Please make sure CLKKEEP (FMI_EMMCCTL [7]) is 1 in order to generate free running clock for DAT3 pin. 1 = From GPIO pin.
[29:15]	Reserved	Reserved.
[14]	WKIEN	**Wake-up Signal Generating Enable Bit** Enable/Disable wake-up signal generating of SD0/eMMC0 controller when card is inserted or removed. 0 = SD0/eMMC0 Card interrupt to wake-up chip Disabled. 1 = SD0/eMMC0 Card interrupt to wake-up chip Enabled.
[13]	DITOIEN	**Data Input Time-out Interrupt Enable Bit** Enable/Disable interrupts generation of SD0/eMMC0 controller when data input time-out. Time-out value is specified at TOUT (FMI_EMMCTOUT[23:0]). 0 = DITOIF (FMI_EMMCINTSTS[13]) trigger interrupt Disabled. 1 = DITOIF (FMI_EMMCINTSTS[13]) trigger interrupt Enabled.
[12]	RTOIEN	**Response Time-out Interrupt Enable Bit** Enable/Disable interrupts generation of SD0/eMMC0 controller when receiving response or R2 time-out. Time-out value is specified at TOUT (FMI_EMMCTOUT[23:0]). 0 = RTOIF (FMI_EMMCINTSTS[12]) trigger interrupt Disabled. 1 = RTOIF (FMI_EMMCINTSTS[12]) trigger interrupt Enabled.
[11]	Reserved	Reserved.

[10]	**SD0_IE**	**SD0/EMMC0 Interrupt Enable Bit for Port** Enable/Disable interrupts generation of SD0/eMMC0 when SD0/eMMC0 card issues an interrupt via DAT [1] to host. 0 = SD0_IF (FMI_EMMCINTSTS[10]) trigger interrupt Disabled. 1 = SD0_IF (FMI_EMMCINTSTS[10]) trigger interrupt Enabled.
[9]	**Reserved**	Reserved.
[8]	**CDIEN**	**SD0/EMMC0 Card Detection Interrupt Enable Bit** Enable/Disable interrupts generation of SD0/eMMC0 controller when card is inserted or removed. 0 = CDIF (FMI_EMMCINTSTS[8]) trigger interrupt Disabled. 1 = CDIF (FMI_EMMCINTSTS[8]) trigger interrupt Enabled.
[7:2]	**Reserved**	Reserved.
[1]	**CRCIEN**	**CRC7, CRC16 and CRC Status Error Interrupt Enable Bit** 0 = CRCIF (FMI_EMMCINTSTS[1]) trigger interrupt Disabled. 1 = CRCIF (FMI_EMMCINTSTS[1]) trigger interrupt Enabled.
[0]	**BLKDIEN**	**Block Transfer Done Interrupt Enable Bit** 0 = BLKDIF (FMI_EMMCINTSTS[0]) trigger interrupt Disabled. 1 = BLKDIF (FMI_EMMCINTSTS[0]) trigger interrupt Enabled.

SD0/EMMC0 Interrupt Status Register (FMI_EMMCINTSTS)

Register	Offset	R/W	Description	Reset Value
FMI_EMMCINTSTS	FMI_BA+0x82C	R/W	SD0/EMMC0 Interrupt Status Register	0x000X_008C

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					DAT1STS	Reserved	CDSTS
15	14	13	12	11	10	9	8
Reserved		DITOIF	RTOIF	Reserved	SD0_IF	Reserved	CDIF
7	6	5	4	3	2	1	0
DAT0STS	CRCSTS			CRC16	CRC7	CRCIF	BLKDIF

Bits	Description	
[31:19]	Reserved	Reserved.
[18]	DAT1STS	**DAT1 Pin Status of SD0/EMMC0 Card (Read Only)** This bit indicates the DAT1 pin status of SD0/eMMC0 card.
[17]	Reserved	Reserved.
[16]	CDSTS	**Card Detect Status of SD0/EMMC0 (Read Only)** This bit indicates the card detect pin status of SD0/eMMC0, and is used for card detection. When there is a card inserted in or removed from SD0/eMMC0, software should check this bit to confirm if there is really a card insertion or removal. If CDSRC (FMI_EMMCINTEN [30]) = 0, to select DAT3 for card detection:. 0 = Card removed. 1 = Card inserted. If CDSRC (FMI_EMMCINTEN [30]) = 1, to select GPIO for card detection:. 0 = Card inserted. 1 = Card removed.
[15:14]	Reserved	Reserved.
[13]	DITOIF	**Data Input Time-out Interrupt Flag (Read Only)** This bit indicates that SD0/eMMC0 counts to time-out value when receiving data (waiting start bit). 0 = Not time-out. 1 = Data input time-out. **NOTE:** This bit is read only, but can be cleared by writing ‘1’ to it.

[12]	**RTOIF**	**Response Time-out Interrupt Flag (Read Only)** This bit indicates that SD0/eMMC0 counts to time-out value when receiving response or R2 (waiting start bit). 0 = Not time-out. 1 = Response time-out. **NOTE:** This bit is read only, but can be cleared by writing ‘1’ to it.
[11]	**Reserved**	Reserved.
[10]	**SD0_IF**	**SD0/EMMC0 Interrupt Flag (Read Only)** This bit indicates that SD0/eMMC0 card issues an interrupt to host. This interrupt is designed to level sensitive. Before clear it, turn off SD0/eMMC0 Interrupt Control Register [SD0_IE] first. 0 = No interrupt is issued by SD0/eMMC0 card. 1 = An interrupt is issued by SD0/eMMC0 card. **NOTE:** This bit is read only, but can be cleared by writing ‘1’ to it.
[9]	**Reserved**	Reserved.
[8]	**CDIF**	**SD0/EMMC0 Card Detection Interrupt Flag (Read Only)** This bit indicates that SD0/eMMC0 card is inserted or removed. Only when CDIEN (FMI_EMMCINTEN [8]) is set to 1, this bit is active. 0 = No card is inserted or removed. 1 = There is a card inserted in or removed from SD0/eMMC0. **NOTE:** This bit is read only, but can be cleared by writing ‘1’ to it.
[7]	**DAT0STS**	**DAT0 Pin Status of Current Selected SD0/EMMC0 Port (Read Only)** This bit is the DAT0 pin status of current selected SD0/eMMC0 port.
[6:4]	**CRCSTS**	**CRC Status Value of Data-out Transfer (Read Only)** SD0/eMMC0 will record CRC status of data-out transfer. Software could use this value to identify what type of error is during data-out transfer. 010 = Positive CRC status. 101 = Negative CRC status. 111 = SD0/eMMC0 device programming error occurs.
[3]	**CRC16**	**CRC16 Check Status of Data-in Transfer (Read Only)** SD0/eMMC0 will check CRC16 correctness after data-in transfer. 0 = Fault. 1 = OK.
[2]	**CRC7**	**CRC7 Check Status (Read Only)** SD0/eMMC0 will check CRC7 correctness during each response in. If that response does not contain CRC7 information (ex. R3), then software should turn off CRCIEN (FMI_EMMCINTEN[1]) and ignore this bit. 0 = Fault. 1 = OK.

[1]	**CRCIF**	**CRC7, CRC16 and CRC Status Error Interrupt Flag (Read Only)** This bit indicates that SD0/eMMC0 has occurred CRC error during response in, data-in or data-out (CRC status error) transfer. When CRC error is occurred, software should reset SD0/eMMC0 engine. Some response (ex. R3) doesn't have CRC7 information with it. However, SD0/eMMC0 will still calculate CRC7, get CRC error and set this flag. In this condition, please ignore CRC error and clears this bit manually. 0 = No CRC error is occurred. 1 = CRC error is occurred. **NOTE:** This bit is read only, but can be cleared by writing '1' to it.
[0]	**BLKDIF**	**Block Transfer Done Interrupt Flag (Read Only)** This bit indicates that SD0/eMMC0 has finished all data-in or data-out block transfer. If there is a CRC16 error or incorrect CRC status during multiple block data transfer, the transfer will be broken and this bit will also be set. 0 = Not finished yet. 1 = Done. **NOTE:** This bit is read only, but can be cleared by writing '1' to it.

SD0/EMMC0 Receiving Response Token Register 0 (FMI_EMMCRESP0)

Register	Offset	R/W	Description	Reset Value
FMI_EMMCRESP0	FMI_BA+0x830	R	SD0/EMMC0 Receiving Response Token Register 0	0x0000_0000

31	30	29	28	27	26	25	24
RESPTK0							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
RESPTK0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RESPTK0							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RESPTK0							

Bits	Description	
[31:0]	**RESPTK0**	**SD0/EMMC0 Receiving Response Token 0 (Read Only)** SD0/eMMC0 controller will receive a response token for getting a reply from SD0/eMMC0 device when RIEN (FMI_EMMCCTL[1]) is set. This field contains response bit 47-16 of the response token.

SD0/EMMC0 Receiving Response Token Register 1 (FMI_EMMCRESP1)

Register	Offset	R/W	Description	Reset Value
FMI_EMMCRESP1	FMI_BA+0x834	R	SD0/EMMC0 Receiving Response Token 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**
RESPTK1							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**RESPTK1**	**SD0/EMMC0 Receiving Response Token 1 (Read Only)** SD0/eMMC0 controller will receive a response token for getting a reply from SD0/eMMC0 device when RIEN (FMI_EMMCCTL[1]) is set. This register contains the bit 15-8 of the response token.

SD0/EMMC0 Block Length Register (FMI_EMMCBLEN)

Register	Offset	R/W	Description	Reset Value
FMI_EMMCBLEN	FMI_BA+0x838	R/W	SD0/EMMC0 Block Length Register	0x0000_01FF

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

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10:0]	**BLKLEN**	**SD0/EMMC0 Block Length in Byte Unit** An 11-bit value specifies the SD0/eMMC0 transfer byte count of a block. The actual byte count is equal to BLKLEN+1. Note : The default SD0/eMMC0 block length is 512 bytes

SD0/EMMC0 Response/Data-in Time-out Register (FMI_EMMCTOUT)

Register	Offset	R/W	Description	Reset Value
FMI_EMMCTOUT	FMI_BA+0x83C	R/W	SD0/EMMC0 Response/Data-in Time-out Register	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**TOUT**	**SD0/EMMC0 Response/Data-in Time-out Value** A 24-bit value specifies the time-out counts of response and data input. SD0/eMMC0 controller will wait start bit of response or data-in until this value reached. The time period is depended on SD0/eMMC0 engine clock frequency. Do not write a small number into this field, or you may never get response or data due to time-out. **NOTE:** Filling 0x0 into this field will disable hardware time-out function.

NAND Flash Control Register (FMI_NANDCTL)

Register	Offset	R/W	Description	Reset Value
FMI_NANDCTL	FMI_BA+0x8A0	R/W	NAND Flash Control Register	0x0288_0090

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**
Reserved						**CS0**	**Reserved**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ECC_EN	**BCH_TSEL**					**PSIZE**	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**SRAM_INT**	**PROT_3BEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ECC_CHK	**Reserved**		**REDUN_AUTO_WEN**	**REDUN_REN**	**DWR_EN**	**DRD_EN**	**SW_RST**

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25]	**CS0**	**NAND Flash Chip Select 0 Enable Bit** 0 = Chip select 0 Enabled. 1 = Chip select 0 Disabled.
[24]	**Reserved**	Reserved.
[23]	**ECC_EN**	**ECC Algorithm Enable Bit** This field is used to select the ECC algorithm for data protecting. The BCH algorithm can correct 8 or 12 or 24 bits. 0 = BCH code encode/decode Disabled. 1 = BCH code encode/decode Enabled. **Note:** If disable ECC_EN and when read data from NAND, NAND controller will ignore its ECC check result. When write data to NAND, NAND controller will write out 0xFF to every parity field. **Note:** The ECC algorithm only protects data area and hardware ECC parity code in default. By setting PROT_3BEN (FMI_NANDCTL[8]) high, the first 3 bytes of redundant data are also protected by ECC algorithm.
[22:18]	**BCH_TSEL**	**BCH Correct Bit Selection** This field is used to select BCH correct bits for data protecting. For BCH algorithm, T can be 8 or 12 or 24 for choosing (correct 8 or 12 or 24 bits). 00001 = Using BCH T24 to encode/decode (T24).(1024 Bytes per block) 00100 = Using BCH T8 to encode/decode (T8). 01000 = Using BCH T12 to encode/decode (T12).

[17:16]	**PSIZE**	**Page Size of NAND** This bit indicates the page size of NAND. There are four page sizes for choose, 2048bytes/page, 4096bytes/page and 8192bytes/page. Before setting PSIZE register, user must set BCH_TSEL register at first. 01 = Page size is 2048bytes/page. 10 = Page size is 4096bytes/page. 11 = Page size is 8192bytes/page.
[15:10]	**Reserved**	Reserved.
[9]	**SRAM_INT**	**SRAM Initial** 0 = Writing 0 to this bit has no effect. 1 = Writing 1 to this bit will reset the internal FMI_NANDRA0~FMI_NANDRA1 to 0xFFFF_FFFF. The contents of control register will not be cleared. This bit will be auto cleared after few clock cycles.
[8]	**PROT_3BEN**	**Protect_3Byte Software Data Enable Bit** The ECC algorithm only protects data area and hardware ECC parity code. User can choose to protect software redundant data first 3 bytes by setting this bit high. 0 = Software redundant data is not protected by ECC algorithm. 1 = Software redundant data first 3 bytes protected by ECC algorithm.
[7]	**ECC_CHK**	**None Used Field ECC Check After Read Page Data** 0 = Disabled. NAND controller will always check ECC result for each field, no matter it is used or not. 1 = Enabled. NAND controller will check 1's count for byte 2, 3 of redundant data of the ECC in each field. If count value is greater than 8, NAND controller will treat this field as none used field; otherwise, it's used. If that field is none used field, NAND controller will ignore its ECC check result.
[6:5]	**Reserved**	Reserved.
[4]	**REDUN_AUTO_WEN**	**Redundant Area Auto Write Enable Bit** This field is used to auto write redundant data out to NAND Flash. The redundant data area is dependent on FMI_NANDRACTL register. 0 = Auto write redundant data out to NAND Flash Disabled. 1 = Auto write redundant data out to NAND Flash Enabled.
[3]	**REDUN_REN**	**Redundant Area Read Enable Bit** This bit enables NAND controller to transfer redundant data from NAND Flash into FMI_NANDRA, the data size is dependent on FMI_NANDRACTL register. 0 = No effect. 1 = Read redundant data transfer Enabled. **NOTE:** When transfer completed, this bit will be cleared automatically.
[2]	**DWR_EN**	**DMA Write Data Enable Bit** This bit enables NAND controller to transfer data (1 page) from DMAC's embedded frame buffer into NAND Flash or NAND type Flash. 0 = No effect. 1 = DMA write data transfer Enabled. **NOTE:** When DMA transfer completed, this bit will be cleared automatically.

[1]	**DRD_EN**	**DMA Read Data Enable Bit** This bit enables NAND controller to transfer data (1 page) from NAND Flash or NAND type Flash into DMAC’s embedded frame buffer. 0 = No effect. 1 = DMA read data transfer Enabled. **NOTE:** When DMA transfer completed, this bit will be cleared automatically.
[0]	**SW_RST**	**Software Engine Reset** 0 = Writing 0 to this bit has no effect. 1 = Writing 1 to this bit will reset the internal state machine and counters (include DWR_EN (FMI_NANDCTL[2]) and DRD_EN (FMI_NANDCTL[1])). The contents of control register will not be cleared. This bit will be auto cleared after few clock cycles.

NAND Flash Timing Control Register (FMI_NANDTMCTL)

Register	Offset	R/W	Description	Reset Value
FMI_NANDTMCTL	FMI_BA+0x8A4	R/W	NAND Flash Timing Control Register	0x0001_0105

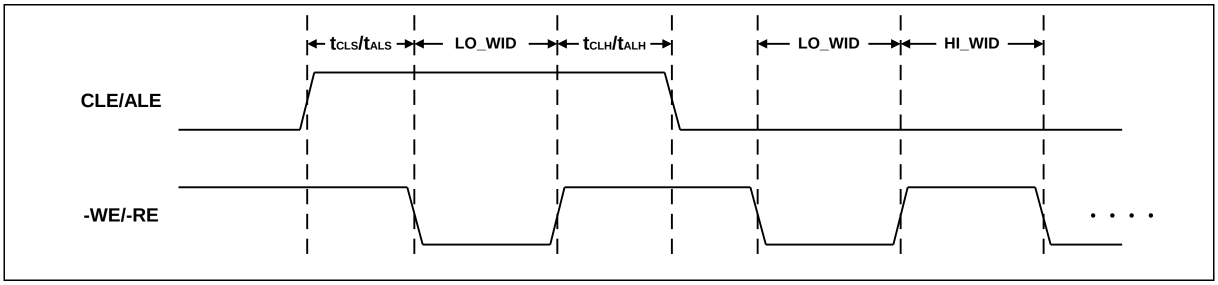
31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	**CALE_SH**						
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
HI_WID							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LO_WID							

Bits	Description	
[31:23]	**Reserved**	Reserved.

[22:16]	**CALE_SH**	**CLE/ALE Setup/Hold Time**
		This field controls the CLE/ALE setup/hold time to –WE.
		The setup/hold time can be calculated using following equation:
		tCLS = (CALE_SH+1)*TAHB.
		tCLH = ((CALE_SH*2)+2)*TAHB.
		tALS = (CALE_SH+1)*TAHB.
		tALH = ((CALE_SH*2)+2)*TAHB.

[15:8]	**HI_WID**	**Read/Write Enable Signal High Pulse Width** This field controls the high pulse width of signals –RE and –WE while H/W mode page access is enabled. The pulse width is a multiple of period of AHB bus clock. (The actual width time will be [clock period*(HI_WID+1)])
[7:0]	**LO_WID**	**Read/Write Enable Signal Low Pulse Width** This field controls the low pulse width of signals –RE and –WE while H/W mode page access is enabled. The pulse width is a multiple of period of AHB bus clock. (The actual width time will be [clock period*(LO_WID+1)])

NOTE1: The reset value calculated based on 150 MHz AHB Clock.



Timing Controlled by FMI_NANDTMCTL Register

NAND Flash Interrupt Enable Register (FMI_NANDINTEN)

Register	Offset	R/W	Description	Reset Value
FMI_NANDINTEN	FMI_BA+0x8A8	R/W	NAND Flash Interrupt Enable Register	0x0000_0000

<table>
<tr><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th></tr>
<tr><td colspan="5">Reserved</td><td>RB0_IE</td><td colspan="2">Reserved</td></tr>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="5">Reserved</td><td>ECC_FLD_IE</td><td>Reserved</td><td>DMA_IE</td></tr>
</table>

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10]	**RB0_IE**	**Ready/-Busy Rising Edge Detect Interrupt Enable Bit** 0 = R/-B rising edge detect interrupt generation Disabled. 1 = R/-B rising edge detect interrupt generation Enabled.
[9:3]	**Reserved**	Reserved.
[2]	**ECC_FLD_IE**	**ECC Field Check Error Interrupt Enable Bit** This bit can check the ECC error on each field (512bytes) of data transfer. Enable this bit to detect error and do error correction. 0 = Disabled. 1 = Enabled.
[1]	**Reserved**	Reserved.
[0]	**DMA_IE**	**DMA Read/Write Data Complete Interrupt Enable Bit** 0 = DMA read/write data complete interrupt generation Disabled. 1 = DMA read/write data complete interrupt generation Enabled.

NAND Flash Interrupt Status Register (FMI_NANDINTSTS)

Register	Offset	R/W	Description	Reset Value
FMI_NANDINTSTS	FMI_BA+0x8AC	R/W	NAND Flash Interrupt Status Register	0x000X_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved					RB0_Status	Reserved	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					RB0_IF	Reserved	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					ECC_FLD_IF	Reserved	DMA_IF

Bits	Description	
[31:19]	**Reserved**	Reserved.
[18]	**RB0_Status**	**Ready/-Busy 0 Pin Status (Read Only)** This bit reflects the Ready/-Busy pin status of NAND Flash.
[17:11]	**Reserved**	Reserved.
[10]	**RB0_IF**	**Ready/-Busy 0 Rising Edge Detect Interrupt Flag (Read Only)** 0 = R/-B rising edge is not detected. 1 = R/-B rising edge is detected. **NOTE:** This bit is read only, but can be cleared by writing ‘1’ to it.
[9:3]	**Reserved**	Reserved.
[2]	**ECC_FLD_IF**	**ECC Field Check Error Interrupt Flag (Read Only)** This bit can check the ECC error on each field (512bytes) of data transfer. Read this bit to check if the error occurred. 0 = No occurrence of ECC error. 1 = Occurrence of ECC error. **NOTE:** This bit is read only, but can be cleared by writing ‘1’ to it.
[1]	**Reserved**	Reserved.
[0]	**DMA_IF**	**DMA Read/Write Data Complete Interrupt Flag (Read Only)** 0 = DMA read/write transfer is not finished yet. 1 = DMA read/write transfer is done. **NOTE:** This bit is read only, but can be cleared by writing ‘1’ to it.

NAND Flash Command Port Register (FMI_NANDCMD)

Register	Offset	R/W	Description	Reset Value
FMI_NANDCMD	FMI_BA+0x8B0	W	NAND Flash Command Port Register	0xXXXX_XXXX

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**COMMAND**	**NAND Flash Command Port** When CPU writes to this port, FMI will send a command to NAND Flash.

NAND Flash Address Port Register (FMI_NANDADDR)

Register	Offset	R/W	Description	Reset Value
FMI_NANDADDR	FMI_BA+0x8B4	W	NAND Flash Address Port Register	0xXXXX_XXXX

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

Bits	Description	
[31]	**EOA**	**End of Address** Writing this bit to indicate if this address is the last one or not. By writing address port with this bit low, NAND Flash controller will set ALE pin to active (HIGH). After the last address is written (with this bit set high), NAND Flash controller will set ALE pin to inactive (LOW). 0 = Not the last address cycle. 1 = The last one address cycle.
[30:8]	**Reserved**	Reserved.
[7:0]	**ADDRESS**	**NAND Flash Address Port** By writing this port, NAND Flash control will send an address to NAND Flash.

NAND Flash Data Port Register (FMI_NANDDATA)

Register	Offset	R/W	Description	Reset Value
FMI_NANDDATA	FMI_BA+0x8B8	R/W	NAND Flash Data Port Register	0xXXXX_XXXX

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

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:0]	**DATA**	**NAND Flash Data Port** CPU can access NAND's memory array through this data port. When CPU WRITE, the lower 8-bit data from CPU will appear on the data bus of NAND controller. When CPU READ, NAND controller will get 8-bit data from data bus.

NAND Flash Redundant Area Control Register (FMI_NANDRACTL)

Register	Offset	R/W	Description	Reset Value
FMI_NANDRACTL	FMI_BA+0x8BC	R/W	NAND Flash Redundant Area Control Register	0x0000_0000

31	30	29	28	27	26	25	24
MECC							
23	22	21	20	19	18	17	16
MECC							
15	14	13	12	11	10	9	8
Reserved							RA128EN
7	6	5	4	3	2	1	0
RA128EN							

Bits	Description	
[31:16]	MECC	**Mask ECC During Write Page Data** These 16 bits registers indicate NAND controller to write out ECC parity or just 0xFF for each field (every 512 bytes) the real parity data will be write out to FMI_NANDRAx. 0x00 = Do not mask the ECC parity for each field. 0x01 = Mask ECC parity and write out FF to NAND ECC parity 2K/4K/8K page size first 512 field. 0x02 = Mask ECC parity and write out FF to NAND ECC parity 2K/4K/8K page size second 512 field. Others = Mask ECC parity and write out FF to NAND ECC parity 2K/4K/8K page size each 512 field.
[15:9]	Reserved	Reserved.
[8:0]	RA128EN	**Redundant Area 128 Byte Enable Bit** These bits indicate NAND Flash extended redundant area. If PSIZE (FMI_NANDCTL[17:16]) = 2'b01, this field will be set 0x40 (64bytes) automatically. If PSIZE (FMI_NANDCTL[17:16]) = 2'b10, this field will be set 0x80 (128 bytes) automatically. If PSIZE (FMI_NANDCTL[17:16]) = 2'b11, this field will be set 0x100 (256bytes) automatically. **Note:** The REA128EN must be 4 byte aligned, so bit1 and bit0 can't be filled 1 to it. The maximum redundant area of the controller is 472Bytes.

NAND Flash Extend Control Regsiter (FMI_NANDECTL)

Register	Offset	R/W	Description	Reset Value
FMI_NANDECTL	FMI_BA+0x8C0	R/W	NAND Flash Extend 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							**WP**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**WP**	**NAND Flash Write Protect Control (Low Active)** Set this bit low to make NAND_nWP functional pin low to prevent the write to NAND Flash device. 0 = NAND Flash is write-protected and is not writeable. 1 = NAND Flash is not write-protected and is writeable.

NAND Flash ECC Error Status 0 Register (FMI_NANDECCES0)

Register	Offset	R/W	Description	Reset Value
FMI_NANDECCES0	FMI_BA+0x8D0	R	NAND Flash ECC Error Status 0 Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	F4_ECNT					F4_STAT	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	F3_ECNT					F3_STAT	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	F2_ECNT					F2_STAT	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	F1_ECNT					F1_STAT	

Bits	Description	
[31]	**Reserved**	Reserved.
[30:26]	**F4_ECNT**	**Error Count of ECC Field 4** This field contains the error counts after ECC correct calculation of Field 4. For this ECC core (BCH algorithm), only when F4_STAT equals to 0x01, the value in this field is meaningful. F4_ECNT means how many errors depending on which ECC is used.
[25:24]	**F4_STAT**	**ECC Status of Field 4** This field contains the ECC correction status (BCH algorithm) of ECC-field 4. 00 = No error. 01 = Correctable error. 10 = Uncorrectable error. 11 = Reserved.
[23]	**Reserved**	Reserved.
[22:18]	**F3_ECNT**	**Error Count of ECC Field 3** This field contains the error counts after ECC correct calculation of Field 3. For this ECC core (BCH algorithm), only when F3_STAT equals to 0x01, the value in this field is meaningful. F3_ECNT means how many errors depending on which ECC is used.
[17:16]	**F3_STAT**	**ECC Status of Field 3** This field contains the ECC correction status (BCH algorithm) of ECC-field 3. 00 = No error. 01 = Correctable error. 10 = Uncorrectable error. 11 = Reserved.
[15]	**Reserved**	Reserved.

[14:10]	**F2_ECNT**	**Error Count of ECC Field 2** This field contains the error counts after ECC correct calculation of Field 2. For this ECC core (BCH algorithm), only when F2_STAT equals to 0x01, the value in this field is meaningful. F2_ECNT means how many errors depending on which ECC is used.
[9:8]	**F2_STAT**	**ECC Status of Field 2** This field contains the ECC correction status (BCH algorithm) of ECC-field 2. 00 = No error. 01 = Correctable error. 10 = Uncorrectable error. 11 = Reserved.
[7]	**Reserved**	Reserved.
[6:2]	**F1_ECNT**	**Error Count of ECC Field 1** This field contains the error counts after ECC correct calculation of Field 1. For this ECC core (BCH algorithm), only when F1_STAT equals to 0x01, the value in this field is meaningful. F1_ECNT means how many errors depending on which ECC is used.
[1:0]	**F1_STAT**	**ECC Status of Field 1** This field contains the ECC correction status (BCH algorithm) of ECC-field 1. 00 = No error. 01 = Correctable error. 10 = Uncorrectable error. 11 = Reserved.

NAND Flash ECC Error Status 1 Register (FMI_NANDECCES1)

Register	Offset	R/W	Description	Reset Value
FMI_NANDECCES1	FMI_BA+0x8D4	R	NAND Flash ECC Error Status 1 Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	F8_ECNT					F8_STAT	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	F7_ECNT					F7_STAT	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	F6_ECNT					F6_STAT	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	F5_ECNT					F5_STAT	

Bits	Description	
[31]	**Reserved**	Reserved.
[30:26]	**F8_ECNT**	**Error Count of ECC Field 8** This field contains the error counts after ECC correct calculation of Field 8. For this ECC core (BCH algorithm), only when F8_STAT equals to 0x01, the value in this field is meaningful. F8_ECNT means how many errors depending on which ECC is used.
[25:24]	**F8_STAT**	**ECC Status of Field 8** This field contains the ECC correction status (BCH algorithm) of ECC-field 8. 00 = No error. 01 = Correctable error. 10 = Uncorrectable error. 11 = Reserved.
[23]	**Reserved**	Reserved.
[22:18]	**F7_ECNT**	**Error Count of ECC Field 7** This field contains the error counts after ECC correct calculation of Field 7. For this ECC core (BCH algorithm), only when F7_STAT equals to 0x01, the value in this field is meaningful. F7_ECNT means how many errors depending on which ECC is used.
[17:16]	**F7_STAT**	**ECC Status of Field 7** This field contains the ECC correction status (BCH algorithm) of ECC-field 7. 00 = No error. 01 = Correctable error. 10 = Uncorrectable error. 11 = Reserved.
[15]	**Reserved**	Reserved.

[14:10]	**F6_ECNT**	**Error Count of ECC Field 6** This field contains the error counts after ECC correct calculation of Field 6. For this ECC core (BCH algorithm), only when F6_STAT equals to 0x01, the value in this field is meaningful. F6_ECNT means how many errors depending on which ECC is used.
[9:8]	**F6_STAT**	**ECC Status of Field 6** This field contains the ECC correction status (BCH algorithm) of ECC-field 6. 00 = No error. 01 = Correctable error. 10 = Uncorrectable error. 11 = Reserved.
[7]	**Reserved**	Reserved.
[6:2]	**F5_ECNT**	**Error Count of ECC Field 5** This field contains the error counts after ECC correct calculation of Field 5. For this ECC core (BCH algorithm), only when F5_STAT equals to 0x01, the value in this field is meaningful. F5_ECNT means how many errors depending on which ECC is used.
[1:0]	**F5_STAT**	**ECC Status of Field 5** This field contains the ECC correction status (BCH algorithm) of ECC-field5. 00 = No error. 01 = Correctable error. 10 = Uncorrectable error. 11 = Reserved.

NAND Flash ECC Error Status 2 Register (FMI_NANDECCES2)

Register	Offset	R/W	Description	Reset Value
FMI_NANDECCES2	FMI_BA+0x8D8	R	NAND Flash ECC Error Status 2 Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	F12_ECNT					F12_STAT	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	F11_ECNT					F11_STAT	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	F10_ECNT					F10_STAT	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	F9_ECNT					F9_STAT	

Bits	Description	
[31]	**Reserved**	Reserved.
[30:26]	**F12_ECNT**	**Error Count of ECC Field 12** This field contains the error counts after ECC correct calculation of Field 12. For this ECC core (BCH algorithm), only when F12_STAT equals to 0x01, the value in this field is meaningful. F12_ECNT means how many errors depending on which ECC is used.
[25:24]	**F12_STAT**	**ECC Status of Field 12** This field contains the ECC correction status (BCH algorithm) of ECC-field 12. 00 = No error. 01 = Correctable error. 10 = Uncorrectable error. 11 = Reserved.
[23]	**Reserved**	Reserved.
[22:18]	**F11_ECNT**	**Error Count of ECC Field 11** This field contains the error counts after ECC correct calculation of Field 11. For this ECC core (BCH algorithm), only when F11_STAT equals to 0x01, the value in this field is meaningful. F11_ECNT means how many errors depending on which ECC is used.
[17:16]	**F11_STAT**	**ECC Status of Field 11** This field contains the ECC correction status (BCH algorithm) of ECC-field 11. 00 = No error. 01 = Correctable error. 10 = Uncorrectable error. 11 = Reserved.
[15]	**Reserved**	Reserved.

[14:10]	**F10_ECNT**	**Error Count of ECC Field 10** This field contains the error counts after ECC correct calculation of Field 10. For this ECC core (BCH algorithm), only when F10_STAT equals to 0x01, the value in this field is meaningful. F10_ECNT means how many errors depending on which ECC is used.
[9:8]	**F10_STAT**	**ECC Status of Field 10** This field contains the ECC correction status (BCH algorithm) of ECC-field 10. 00 = No error. 01 = Correctable error. 10 = Uncorrectable error. 11 = Reserved.
[7]	**Reserved**	Reserved.
[6:2]	**F9_ECNT**	**Error Count of ECC Field 9** This field contains the error counts after ECC correct calculation of Field 9. For this ECC core (BCH algorithm), only when F9_STAT equals to 0x01, the value in this field is meaningful. F9_ECNT means how many errors depending on which ECC is used.
[1:0]	**F9_STAT**	**ECC Status of Field 9** This field contains the ECC correction status (BCH algorithm) of ECC-field 9. 00 = No error. 01 = Correctable error. 10 = Uncorrectable error. 11 = Reserved.

NAND Flash ECC Error Status 3 Register (FMI_NANDECCES3)

Register	Offset	R/W	Description	Reset Value
FMI_NANDECCES3	FMI_BA+0x8DC	R	NAND Flash ECC Error Status 3 Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	F16_ECNT					F16_STAT	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	F15_ECNT					F15_STAT	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	F14_ECNT					F14_STAT	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	F13_ECNT					F13_STAT	

Bits	Description	
[31]	**Reserved**	Reserved.
[30:26]	**F16_ECNT**	**Error Count of ECC Field 16** This field contains the error counts after ECC correct calculation of Field 16. For this ECC core (BCH algorithm), only when F16_STAT equals to 0x01, the value in this field is meaningful. F16_ECNT means how many errors depending on which ECC is used.
[25:24]	**F16_STAT**	**ECC Status of Field 16** This field contains the ECC correction status (BCH algorithm) of ECC-field 16. 00 = No error. 01 = Correctable error. 10 = Uncorrectable error. 11 = Reserved.
[23]	**Reserved**	Reserved.
[22:18]	**F15_ECNT**	**Error Count of ECC Field 15** This field contains the error counts after ECC correct calculation of Field 15. For this ECC core (BCH algorithm), only when F15_STAT equals to 0x01, the value in this field is meaningful. F15_ECNT means how many errors depending on which ECC is used.
[17:16]	**F15_STAT**	**ECC Status of Field 15** This field contains the ECC correction status (BCH algorithm) of ECC-field 15. 00 = No error. 01 = Correctable error. 10 = Uncorrectable error. 11 = Reserved.
[15]	**Reserved**	Reserved.

[14:10]	**F14_ECNT**	**Error Count of ECC Field 14** This field contains the error counts after ECC correct calculation of Field 14. For this ECC core (BCH algorithm), only when F14_STAT equals to 0x01, the value in this field is meaningful. F14_ECNT means how many errors depending on which ECC is used.
[9:8]	**F14_STAT**	**ECC Status of Field 14** This field contains the ECC correction status (BCH algorithm) of ECC-field 14. 00 = No error. 01 = Correctable error. 10 = Uncorrectable error. 11 = Reserved.
[7]	**Reserved**	Reserved.
[6:2]	**F13_ECNT**	**Error Count of ECC Field 13** This field contains the error counts after ECC correct calculation of Field 13. For this ECC core (BCH algorithm), only when F13_STAT equals to 0x01, the value in this field is meaningful. F13_ECNT means how many errors depending on which ECC is used.
[1:0]	**F13_STAT**	**ECC Status of Field 13** This field contains the ECC correction status (BCH algorithm) of ECC-field 13. 00 = No error. 01 = Correctable error. 10 = Uncorrectable error. 11 = Reserved.

NAND Flash ECC Error Byte Address 0 Register (FMI_NANDECCEA0)

Register	Offset	R/W	Description	Reset Value
FMI_NANDECCEA0	FMI_BA+0x900	R	NAND Flash ECC Error Byte Address 0 Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					ERR_ADDR1		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ERR_ADDR1							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					ERR_ADDR0		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ERR_ADDR0							

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:16]	**ERR_ADDR1**	**ECC Error Address First Field of Error 1** This field contains an 11-bit ECC error address 1 of first field. If it is a correctable error, please read the error data, ERR_DATA1 (FMI_NANDECCED0[15:8]), to correct this error.
[15:11]	**Reserved**	Reserved.
[10:0]	**ERR_ADDR0**	**ECC Error Address First Field of Error 0** This field contains an 11-bit ECC error address 0 of first field. If it is a correctable error, please read the error data, ERR_DATA0 (FMI_NANDECCED0[7:0]), to correct this error.

NAND Flash ECC Error Byte Address 1 Register (FMI_NANDECCEA1)

Register	Offset	R/W	Description	Reset Value
FMI_NANDECCEA1	FMI_BA+0x904	R	NAND Flash ECC Error Byte Address 1 Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					**ERR_ADDR3**		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ERR_ADDR3							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**ERR_ADDR2**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ERR_ADDR2							

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:16]	**ERR_ADDR3**	**ECC Error Address First Field of Error 3** This field contains an 11-bit ECC error address 3 of first field. If it is a correctable error, please read the error data, ERR_DATA3 (FMI_NANDECCED0[31:24]), to correct this error.
[15:11]	**Reserved**	Reserved.
[10:0]	**ERR_ADDR2**	**ECC Error Address First Field of Error 2** This field contains an 11-bit ECC error address 2 of first field. If it is a correctable error, please read the error data, ERR_DATA2 (FMI_NANDECCED0[23:16]), to correct this error.

NAND Flash ECC Error Byte Address 2 Register (FMI_NANDECCEA2)

Register	Offset	R/W	Description	Reset Value
FMI_NANDECCEA2	FMI_BA+0x908	R	NAND Flash ECC Error Byte Address 2 Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					ERR_ADDR5		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ERR_ADDR5							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					ERR_ADDR4		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ERR_ADDR4							

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:16]	**ERR_ADDR5**	**ECC Error Address First Field of Error 5** This field contains an 11-bit ECC error address 5 of first field. If it is a correctable error, please read the error data, ERR_DATA5 (FMI_NANDECCED1[15:8]), to correct this error.
[15:11]	**Reserved**	Reserved.
[10:0]	**ERR_ADDR4**	**ECC Error Address First Field of Error 4** This field contains an 11-bit ECC error address 4 of first field. If it is a correctable error, please read the error data, ERR_DATA4 (FMI_NANDECCED1[7:0]), to correct this error.

NAND Flash ECC Error Byte Address 3 Register (FMI_NANDECCEA3)

Register	Offset	R/W	Description	Reset Value
FMI_NANDECCEA3	FMI_BA+0x90C	R	NAND Flash ECC Error Byte Address 3 Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					**ERR_ADD7**		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ERR_ADDR7							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**ERR_ADDR6**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ERR_ADDR6							

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:16]	**ERR_ADDR7**	**ECC Error Address First Field of Error 7** This field contains an 11-bit ECC error address 7 of first field. If it is a correctable error, please read the error data, ERR_DATA7 (FMI_NANDECCED1[31:24]), to correct this error.
[15:11]	**Reserved**	Reserved.
[10:0]	**ERR_ADDR6**	**ECC Error Address First Field of Error 6** This field contains an 11-bit ECC error address 6 of first field. If it is a correctable error, please read the error data, ERR_DATA6 (FMI_NANDECCED1[23:16]), to correct this error.

NAND Flash ECC Error Byte Address 4 Register (FMI_NANDECCEA4)

Register	Offset	R/W	Description	Reset Value
FMI_NANDECCEA4	FMI_BA+0x910	R	NAND Flash ECC Error Byte Address 4 Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					ERR_ADDR9		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ERR_ADDR9							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					ERR_ADDR8		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ERR_ADDR8							

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:16]	**ERR_ADDR9**	**ECC Error Address First Field of Error 9** This field contains an 11-bit ECC error address 9 of first field. If it is a correctable error, please read the error data, ERR_DATA9 (FMI_NANDECCED2[15:8]), to correct this error.
[15:11]	**Reserved**	Reserved.
[10:0]	**ERR_ADDR8**	**ECC Error Address First Field of Error 8** This field contains an 11-bit ECC error address 8 of first field. If it is a correctable error, please read the error data, ERR_DATA8 (FMI_NANDECCED2[7:0]), to correct this error.

NAND Flash ECC Error Byte Address 5 Register (FMI_NANDECCEA5)

Register	Offset	R/W	Description	Reset Value
FMI_NANDECCEA5	FMI_BA+0x914	R	NAND Flash ECC Error Byte Address 5 Register	0x0000_0000

<table>
<tr><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th></tr>
<tr><td colspan="5">Reserved</td><td colspan="3">ERR_ADDR11</td></tr>
<tr><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th></tr>
<tr><td colspan="8">ERR_ADDR11</td></tr>
<tr><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th></tr>
<tr><td colspan="5">Reserved</td><td colspan="3">ERR_ADDR10</td></tr>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">ERR_ADDR10</td></tr>
</table>

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:16]	**ERR_ADDR11**	**ECC Error Address First Field of Error 11** This field contains an 11-bit ECC error address 11 of first field. If it is a correctable error, please read the error data, ERR_DATA11 (FMI_NANDECCED2[31:24]), to correct this error.
[15:11]	**Reserved**	Reserved.
[10:0]	**ERR_ADDR10**	**ECC Error Address First Field of Error 10** This field contains an 11-bit ECC error address 10 of first field. If it is a correctable error, please read the error data, ERR_DATA10 (FMI_NANDECCED2[23:16]), to correct this error.

NAND Flash ECC Error Byte Address 6 Register (FMI_NANDECCEA6)

Register	Offset	R/W	Description	Reset Value
FMI_NANDECCEA6	FMI_BA+0x918	R	NAND Flash ECC Error Byte Address 6 Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					ERR_ADDR13		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ERR_ADDR13							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					ERR_ADDR12		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ERR_ADDR12							

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:16]	**ERR_ADDR13**	**ECC Error Address First Field of Error 13** This field contains an 11-bit ECC error address 13 of first field. If it is a correctable error, please read the error data, ERR_DATA13 (FMI_NANDECCED3[15:8]), to correct this error.
[15:11]	**Reserved**	Reserved.
[10:0]	**ERR_ADDR12**	**ECC Error Address First Field of Error 12** This field contains an 11-bit ECC error address 12 of first field. If it is a correctable error, please read the error data, ERR_DATA12 (FMI_NANDECCED3[7:0]), to correct this error.

NAND Flash ECC Error Byte Address 7 Register (FMI_NANDECCEA7)

Register	Offset	R/W	Description	Reset Value
FMI_NANDECCEA7	FMI_BA+0x91C	R	NAND Flash ECC Error Byte Address 7 Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					ERR_ADDR15		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ERR_ADDR15							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					ERR_ADDR14		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ERR_ADDR14							

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:16]	**ERR_ADDR15**	**ECC Error Address First Field of Error 15** This field contains an 11-bit ECC error address 15 of first field. If it is a correctable error, please read the error data, ERR_DATA15 (FMI_NANDECCED3[31:24]), to correct this error.
[15:11]	**Reserved**	Reserved.
[10:0]	**ERR_ADDR14**	**ECC Error Address First Field of Error 14** This field contains an 11-bit ECC error address 14 of first field. If it is a correctable error, please read the error data, ERR_DATA14 (FMI_NANDECCED3[23:16]), to correct this error.

NAND Flash ECC Error Byte Address 8 Register (FMI_NANDECCEA8)

Register	Offset	R/W	Description	Reset Value
FMI_NANDECCEA8	FMI_BA+0x920	R	NAND Flash ECC Error Byte Address 8 Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					**ERR_ADDR17**		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ERR_ADDR17							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**ERR_ADDR16**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ERR_ADDR16							

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:16]	**ERR_ADDR17**	**ECC Error Address First Field of Error 17** This field contains an 11-bit ECC error address 17 of first field. If it is a correctable error, please read the error data, ERR_DATA17 (FMI_NANDECCED4[15:8]), to correct this error.
[15:11]	**Reserved**	Reserved.
[10:0]	**ERR_ADDR16**	**ECC Error Address First Field of Error 16** This field contains an 11-bit ECC error address 16 of first field. If it is a correctable error, please read the error data, ERR_DATA16 (FMI_NANDECCED4[7:0]), to correct this error.

NAND Flash ECC Error Byte Address 9 Register (FMI_NANDECCEA9)

Register	Offset	R/W	Description	Reset Value
FMI_NANDECCEA9	FMI_BA+0x924	R	NAND Flash ECC Error Byte Address 9 Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					ERR_ADDR19		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ERR_ADDR19							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					ERR_ADDR18		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ERR_ADDR18							

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:16]	**ERR_ADDR19**	**ECC Error Address First Field of Error 19** This field contains an 11-bit ECC error address 19 of first field. If it is a correctable error, please read the error data, ERR_DATA19 (FMI_NANDECCED4[31:24]), to correct this error.
[15:11]	**Reserved**	Reserved.
[10:0]	**ERR_ADDR18**	**ECC Error Address First Field of Error 18** This field contains an 11-bit ECC error address 18 of first field. If it is a correctable error, please read the error data, ERR_DATA18 (FMI_NANDECCED4[23:16]), to correct this error.

NAND Flash ECC Error Byte Address 10 Register (FMI_NANDECCEA10)

Register	Offset	R/W	Description	Reset Value
FMI_NANDECCEA10	FMI_BA+0x928	R	NAND Flash ECC Error Byte Address 10 Register	0x0000_0000

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**
Reserved					**ERR_ADDR21**		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ERR_ADDR21							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**ERR_ADDR20**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ERR_ADDR20							

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:16]	**ERR_ADDR21**	**ECC Error Address First Field of Error 21** This field contains an 11-bit ECC error address 21 of first field. If it is a correctable error, please read the error data, ERR_DATA21 (FMI_NANDECCED5[15:8]), to correct this error.
[15:11]	**Reserved**	Reserved.
[10:0]	**ERR_ADDR20**	**ECC Error Address First Field of Error 20** This field contains an 11-bit ECC error address 20 of first field. If it is a correctable error, please read the error data, ERR_DATA20 (FMI_NANDECCED5[7:0]), to correct this error.

NAND Flash ECC Error Byte Address 11 Register (FMI_NANDECCEA11)

Register	Offset	R/W	Description	Reset Value
FMI_NANDECCEA11	FMI_BA+0x92C	R	NAND Flash ECC Error Byte Address 11 Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					**ERR_ADDR23**		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ERR_ADDR23							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**ERR_ADDR22**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ERR_ADDR22							

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:16]	**ERR_ADDR23**	**ECC Error Address First Field of Error 23** This field contains an 11-bit ECC error address 23 of first field. If it is a correctable error, please read the error data, ERR_DATA23 (FMI_NANDECCED5[31:24]), to correct this error.
[15:11]	**Reserved**	Reserved.
[10:0]	**ERR_ADDR22**	**ECC Error Address First Field of Error 22** This field contains an 11-bit ECC error address 22 of first field. If it is a correctable error, please read the error data, ERR_DATA22 (FMI_NANDECCED5[23:16]), to correct this error.

NAND Flash ECC Error Data Register 0 (FMI_NANDECCED0)

Register	Offset	R/W	Description	Reset Value
FMI_NANDECCED0	FMI_BA+0x960	R	NAND Flash ECC Error Data Register 0	0x8080_8080

31	30	29	28	27	26	25	24
ERR_DATA3							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ERR_DATA2							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ERR_DATA1							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ERR_DATA0							

Bits	Description	
[31:24]	**ERR_DATA3**	**ECC Error Data of First Field 3** This field contains an 8-bit BCH ECC error data 3 of first field. If it is a correctable error, please read out the error data in this field and doing bitwise XOR with received data locating at address ERR_ADDR3 (FMI_NANDECCEA1[26:16]), and then the result will be the correct data.
[23:16]	**ERR_DATA2**	**ECC Error Data of First Field 2** This field contains an 8-bit BCH ECC error data 2 of first field. If it is a correctable error, please read out the error data in this field and doing bitwise XOR with received data locating at address ERR_ADDR2 (FMI_NANDECCEA1[10:0]), and then the result will be the correct data.
[15:8]	**ERR_DATA1**	**ECC Error Data of First Field 1** This field contains an 8-bit BCH ECC error data 1 of first field. If it is a correctable error, please read out the error data in this field and doing bitwise XOR with received data locating at address ERR_ADDR1 (FMI_NANDECCEA0[26:16]), and then the result will be the correct data.
[7:0]	**ERR_DATA0**	**ECC Error Data of First Field 0** This field contains an 8-bit BCH ECC error data 0 of first field. If it is a correctable error, please read out the error data in this field and doing bitwise XOR with received data locating at address ERR_ADDR0 (FMI_ANNDECCEA0[10:0]), and then the result will be the correct data.

NAND Flash ECC Error Data Register 1 (FMI_NANDECCED1)

Register	Offset	R/W	Description	Reset Value
FMI_NANDECCED1	FMI_BA+0x964	R	NAND Flash ECC Error Data Register 1	0x8080_8080

31	30	29	28	27	26	25	24
ERR_DATA7							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ERR_DATA6							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ERR_DATA5							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ERR_DATA4							

Bits	Description	
[31:24]	**ERR_DATA7**	**ECC Error Data of First Field 7** This field contains an 8-bit BCH ECC error data 7 of first field. If it is a correctable error, please read out the error data in this field and doing bitwise XOR with received data locating at address ERR_ADDR7 (FMI_NANDECCEA3[26:16]), and then the result will be the correct data.
[23:16]	**ERR_DATA6**	**ECC Error Data of First Field 6** This field contains an 8-bit BCH ECC error data 6 of first field. If it is a correctable error, please read out the error data in this field and doing bitwise XOR with received data locating at address ERR_ADDR6 (FMI_NANDECCEA3[10:0]), and then the result will be the correct data.
[15:8]	**ERR_DATA5**	**ECC Error Data of First Field 5** This field contains an 8-bit BCH ECC error data 5 of first field. If it is a correctable error, please read out the error data in this field and doing bitwise XOR with received data locating at address ERR_ADDR5 (FMI_NANDECCEA2[26:16]), and then the result will be the correct data.
[7:0]	**ERR_DATA4**	**ECC Error Data of First Field 4** This field contains an 8-bit BCH ECC error data 4 of first field. If it is a correctable error, please read out the error data in this field and doing bitwise XOR with received data locating at address ERR_ADDR4 (FMI_NANDECCEA2[10:0]), and then the result will be the correct data.

NAND Flash ECC Error Data Register 2 (FMI_NANDECCED2)

Register	Offset	R/W	Description	Reset Value
FMI_NANDECCED2	FMI_BA+0x968	R	NAND Flash ECC Error Data Register 2	0x8080_8080

31	30	29	28	27	26	25	24
ERR_DATA11							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ERR_DATA10							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ERR_DATA9							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ERR_DATA8							

Bits	Description	
[31:24]	**ERR_DATA11**	**ECC Error Data of First Field 11** This field contains an 8-bit BCH ECC error data 11 of first field. If it is a correctable error, please read out the error data in this field and doing bitwise XOR with received data locating at address ERR_ADDR11 (FMI_NANDECCEA5[26:16]), and then the result will be the correct data.
[23:16]	**ERR_DATA10**	**ECC Error Data of First Field 10** This field contains an 8-bit BCH ECC error data 10 of first field. If it is a correctable error, please read out the error data in this field and doing bitwise XOR with received data locating at address ERR_ADDR10 (FMI_NANDECCEA5[10:0]), and then the result will be the correct data.
[15:8]	**ERR_DATA9**	**ECC Error Data of First Field 9** This field contains an 8-bit BCH ECC error data 9 of first field. If it is a correctable error, please read out the error data in this field and doing bitwise XOR with received data locating at address ERR_ADDR9 (FMI_NANDECCEA4[26:16]), and then the result will be the correct data.
[7:0]	**ERR_DATA8**	**ECC Error Data of First Field 8** This field contains an 8-bit BCH ECC error data 8 of first field. If it is a correctable error, please read out the error data in this field and doing bitwise XOR with received data locating at address ERR_ADDR8 (FMI_NANDECCEA4[10:0]), and then the result will be the correct data.

NAND Flash ECC Error Data Register 3 (FMI_NANDECCED3)

Register	Offset	R/W	Description	Reset Value
FMI_NANDECCED3	FMI_BA+0x96C	R	NAND Flash ECC Error Data Register 3	0x8080_8080

31	30	29	28	27	26	25	24
ERR_DATA15							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ERR_DATA14							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ERR_DATA13							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ERR_DATA12							

Bits	Description	
[31:24]	**ERR_DATA15**	**ECC Error Data of First Field 15** This field contains an 8-bit BCH ECC error data 15 of first field. If it is a correctable error, please read out the error data in this field and doing bitwise XOR with received data locating at address ERR_ADDR15 (FMI_NANDECCEA7[26:16]), and then the result will be the correct data.
[23:16]	**ERR_DATA14**	**ECC Error Data of First Field 14** This field contains an 8-bit BCH ECC error data 14 of first field. If it is a correctable error, please read out the error data in this field and doing bitwise XOR with received data locating at address ERR_ADDR14 (FMI_NANDECCEA7[10:0]), and then the result will be the correct data.
[15:8]	**ERR_DATA13**	**ECC Error Data of First Field 13** This field contains an 8-bit BCH ECC error data 13 of first field. If it is a correctable error, please read out the error data in this field and doing bitwise XOR with received data locating at address ERR_ADDR13 (FMI_NANDECCEA6[26:16]), and then the result will be the correct data.
[7:0]	**ERR_DATA12**	**ECC Error Data of First Field 12** This field contains an 8-bit BCH ECC error data 12 of first field. If it is a correctable error, please read out the error data in this field and doing bitwise XOR with received data locating at address ERR_ADDR12 (FMI_NANDECCEA6[10:0]), and then the result will be the correct data.

NAND Flash ECC Error Data Register 4 (FMI_NANDECCED4)

Register	Offset	R/W	Description	Reset Value
FMI_NANDECCED4	FMI_BA+0x970	R	NAND Flash ECC Error Data Register 4	0x8080_8080

31	30	29	28	27	26	25	24
ERR_DATA19							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ERR_DATA18							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ERR_DATA17							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ERR_DATA16							

Bits	Description	
[31:24]	**ERR_DATA19**	**ECC Error Data of First Field 19** This field contains an 8-bit BCH ECC error data 19 of first field. If it is a correctable error, please read out the error data in this field and doing bitwise XOR with received data locating at address ERR_ADDR19 (FMI_NANDECCEA9[26:16]), and then the result will be the correct data.
[23:16]	**ERR_DATA18**	**ECC Error Data of First Field 18** This field contains an 8-bit BCH ECC error data 18 of first field. If it is a correctable error, please read out the error data in this field and doing bitwise XOR with received data locating at address ERR_ADDR18 (FMI_NANDECCEA9[10:0]), and then the result will be the correct data.
[15:8]	**ERR_DATA17**	**ECC Error Data of First Field 17** This field contains an 8-bit BCH ECC error data 17 of first field. If it is a correctable error, please read out the error data in this field and doing bitwise XOR with received data locating at address ERR_ADDR17 (FMI_NANDECCEA8[26:16]), and then the result will be the correct data.
[7:0]	**ERR_DATA16**	**ECC Error Data of First Field 16** This field contains an 8-bit BCH ECC error data 16 of first field. If it is a correctable error, please read out the error data in this field and doing bitwise XOR with received data locating at address ERR_ADDR16 (FMI_NANDECCEA8[10:0]), and then the result will be the correct data.

NAND Flash ECC Error Data Register 5 (FMI_NANDECCED5)

Register	Offset	R/W	Description	Reset Value
FMI_NANDECCED5	FMI_BA+0x974	R	NAND Flash ECC Error Data Register 5	0x8080_8080

31	30	29	28	27	26	25	24
ERR_DATA23							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ERR_DATA22							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ERR_DATA21							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ERR_DATA20							

Bits	Description	
[31:24]	**ERR_DATA23**	**ECC Error Data of First Field 23** This field contains an 8-bit BCH ECC error data 23 of first field. If it is a correctable error, please read out the error data in this field and doing bitwise XOR with received data locating at address ERR_ADDR23 (FMI_NANDECCEA11[26:16]), and then the result will be the correct data.
[23:16]	**ERR_DATA22**	**ECC Error Data of First Field 22** This field contains an 8-bit BCH ECC error data 22 of first field. If it is a correctable error, please read out the error data in this field and doing bitwise XOR with received data locating at address ERR_ADDR22 (FMI_NANDECCEA11[10:0]), and then the result will be the correct data.
[15:8]	**ERR_DATA21**	**ECC Error Data of First Field 21** This field contains an 8-bit BCH ECC error data 21 of first field. If it is a correctable error, please read out the error data in this field and doing bitwise XOR with received data locating at address ERR_ADDR21 (FMI_NANDECCEA10[26:16]), and then the result will be the correct data.
[7:0]	**ERR_DATA20**	**ECC Error Data of First Field 20** This field contains an 8-bit BCH ECC error data 20 of first field. If it is a correctable error, please read out the error data in this field and doing bitwise XOR with received data locating at address ERR_ADDR20 (FMI_NANDECCEA10[10:0]), and then the result will be the correct data.

6.25 Secure Digital Host Controller (SDH)

6.25.1 Overview

The Secure Digital Host Controller (SD Host) has DMAC unit and SD unit. The DMAC unit provides a DMA (Direct Memory Access) function for SD to exchange data between system memory and shared buffer (128 bytes), and the SD unit controls the interface of SD/SDHC. The SD Host Controller can support SD/SDHC and cooperated with DMAC to provide a fast data transfer between system memory and cards.

6.25.2 Features

- AMBA AHB master/slave interface compatible, for data transfer and register read/write.
- Supports single DMA channel.
- Supports hardware Scatter-Gather function..
- Using single 128 Bytes shared buffer for data exchange between system memory and cards.
- Synchronous design for DMA with single clock domain, AHB bus clock (HCLK).
- Interface with DMAC for register read/write and data transfer.
- Supports SD/SDHC card.
- Completely asynchronous design for Secure Digital with two clock domains, HCLK and Engine clock, note that frequency of HCLK should be higher than the frequency of peripheral clock.

6.25.3 Block Diagram

The block diagram and Card Pad Assignment of SD Host Controller is shown as follows.

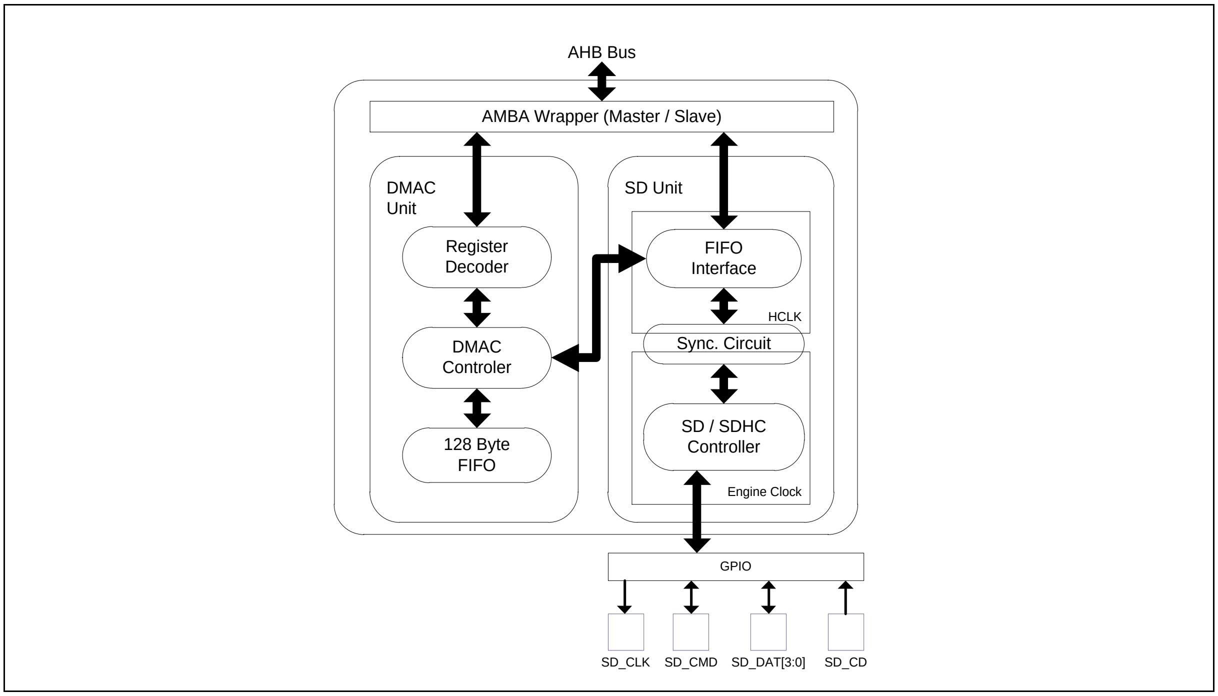


Figure 6.25-1 SD Host Controller Block Diagram

6.25.4 Basic Configuration

6.25.4.1 SD1 Basic Configuration

- Clock source Configuration
  - Select the source clock divide number for clock divider of APLL and UPLL on SDH_SDIV (CLK_DIVCTL9[2:0]).
  - Select the source of SD1 engine clock on SDH_S (CLK_DIVCTL9[4:3]).
  - Select the clock divider number of SD1 engine clock on SDH_N (CLK_DIVCTL9[15:8]).
  - Enable SD1 engine clock in SDH (CLK_HCLKEN[30]).
- Reset Configuration
  - Reset SD1 controller in SDHRST (SYS_AHBIPRST[24]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
SD1	SD1_CMD	PF.0	MFP2
	SD1_CLK	PF.1	MFP2
	SD1_DAT0	PF.2	MFP2
	SD1_DAT1	PF.3	MFP2
	SD1_DAT2	PF.4	MFP2
	SD1_DAT3	PF.5	MFP2
	SD1_nCD	PF.6	MFP2

Table 6.25-1 SD1 Pin Configuration

6.25.4.2 eMMC1 Basic Configuration

- Clock source Configuration
  - Select the source clock divide number for clock divider of APLL and UPLL on SDH_SDIV (CLK_DIVCTL9[2:0]).
  - Select the source of eMMC1 engine clock on SDH_S (CLK_DIVCTL9[4:3]).
  - Select the clock divider number of eMMC1 engine clock on SDH_N (CLK_DIVCTL9[15:8]).
  - Enable eMMC1 engine clock in SDH (CLK_HCLKEN[30]).
- Reset Configuration
  - Reset eMMC1 controller in SDHRST (SYS_AHBIPRST[24]).
- Pin Configuration

Group	Pin Name	GPIO	MFP
eMMC1	eMMC1_CMD	PF.0	MFP2
	eMMC1_CLK	PF.1	MFP2
	eMMC1_DATA0	PF.2	MFP2

	eMMC1_DATA1	PF.3	MFP2
	eMMC1_DATA2	PF.4	MFP2
	eMMC1_DATA3	PF.5	MFP2

Table 2 eMMC1 Pin Configuration

6.25.5 Functional Description

The SD1/eMMC1 controller provides an interface for SD/SDHC card access with 1-bit/4-bit data bus width.

The SD1/eMMC1 controller uses an independent clock source named SDCLK as engine clock. SDCLK can be completely asynchronous with system clock HCLK, software can change SD clock arbitrary. However, HCLK should be faster than SDCLK.

6.25.5.1 Basic Operation

This SD1/eMMC1 controller can generate all types of 48-bit command to the SD card and retrieve all types of response from SD card. After response in, the content of response will be stored at SDRSP0 and SDRSP1. SD1/eMMC1 controller will calculate CRC7 and check its correctness for response. If CRC7 is error, CRCIF (SDH_INTSTS[1]) will be set and CRC7 (SDH_INTSTS[2]) will be ‘0’. For response R1b, software should notice that after response in, SD card will put busy signal on data line DAT0; software should check this status with clock polling until it became high. For response R3, CRC7 is invalid; but SD1/eMMC1 controller will still calculate CRC7 and get an error result, software should ignore this error and clear CRCIF flag (SDH_INTSTS[1]).

The SD1/eMMC1 controller is composed of two state machines – command/response part and data part. For command/response part, the trigger bits are COEN, RIEN, R2EN, CLK74OE and CLK8OE in SDH_CTL register. If software enables all of these bits, the execution priority will be CLK74OE > COEN > RIEN/R2EN > CLK8OE, note that RIEN and R2EN can’t be triggered at the same time. For data part, there are DIEN and DOEN for selection. Software can only trigger one of them at one time. If DIEN is triggered, the SD1/eMMC1 controller waits start bit from data line DAT0 immediately, and then get the specified amount data from SD card. After data-in, the SD1/eMMC1 controller will check the correctness of CRC16; if it is incorrect, CRCIF (SDH_INTSTS[1]) will be set and CRC16 (SDH_INTSTS[3]) will be ‘0’. If DOEN is triggered, the SD1/eMMC1 controller will wait until response in is finished, and then send specified amount data to the SD card. After data-out, the SD1/eMMC1 controller will get CRC status from SD card and check its correctness; it should be ‘010’, otherwise CRCIF (SDH_INTSTS[1]) will be set and CRCSTS (SDH_INTSTS[6:4]) will be the value it received.

If R2EN is triggered, the SD1/eMMC1 controller will receive response R2 (136 bits) from SD card, CRC7 and end bit will be dropped. The receiving data will be placed at DMA’s buffer, starting from address offset 0x0.

6.25.5.2 Multiple Block Transfer

The SD1/eMMC1 controller also provides multiple block transfer function (change BLKLEN to change the block length). Software can use this function to accelerate data transfer throughput. If CRC7, CRC16 or CRC status is error, SD1/eMMC1 controller will stop transfer and set CRCIF (SDH_INTSTS[1]), software should do engine reset when this situation occurred.

There is a hardware time-out mechanism for response in and data in inside SD1/eMMC1 engine. Software can specify a 24-bit time-out value at TOUT, and then SD1/eMMC1 controller will decide when to time-out according to this value.

6.25.5.3 DMA Controller

The SD host DMA controller provides a DMA (Direct Memory Access) function for SD1/eMMC1 controller to exchange data between system memory (SRAM) and shared buffer (128 bytes).

Arbitration of DMA request between SD Host is done by DMA's bus master. Software just simply fills in the starting address and enables DMA, and then let DMA to handle the data transfer automatically.

There is a 128 bytes shared buffer inside DMA, it can provide multi-block transfers for SD host. Software can access these shared buffers directly when SD host is not in busy.

6.25.5.4 Programming Flow

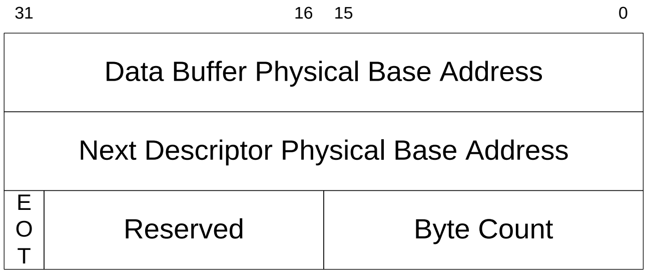
<u>Here is a simple example programming flow without DMA Scatter-Gather enable.</u>

1. Set DMAEN (SDH_DMACTL[0]) to enable DMA.
2. Fill corresponding starting address in SDH_DMASA for SD Host.
3. Trigger SD Host to start DMA transfer.
4. Wait until transfer is finished.

<u>Here is a simple example programming flow with DMA Scatter-Gather enable.</u>

1. Set DMAEN (SDH_DMACTL[0]) to enable DMA and SGEN (SDH_DMACTL[3]) to enable Scatter-Gather function.
2. Fill corresponding starting address of Physical Address Descriptor (PAD) table in SDH_DMASA for SD Host.
3. When bit-0 of SDH_DMASA is 1, the PAD will fetch in out of order, otherwise, it's fetched in order from PAD. The first time of writing bit-0 with 1 or not is not available for this function. The bits will be available in PAD table.
4. Trigger SD host to start DMA transfer.
5. Wait until transfer is finished.

The format of PAD table is shown in Figure 6.25-2 Note that the total byte count of all Pads must be equal to the byte count filled in SD host. EOT should be set to 1 in the last descriptor.


31
16
15
0
Data Buffer Physical Base Address
Next Descriptor Physical Base Address
EOT
Reserved
Byte Count

Physical Base Address: 32-bit

Byte Count: must be multiples of 4 bytes,
Max:65532 bytes

EOT: End of PAD Table

Figure 6.25-2 PAD (Physical Address Descriptor) Table Format

6.25.6 Registers Map

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

Register	Offset	R/W	Description	Reset Value
SDH Base Address: **SDH_BA = 0xB001_8000**				
SDH_FB_n **n=0,1..31**	SDH_BA+0x000 + 0x4 * n	R/W	Shared Buffer (FIFO)	0x0000_0000
SDH_DMACTL	SDH_BA+0x400	R/W	DMA Control and Status Register	0x0000_0000
SDH_DMASA	SDH_BA+0x408	R/W	DMA Transfer Starting Address Register	0x0000_0000
SDH_DMABCNT	SDH_BA+0x40C	R	DMA Transfer Byte Count Register	0x0000_0000
SDH_DMAINTEN	SDH_BA+0x410	R/W	DMA Interrupt Enable Control Register	0x0000_0001
SDH_DMAINTSTS	SDH_BA+0x414	R/W	DMA Interrupt Status Register	0x0000_0000
SDH_GCTL	SDH_BA+0x800	R/W	Global Control and Status Register	0x0000_0000
SDH_GINTEN	SDH_BA+0x804	R/W	Global Interrupt Control Register	0x0000_0001
SDH_GINTSTS	SDH_BA+0x808	R/W	Global Interrupt Status Register	0x0000_0000
SDH_CTL	SDH_BA+0x820	R/W	SD1/EMMC1 Control and Status Register	0x0101_0000
SDH_CMDARG	SDH_BA+0x824	R/W	SD1/EMMC1 Command Argument Register	0x0000_0000
SDH_INTEN	SDH_BA+0x828	R/W	SD1/EMMC1 Interrupt Control Register	0x0000_0000
SDH_INTSTS	SDH_BA+0x82C	R/W	SD1/EMMC1 Interrupt Status Register	0x000X_008C
SDH_RESP0	SDH_BA+0x830	R	SD1/EMMC1 Receiving Response Token Register 0	0x0000_0000
SDH_RESP1	SDH_BA+0x834	R	SD1/EMMC1 Receiving Response Token Register 1	0x0000_0000
SDH_BLEN	SDH_BA+0x838	R/W	SD1/EMMC1 Block Length Register	0x0000_01FF
SDH_TOUT	SDH_BA+0x83C	R/W	SD1/EMMC1 Response/Data-in Time-out Register	0x0000_0000

6.25.7 Register Description

DMA Control and Status Register (SDH_DMACTL)

Register	Offset	R/W	Description	Reset Value
SDH_DMACTL	SDH_BA+0x400	R/W	DMA Control and 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						DMABUSY	Reserved
7	6	5	4	3	2	1	0
Reserved				SGEN	Reserved	DMARST	DMAEN

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**DMABUSY**	**DMA Transfer Is in Progress** This bit indicates if SD Host is granted and doing DMA transfer or not. 0 = DMA transfer is not in progress. 1 = DMA transfer is in progress.
[8:4]	**Reserved**	Reserved.
[3]	**SGEN**	**Scatter-gather Function Enable Bit** 0 = Scatter-gather function Disabled (DMA will treat the starting address in DMASA as starting pointer of a single block memory). 1 = Scatter-gather function Enabled (DMA will treat the starting address in DMASA as a starting address of Physical Address Descriptor (PAD) table. The format of these Pads' will be described later.
[2]	**Reserved**	Reserved.
[1]	**DMARST**	**Software Engine Reset** 0 = No effect. 1 = Reset internal state machine and pointers. The contents of control register will not be cleared. This bit will auto be cleared after few clock cycles. **Note:** The software reset DMA related registers.
[0]	**DMAEN**	**DMA Controller Engine Enable Bit** 0 = DMA Controller Disabled. 1 = DMA Controller Enabled. **Note1:** If this bit is cleared, DMA will ignore all requests from SD1/eMMC1 and force bus master into IDLE state. **Note2:** If target abort occurred, DMAEN will be cleared.

DMA Transfer Starting Address Register (SDH_DMASA)

Register	Offset	R/W	Description	Reset Value
SDH_DMASA	SDH_BA+0x408	R/W	DMA Transfer Starting Address Register	0x0000_0000

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

Bits	Description	
[31:1]	**DMASA**	**DMA Transfer Starting Address** This field pads 0 as least significant bit indicates a 32-bit starting address of system memory (SRAM) for DMA to retrieve or fill in data. If DMA is not in normal mode, this field will be interpreted as a starting address of Physical Address Descriptor (PAD) table. **Note:** Starting address of the SRAM must be word aligned, for example, 0x0000_0000, 0x0000_0004.
[0]	**ORDER**	**Determined to the PAD Table Fetching Is in Order or Out of Order** 0 = PAD table is fetched in order. 1 = PAD table is fetched out of order. **Note:** the bit 0 is valid in scatter-gather mode when SGEN (SDH_DMACTL[3]) = 1.

DMA Transfer Byte Count Register (SDH_DMABCNT)

Register	Offset	R/W	Description	Reset Value
SDH_DMABCNT	SDH_BA+0x40C	R	DMA Transfer Byte Count Register	0x0000_0000

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

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25:0]	**BCNT**	**DMA Transfer Byte Count (Read Only)** This field indicates the remained byte count of DMA transfer. The value of this field is valid only when FMI is busy; otherwise, it is zero.

NUC980 SERIES TECHNICAL REFERENCE MANUAL

DMA Interrupt Enable Control Register (SDH_DMAINTEN)

Register	Offset	R/W	Description	Reset Value
SDH_DMAINTEN	SDH_BA+0x410	R/W	DMA Interrupt Enable Control Register	0x0000_0001

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

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**WEOTIEN**	**Wrong EOT (End of Transfer) Encountered Interrupt Enable Bit** 0 = Interrupt generation Disabled when wrong EOT (End of Transfer) is encountered. 1 = Interrupt generation Enabled when wrong EOT (End of Transfer) is encountered.
[0]	**ABORTIEN**	**DMA Read/Write Target Abort Interrupt Enable Bit** 0 = Target abort interrupt generation Disabled during DMA transfer. 1 = Target abort interrupt generation Enabled during DMA transfer.

DMA Interrupt Status Register (SDH_DMAINTSTS)

Register	Offset	R/W	Description	Reset Value
SDH_DMAINTSTS	SDH_BA+0x414	R/W	DMA 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						WEOTIF	ABORTIF

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**WEOTIF**	**Wrong EOT Encountered Interrupt Flag (Read Only)** When DMA Scatter-Gather function is enabled, and EOT of the descriptor is encountered before DMA transfer finished (that means the total sector count of all PAD is less than the sector count of SD host), this bit will be set. 0 = No EOT encountered before DMA transfer finished. 1 = EOT encountered before DMA transfer finished. **Note:** This bit is read only, but can be cleared by writing '1' to it.
[0]	**ABORTIF**	**DMA Read/Write Target Abort Interrupt Flag (Read Only)** 0 = No bus ERROR response received. 1 = Bus ERROR response received. **Note 1:** This bit is read only, but can be cleared by writing '1' to it. **Note 2:** When DMA's bus master received ERROR response, it means that target abort is happened. DMA will stop transfer and respond this event and then go to IDLE state. When target abort occurred or WEOTIF is set, software must reset DMA and SD1/eMMC1 Controller, and then transfer those data again.

Global Control and Status Register (SDH_GCTL)

Register	Offset	R/W	Description	Reset Value
SDH_GCTL	SDH_BA+0x800	R/W	Global Control and 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						**SDEN**	**GCTLRST**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**SDEN**	**SD1/EMMC1 Functionality Enable Bit** 0 = SD1/eMMC1 functionality Disabled. 1 = SD1/eMMC1 functionality Enabled.
[0]	**GCTLRST**	**Software Engine Reset** 0 = No effect. 1 = Reset SD1/eMMC1. The contents of control register will not be cleared. This bit will auto cleared after reset complete.

Global Interrupt Control Register (SDH_GINTEN)

Register	Offset	R/W	Description	Reset Value
SDH_GINTEN	SDH_BA+0x804	R/W	Global Interrupt 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							**DTAIEN**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**DTAIEN**	**DMA READ/WRITE Target Abort Interrupt Enable Bit** 0 = DMA READ/WRITE target abort interrupt generation Disabled. 1 = DMA READ/WRITE target abort interrupt generation Enabled.

Global Interrupt Status Register (SDH_GINTSTS)

Register	Offset	R/W	Description	Reset Value
SDH_GINTSTS	SDH_BA+0x808	R/W	Global 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							DTAIF

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**DTAIF**	**DMA READ/WRITE Target Abort Interrupt Flag (Read Only)** This bit indicates DMA received an ERROR response from internal AHB bus during DMA read/write operation. When Target Abort is occurred, please reset all engine. 0 = No bus ERROR response received. 1 = Bus ERROR response received. **Note:** This bit is read only, but can be cleared by writing ‘1’ to it.

SD1/EMMC1 Control and Status Register (SDH_CTL)

Register	Offset	R/W	Description	Reset Value
SDH_CTL	SDH_BA+0x820	R/W	SD1/EMMC1 Control and Status Register	0x0101_0000

31	30	29	28	27	26	25	24
Reserved				**SDNWR**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
BLKCNT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DBW	**CTLRST**	**CMDCODE**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CLKKEEP	**CLK8OEN**	**CLK74OEN**	**R2EN**	**DOEN**	**DIEN**	**RIEN**	**COEN**

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:24]	**SDNWR**	**NWR Parameter for Block Write Operation** This value indicates the NWR parameter for data block write operation in SD1/eMMC1 clock counts. The actual clock cycle will be SDNWR+1.
[23:16]	**BLKCNT**	**Block Counts to Be Transferred or Received** This field contains the block counts for data-in and data-out transfer. For READ_MULTIPLE_BLOCK and WRITE_MULTIPLE_BLOCK command, software can use this function to accelerate data transfer and improve performance. Don't fill 0x0 to this field. **Note:** For READ_MULTIPLE_BLOCK and WRITE_MULTIPLE_BLOCK command, the actual total length is BLKCNT * (SDH_BLEN[10:0] +1).
[15]	**DBW**	**SD1/EMMC1 Data Bus Width (for 1-bit / 4-bit Selection)** 0 = Data bus width is 1-bit. 1 = Data bus width is 4-bit.
[14]	**CTLRST**	**Software Engine Reset** 0 = Writing 0 to this bit has no effect. 1 = Reset the internal state machine and counters. The contents of control register will not be cleared (but RIEN, DIEN, DOEN and R2EN will be cleared). This bit will be auto cleared after few clock cycles.
[13:8]	**CMDCODE**	**SD1/EMMC1 Command Code** The bits contain the SD1/eMMC1 command code (0x00 – 0x3F).
[7]	**CLKKEEP**	**SD1/EMMC1 Clock Enable Control** 0 = SD1/eMMC1 decided when to output clock and when to disable clock output automatically. 1 = SD1/eMMC1 clock always keeps free running.

[6]	**CLK8OEN**	**Generating 8 Clock Cycles Output Enable Bit** 0 = No effect. (Please use DMARST (SDH_DMACTL [1]) to clear this bit.) 1 = Enabled. The SD1/eMMC1 will output 8 clock cycles. **Note:** When operation is finished, this bit will be cleared automatically, so don't write 0 to this bit (the controller will be abnormal).
[5]	**CLK74OEN**	**Initial 74 Clock Cycles Output Enable Bit** 0 = No effect. (Please use DMARST (SDH_DMACTL [1]) to clear this bit.) 1 = Enabled. The SD1/eMMC1 will output 74 clock cycles to SD card. **Note:** When operation is finished, this bit will be cleared automatically, so don't write 0 to this bit (the controller will be abnormal).
[4]	**R2EN**	**Response R2 Input Enable Bit** 0 = No effect. (Please use DMARST (SDH_DMACTL [1]) to clear this bit.) 1 = Enabled. The SD1/eMMC1 will wait to receive a response R2 from SD card and store the response data into DMAC's Flash buffer (exclude CRC7). **Note:** When operation is finished, this bit will be cleared automatically, so don't write 0 to this bit (the controller will be abnormal).
[3]	**DOEN**	**Data Output Enable Bit** 0 = No effect. (Please use DMARST (SDH_DMACTL [1]) to clear this bit.) 1 = Enabled. The SD1/eMMC1 will transfer block data and the CRC16 value to SD card. **Note:** When operation is finished, this bit will be cleared automatically, so don't write 0 to this bit (the controller will be abnormal).
[2]	**DIEN**	**Data Input Enable Bit** 0 = No effect. (Please use DMARST (SDH_DMACTL [1]) to clear this bit.) 1 = Enabled. The SD1/eMMC1 will wait to receive block data and the CRC16 value from SD card. **Note:** When operation is finished, this bit will be cleared automatically, so don't write 0 to this bit (the controller will be abnormal).
[1]	**RIEN**	**Response Input Enable Bit** 0 = No effect. (Please use DMARST (SDH_DMACTL [1]) to clear this bit.) 1 = Enabled. The SD1/eMMC1 will wait to receive a response from SD card. **Note:** When operation is finished, this bit will be cleared automatically, so don't write 0 to this bit (the controller will be abnormal).
[0]	**COEN**	**Command Output Enable Bit** 0 = No effect. (Please use DMARST (SDH_DMACTL [1]) to clear this bit.) 1 = Enabled. The SD1/eMMC1 will output a command to SD card. **Note:** When operation is finished, this bit will be cleared automatically, so don't write 0 to this bit (the controller will be abnormal).

SD1/EMMC1 Command Argument Register (SDH_CMDARG)

Register	Offset	R/W	Description	Reset Value
SDH_CMDARG	SDH_BA+0x824	R/W	SD1/EMMC1 Command Argument Register	0x0000_0000

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

Bits	Description	
[31:0]	ARGUMENT	**SD1/EMMC1 Command Argument** This register contains a 32-bit value specifies the argument of SD1/eMMC1 command from host controller to SD card. Before trigger COEN (SDH_CTL[0]), software should fill argument in this field.

SD1/EMMC1 Interrupt Control Register (SDH_INTEN)

Register	Offset	R/W	Description	Reset Value
SDH_INTEN	SDH_BA+0x828	R/W	SD1/EMMC1 Interrupt Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	CDSRC	Reserved					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	WKIEN	DITOIEN	RTOIEN	Reserved	SD1_IE	Reserved	CDIEN
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						CRCIEN	BLKDIEN

Bits	Description	
[31]	**Reserved**	Reserved.
[30]	**CDSRC**	**SD1/EMMC1 Card Detect Source Selection** 0 = From SD1/eMMC1 card’s DAT3 pin. Host need clock to got data on pin DAT3. Please make sure CLKKEEP (SDH_CTL[7]) is 1 in order to generate free running clock for DAT3 pin. 1 = From GPIO pin.
[29:15]	**Reserved**	Reserved.
[14]	**WKIEN**	**Wake-up Signal Generating Enable Bit** Enable/Disable wake-up signal generating of SD1/eMMC1 controller when card is inserted or removed. 0 = SD1/eMMC1 Card interrupt to wake-up chip Disabled. 1 = SD1/eMMC1 Card interrupt to wake-up chip Enabled.
[13]	**DITOIEN**	**Data Input Time-out Interrupt Enable Bit** Enable/Disable interrupts generation of SD1/eMMC1 controller when data input time-out. Time-out value is specified at TOUT (SDH_TOUT[23:0]). 0 = DITOIF (SDH_INTSTS[13]) trigger interrupt Disabled. 1 = DITOIF (SDH_INTSTS [13]) trigger interrupt Enabled.
[12]	**RTOIEN**	**Response Time-out Interrupt Enable Bit** Enable/Disable interrupts generation of SD1/eMMC1 controller when receiving response or R2 time-out. Time-out value is specified at TOUT (SDH_TOUT[23:0]). 0 = RTOIF (SDH_INTSTS [12]) trigger interrupt Disabled. 1 = RTOIF (SDH_INTSTS [12]) trigger interrupt Enabled.
[11]	**Reserved**	Reserved.

[10]	**SD1_IE**	**SD1/EMMC1 Interrupt Enable for Port** Enable/Disable interrupts generation of SD1/eMMC1 host when SD card issues an interrupt via DAT [1] to host. 0 = SD1_IF (SDH_INTSTS [10]) trigger interrupt Disabled. 1 = SD1_IF (SDH_INTSTS [10]) trigger interrupt Enabled.
[9]	**Reserved**	Reserved.
[8]	**CDIEN**	**SD1/EMMC1 Card Detection Interrupt Enable Bit** Enable/Disable interrupts generation of SD1/eMMC1 controller when card is inserted or removed. 0 = CDIF (SDH_INTSTS [8]) trigger interrupt Disabled. 1 = CDIF (SDH_INTSTS [8]) trigger interrupt Enabled.
[7:2]	**Reserved**	Reserved.
[1]	**CRCIEN**	**CRC7, CRC16 and CRC Status Error Interrupt Enable Bit** 0 = CRCIF (SDH_INTSTS [1]) trigger interrupt Disabled. 1 = CRCIF (SDH_INTSTS [1]) trigger interrupt Enabled.
[0]	**BLKDIEN**	**Block Transfer Done Interrupt Enable Bit** 0 = BLKDIF (SDH_INTSTS [0]) trigger interrupt Disabled. 1 = BLKDIF (SDH_INTSTS [0]) trigger interrupt Enabled.

SD1/EMMC1 Interrupt Status Register (SDH_INTSTS)

Register	Offset	R/W	Description	Reset Value
SDH_INTSTS	SDH_BA+0x82C	R/W	SD1/EMMC1 Interrupt Status Register	0x000X_008C

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved					**DAT1STS**	**Reserved**	**CDSTS**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**DITOIF**	**RTOIF**	**Reserved**	**SD1_IF**	**Reserved**	**CDIF**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DAT0STS	**CRCSTS**			**CRC16**	**CRC7**	**CRCIF**	**BLKDIF**

Bits	Description	
[31:19]	**Reserved**	Reserved.
[18]	**DAT1STS**	**DAT1 Pin Status of SD1/EMMC1 Card (Read Only)** This bit indicates the DAT1 pin status of SD card.
[17]	**Reserved**	Reserved.
[16]	**CDSTS**	**Card Detect Status of SD1/EMMC1 (Read Only)** This bit indicates the card detect pin status of SD1/eMMC1, and is used for card detection. When there is a card inserted in or removed from SD1/eMMC1, software should check this bit to confirm if there is really a card insertion or removal. If CDSRC (SDH_INTEN[30]) = 0, to select DAT3 for card detection:. 0 = Card removed. 1 = Card inserted. If CDSRC (SDH_INTEN[30]) = 1, to select GPIO for card detection:. 0 = Card inserted. 1 = Card removed.
[15:14]	**Reserved**	Reserved.
[13]	**DITOIF**	**Data Input Time-out Interrupt Flag (Read Only)** This bit indicates that SD1/eMMC1 counts to time-out value when receiving data (waiting start bit). 0 = Not time-out. 1 = Data input time-out. **Note:** This bit is read only, but can be cleared by writing '1' to it.

[12]	**RTOIF**	**Response Time-out Interrupt Flag (Read Only)** This bit indicates that SD1/eMMC1 counts to time-out value when receiving response or R2 (waiting start bit). 0 = Not time-out. 1 = Response time-out. **Note:** This bit is read only, but can be cleared by writing ‘1’ to it.
[11]	**Reserved**	Reserved.
[10]	**SD1_IF**	**SD1/EMMC1 Interrupt Flag (Read Only)** This bit indicates that SD1/eMMC1 card issues an interrupt to host. This interrupt is designed to level sensitive. Before clear it, turn off SD1/eMMC1 Interrupt Control Register [SD1_IE] first. 0 = No interrupt is issued by SD card. 1 = an interrupt is issued by SD card. **Note:** This bit is read only, but can be cleared by writing ‘1’ to it.
[9]	**Reserved**	Reserved.
[8]	**CDIF**	**SD1/EMMC1 Card Detection Interrupt Flag (Read Only)** This bit indicates that SD card is inserted or removed. Only when CDIEN (SDH_INTEN[8]) is set to 1, this bit is active. 0 = No card is inserted or removed. 1 = There is a card inserted in or removed from SD1/eMMC1. **Note:** This bit is read only, but can be cleared by writing ‘1’ to it.
[7]	**DAT0STS**	**DAT0 Pin Status of Current Selected SD1/EMMC1 Port (Read Only)** This bit is the DAT0 pin status of current selected SD1/eMMC1 port.
[6:4]	**CRCSTS**	**CRC Status Value of Data-out Transfer (Read Only)** SD1/eMMC1 will record CRC status of data-out transfer. Software could use this value to identify what type of error is during data-out transfer. 010 = Positive CRC status. 101 = Negative CRC status. 111 = SD card programming error occurs.
[3]	**CRC16**	**CRC16 Check Status of Data-in Transfer (Read Only)** SD1/eMMC1 will check CRC16 correctness after data-in transfer. 0 = Fault. 1 = OK.
[2]	**CRC7**	**CRC7 Check Status (Read Only)** SD1/eMMC1 will check CRC7 correctness during each response in. If that response does not contain CRC7 information (ex. R3), then software should turn off CRCIEN (SDH_INTEN[1]) and ignore this bit. 0 = Fault. 1 = OK.

[1]	**CRCIF**	**CRC7, CRC16 and CRC Status Error Interrupt Flag (Read Only)** This bit indicates that SD1/eMMC1 has occurred CRC error during response in, data-in or data-out (CRC status error) transfer. When CRC error is occurred, software should reset SD1/eMMC1 engine. Some response (ex. R3) doesn’t have CRC7 information with it; SD1/eMMC1 will still calculate CRC7, get CRC error and set this flag. In this condition, software should ignore CRC error and clears this bit manually. 0 = No CRC error is occurred. 1 = CRC error is occurred. **Note:** This bit is read only, but can be cleared by writing ‘1’ to it.
[0]	**BLKDIF**	**Block Transfer Done Interrupt Flag (Read Only)** This bit indicates that SD1/eMMC1 has finished all data-in or data-out block transfer. If there is a CRC16 error or incorrect CRC status during multiple block data transfer, the transfer will be broken and this bit will also be set. 0 = Not finished yet. 1 = Done. **Note:** This bit is read only, but can be cleared by writing ‘1’ to it.

SD1/EMMC1 Receiving Response Token Register 0 (SDH_RESP0)

Register	Offset	R/W	Description	Reset Value
SDH_RESP0	SDH_BA+0x830	R	SD1/EMMC1 Receiving Response Token Register 0	0x0000_0000

31	30	29	28	27	26	25	24
RESPTK0							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
RESPTK0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RESPTK0							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RESPTK0							

Bits	Description	
[31:0]	**RESPTK0**	**SD1/EMMC1 Receiving Response Token 0 (Read Only)** SD1/eMMC1 controller will receive a response token for getting a reply from SD card when RIEN (SDH_CTL[1]) is set. This field contains response bit 47-16 of the response token.

SD1/EMMC1 Receiving Response Token Register 1 (SDH_RESP1)

Register	Offset	R/W	Description	Reset Value
SDH_RESP1	SDH_BA+0x834	R	SD1/EMMC1 Receiving Response Token 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**
RESPTK1							

Bits	Description	
[7:0]	**RESPTK1**	**SD1/EMMC1 Receiving Response Token 1 (Read Only)** SD1/eMMC1 controller will receive a response token for getting a reply from SD card when RIEN (SDH_CTL[1]) is set. This register contains the bit 15-8 of the response token.

NUC980 SERIES TECHNICAL REFERENCE MANUAL

SD1/EMMC1 Block Length Register (SDH_BLEN)

Register	Offset	R/W	Description	Reset Value
SDH_BLEN	SDH_BA+0x838	R/W	SD1/EMMC1 Block Length Register	0x0000_01FF

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

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10:0]	**BLKLEN**	**SD1/EMMC1 Block Length in Byte Unit** An 11-bit value specifies the SD1/eMMC1 transfer byte count of a block. The actual byte count is equal to BLKLEN+1. **Note :** The default SD block length is 512 bytes

SD1/EMMC1 Response/Data-in Time-out Register (SDH_TOUT)

Register	Offset	R/W	Description	Reset Value
SDH_TOUT	SDH_BA+0x83C	R/W	SD1/EMMC1 Response/Data-in Time-out Register	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**TOUT**	**SD1/EMMC1 Response/Data-in Time-out Value** A 24-bit value specifies the time-out counts of response and data input. SD1/eMMC1 controller will wait start bit of response or data-in until this value reached. The time period depends on SD1/eMMC1 engine clock frequency. Do not write a small number into this field, or you may never get response or data due to time-out. **Note:** Filling 0x0 into this field will disable hardware time-out function.

6.26 Cryptographic Accelerator (CRYPTO)

6.26.1 Overview

The Crypto (Cryptographic Accelerator) includes a secure pseudo random number generator (PRNG) core and supports AES, SHA, HMAC, RSA and ECC algorithms.

The PRNG core supports 64 bits, 128 bits, 192 bits, and 256 bits random number generation.

The AES accelerator is an implementation fully compliant with the AES (Advance Encryption Standard) encryption and decryption algorithm. The AES accelerator supports ECB, CBC, CFB, OFB, CTR, CBC-CS1, CBC-CS2, and CBC-CS3 mode.

The SHA accelerator is an implementation fully compliant with the SHA-160, SHA-224, SHA-256, SHA-384, and SHA-512 and corresponding HMAC algorithms.

The ECC accelerator is an implementation fully compliant with elliptic curve cryptography by using polynomial basis in binary field and prime filed.

The RSA accelerator is an implementation fully compliant with 1024 and 2048 bit RSA cryptography.

6.26.2 Features

- PRNG
  - Supports 64 bits, 128 bits, 192 bits, and 256 bits random number generation
- AES
  - Supports FIPS NIST 197
  - Supports SP800-38A and addendum
  - Supports 128, 192, and 256 bits key
  - Supports both encryption and decryption
  - Supports ECB, CBC, CFB, OFB, CTR, CBC-CS1, CBC-CS2, and CBC-CS3 mode
  - Supports key expander
- SHA
  - Supports FIPS NIST 180, 180-2
  - Supports SHA-160, SHA-224, SHA-256, SHA-384, and SHA-512
- HMAC
  - Supports FIPS NIST 180, 180-2
  - Supports HMAC-SHA-160, HMAC-SHA-224, HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512
- ECC
  - Supports both prime field GF(p) and binary filed GF(2^m)
  - Supports NIST P-192, P-224, P-256, P-384, and P-521
  - Supports NIST B-163, B-233, B-283, B-409, and B-571
  - Supports NIST K-163, K-233, K-283, K-409, and K-571
  - Supports point multiplication, addition and doubling operations in GF(p) and GF(2^m)
  - Supports modulus division, multiplication, addition and subtraction operations in GF(p)

- RSA
  - Supports both encryption and decryption
  - Supports up to 2048 bits

6.26.3 Block Diagram

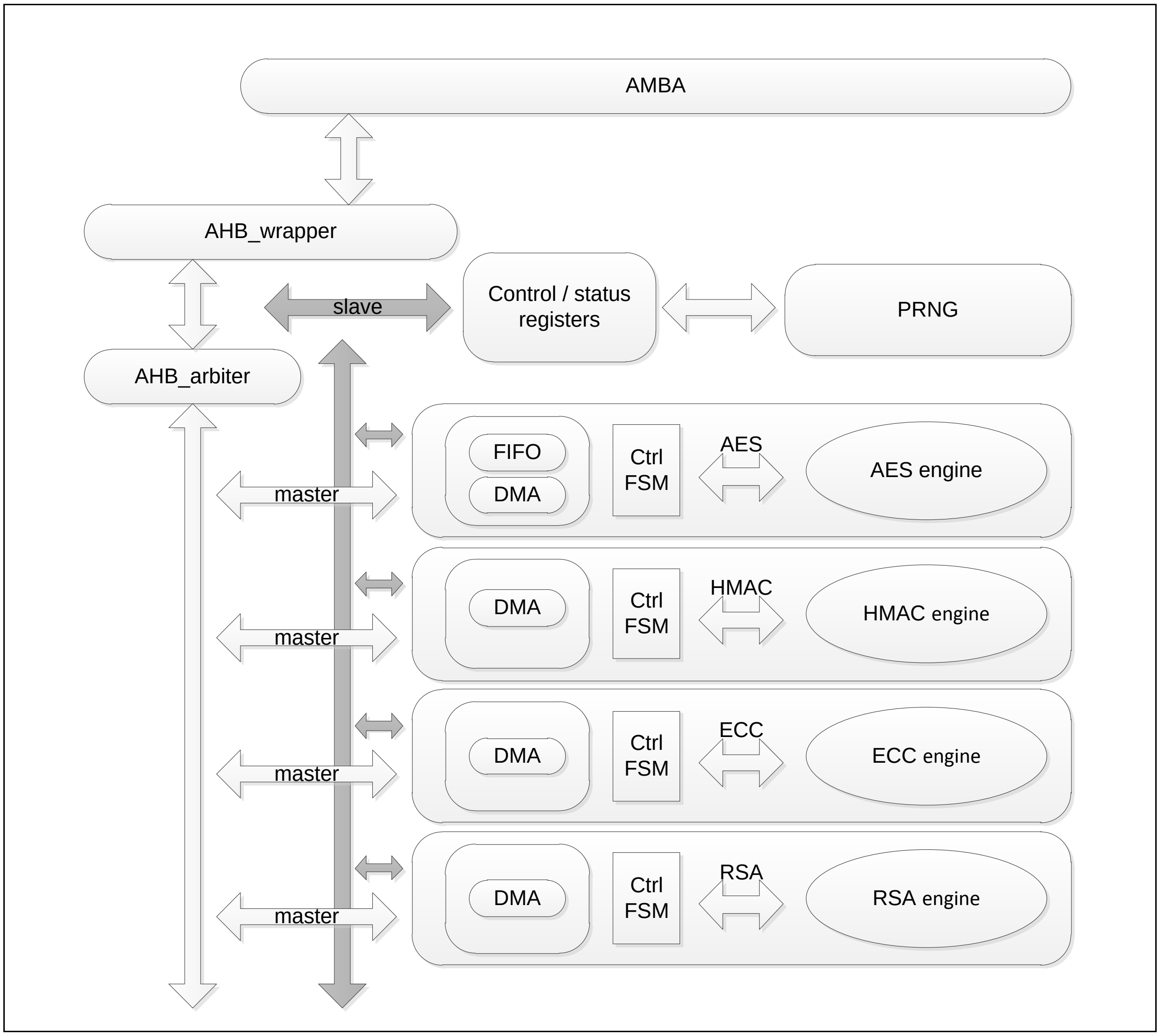


Figure 6.26-1 Cryptographic Accelerator Block Diagram

6.26.4 Basic Configuration

Before using cryptographic engine, it’s necessary to enable clock of cryptographic engine. Set CRYPTO (CLK_HCLKEN[23]) high to enable clock for cryptographic engine operation.

6.26.5 Functional Description

The cryptographic accelerator includes a secure pseudo random number generator (PRNG) core and supports AES, SHA, HMAC, ECC and RSA algorithms. The accelerator can be used in different data security applications, such as secure communications that need cryptographic protection and integrity.

1. The PRNG core supports 64 bits, 128 bits, 192 bits, and 256 bits random number generation

configured by KEYSZ.

2. The AES accelerator is a fully compliant implementation of the AES (Advance Encryption Standard) encryption and decryption algorithm. The AES accelerator supports ECB, CBC, CFB, OFB, CTR, CBC-CS1, CBC-CS2, and CBC-CS3 mode. The AES accelerator provides the DMA function to reduce the CPU intervention, and supports three burst lengths, sixteen-words, eight-words, and four-words.
3. The SHA/HMAC accelerator is a fully compliant implementation of the SHA-160, SHA-224, SHA-256, SHA-384, SHA-512, and corresponding HMAC algorithm. The SHA/HMAC accelerator also supports the DMA function to reduce the CPU intervention. It supports three burst lengths, sixteen-words, eight-words, and four-words.
4. The ECC accelerator is a fully compliant implementation of the prime field GF(p) and binary field GF(2^m) algorithm. The prime field GF(p) supports NIST P-192, P-224, P-256, P-384 and P-521. The binary field GF(2^m) supports NIST B-163, B-233, B-283, B-409, B-571 and NIST K-163, K-233, K-283, K-409 and K-571.
5. The RSA accelerator is a fully compliant implementation of RSA cryptography with 1024-bit and 2048-bit encryption/decryption.

Software can control the data flow by enabling the CRPT_INTEN, and monitor the accelerator status by checking the CRPT_INTSTS.

The cryptographic accelerator supports the following features to enhance the performance.

DMA mode:

Once DMA source address register, destination address register, and byte count register are configurated by CPU, moving data from and to accelerator is done by DMA logic totally. This mode can off-load the loading from the CPU. The cryptographic accelerator embeds one hardware DMA channel for AES, SHA/HMAC, RSA and ECC engine.

DMA Cascade mode:

In the case that the data SRAM resource is tight, or another peripheral is scheduled to switch, the data source or sink needs an update, while the setting for the accelerator operation is planned to be kept. In this mode, software can update DMA source address register, destination address register, and byte count register during a cascade operation, without finishing the accelerator operation.

Non-DMA mode:

In the case that the input data is small in size, DMA mode is not preferred. This mode can reduce the processing time for the accelerator, since no DMA related register needs a configuration, and no latency in DMA logic is introduced. Input data was feeding to cryptographic engine via writing to data input register.

6.26.5.2 PRNG (Pseudo Random number Generator)

Figure 6.26-2 is depicted below. The core supports 64 bits, 128 bits, 192 bits, and 256 bits random number generation configured by KEYSZ(CRPT_PRNG_CTL[3:2]).

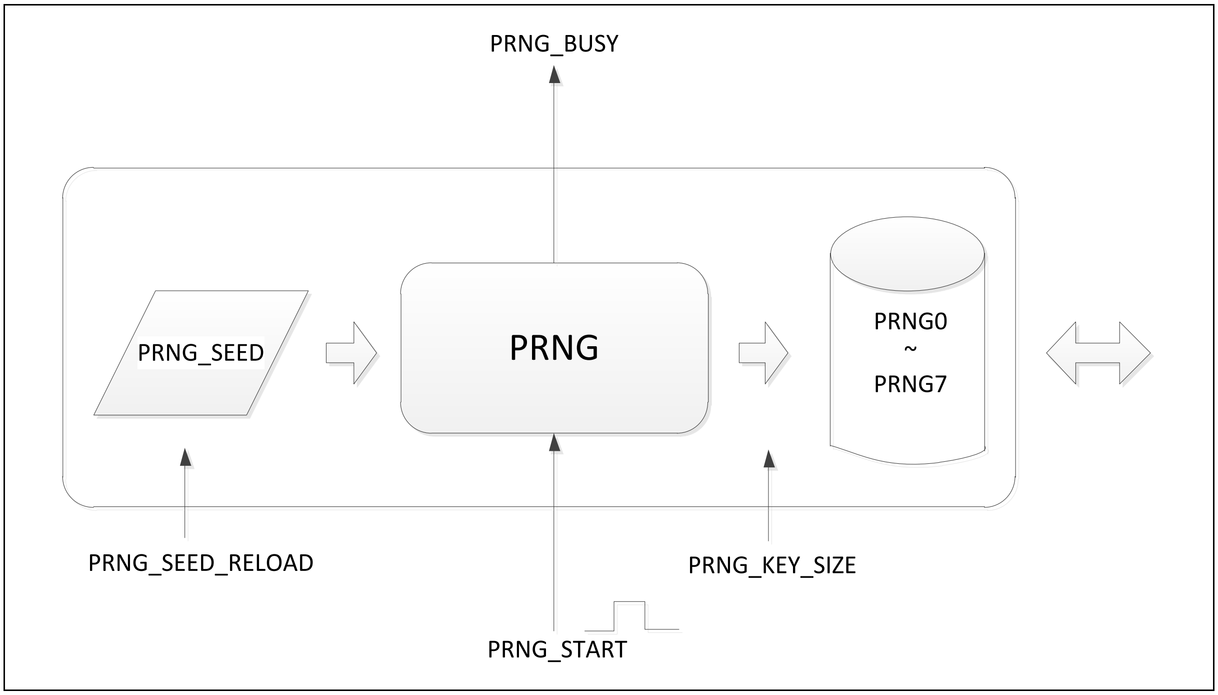


Figure 6.26-2 PRNG Function Diagram

Program steps to get the pseudo random number are depicted below.

1. Check the BUSY(CRPT_PRNG_CTL[8]) until it comes to 0.
2. Initialize PRNG parameters. Configure KEYSZ (CRPT_PRNG_CTL[3:2]), and write a random seed to CRPT_PRNG _SEED. Note that CRPT_PRNG_SEED should be initialized since it's not initialized as the chip powers up.
3. Configure PRNG control register CRPT_PRNG_CTL.
4. Software checks BUSY(CRPT_PRNG_CTL[8]) until it comes to 0, or waits for the PRNG done interrupt (must enable the corresponding interrupt enable register). Then software can read the output random numbers from CRPT_PRNG_KEY0 ~ CRPT_PRNG_KEY7.

6.26.5.3 AES (Advanced Encryption Standard)

Electronic Codebook Mode:

The Electronic Codebook (ECB) mode is a confidentiality mode that features the assignment of a fixed ciphertext block to each plaintext block, for a given key. It's analogous to the assignment of code words in a codebook.

In ECB encryption, each block of the plaintext is applied to the forward cipher function CIPH_k directly and independently. The resulting sequence of output blocks is the ciphertext. In ECB dencryption, each block of the ciphertext is applied to the inverse cipher function CIPH^{-1}_k directly and independently. The resulting sequence of output blocks is the plaintext.

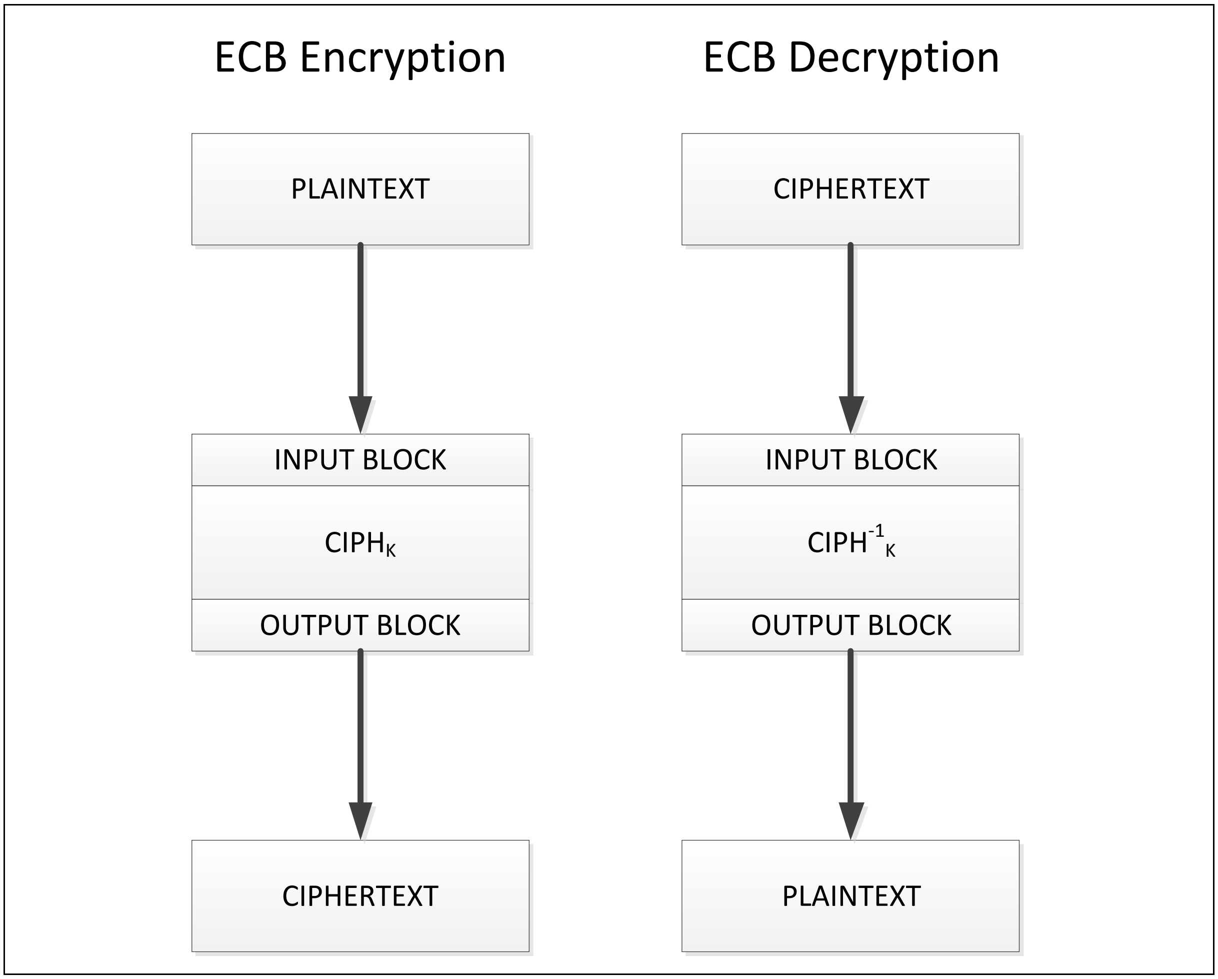


Figure 6.26-3 Electronic Codebook Mode

In ECB mode, any given plaintext block always gets encrypted to the same ciphertext block under a given key. If this property is undesirable in a particular application, the ECB mode should not be used.

Cipher Block Chaining Mode:

The Cipher Block Chaining (CBC) mode is a confidentiality mode whose encryption process features the combining chaining of the plaintext blocks with the previous ciphertext blocks. The CBC mode requires an initialization vector (IV) to combine with the first plaintext block. The IV does not need to be secret, but it must be unpredictable.

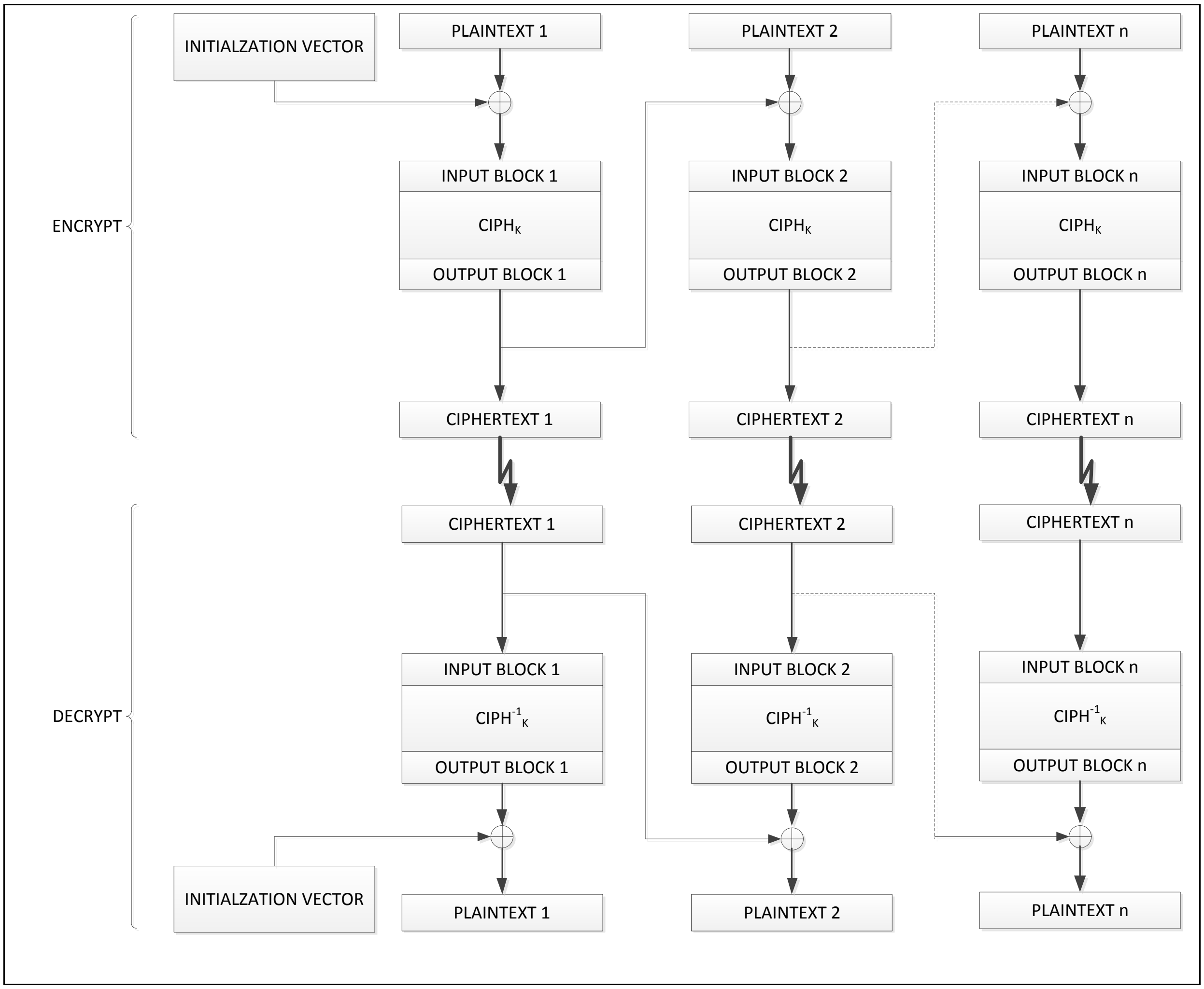


Figure 6.26-4 Cipher Block Chaining Mode

<u>Cipher Feedback Mode (CFB):</u>

The Cipher Feedback (CFB) mode is a confidentiality mode that features the feedback of successive ciphertext segments into the input blocks of the forward cipher to generate output blocks that are exclusive-ORed with the plaintext to produce the ciphertext, and vice versa. The CFB mode requires an IV as the initial input block. The IV need not be secret, but it must be unpredictable

.

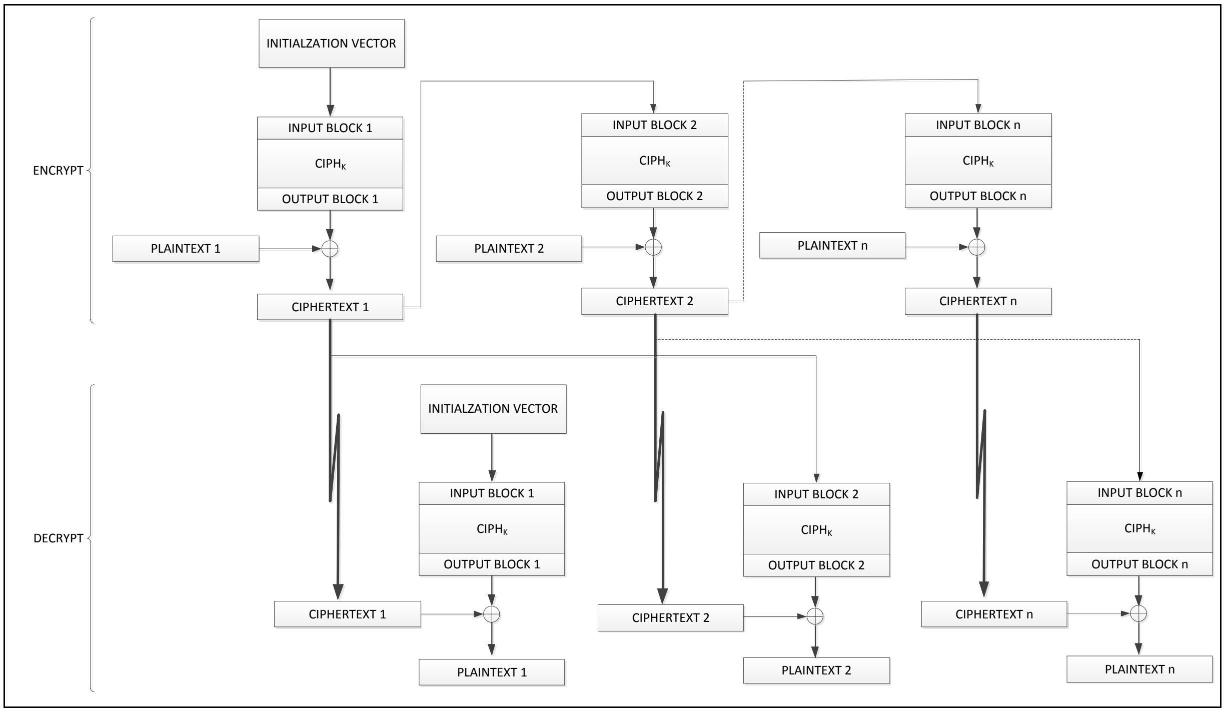


Figure 6.26-5 Cipher Feedback Mode

Output Feedback Mode:

The Output Feedback (OFB) mode is a confidentiality mode that features the iteration of the forward cipher on an IV to generate a sequence of output blocks that are exclusive-ORed with the plaintext to produce the ciphertext, and vice versa. The OFB mode requires that the IV is a nonce, i.e., the IV must be unique for each execution of the mode under the given key.

The OFB mode requires a unique IV for every message that is ever encrypted under the given key. If, contrary to this requirement, the same IV is used for the encryption of more than one message, then the confidentiality of those messages may be compromised. Confidentiality may be similarly be compromised if any of the input blocks to the forward cipher function for the encryption of a message is designated as the IV for the encryption of another message under the given key.

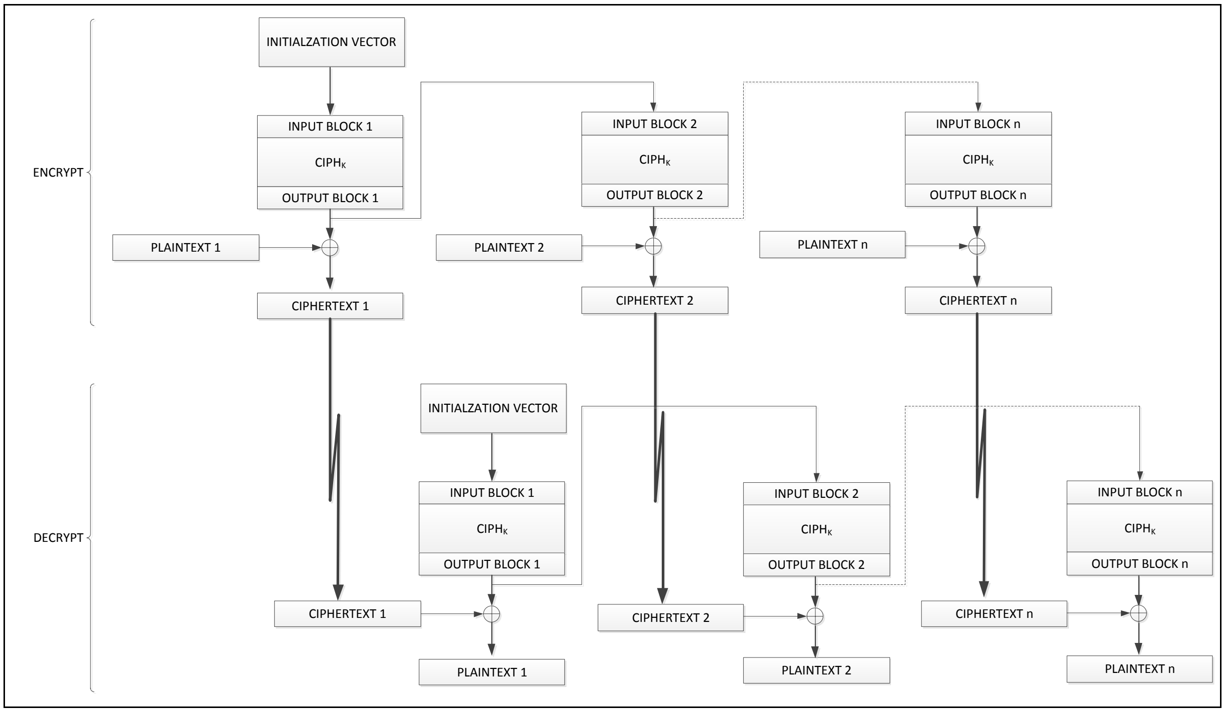


Figure 6.26-6 Output Feedback Mode

Counter Mode (CTR):

The Counter (CTR) mode is a confidentiality mode that features the application of the forward cipher to a set of input blocks, called counters, to produce a sequence of output blocks that are exclusive-ORed with the plaintext to produce the ciphertext, and vice versa. The sequence of counters must have the property that each block in the sequence is different from every other block. This condition is not restricted to a single message: across all of the messages that are encrypted under the given key, all of the counters must be distinct.

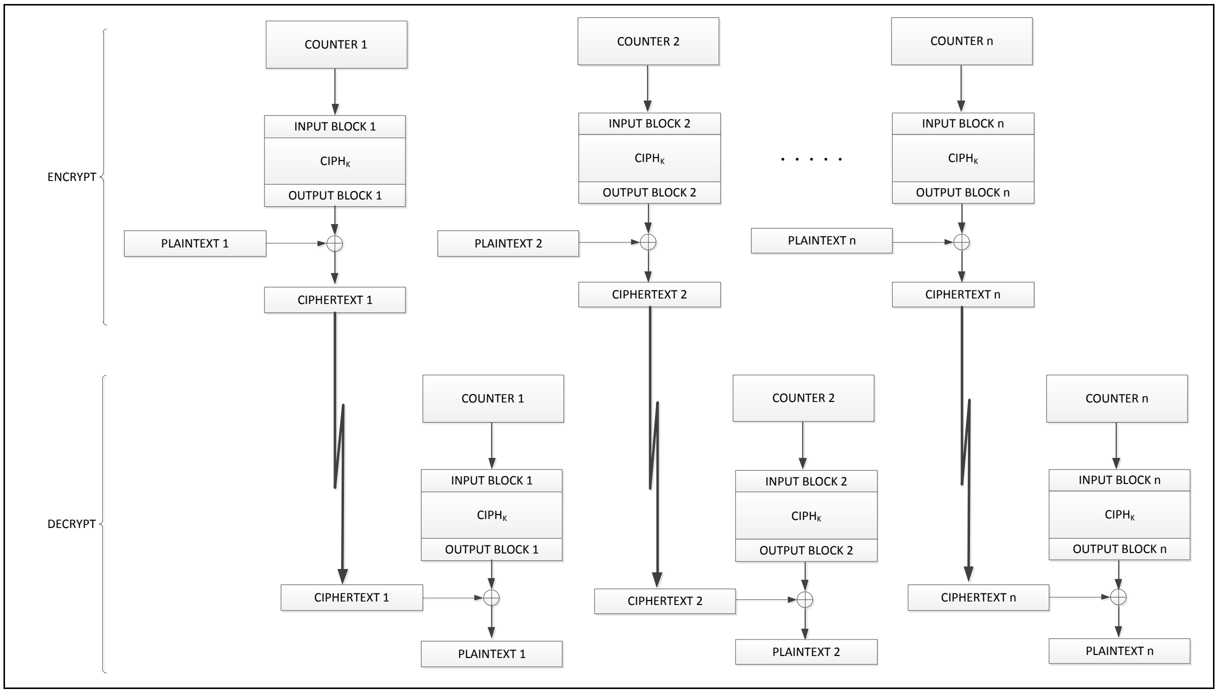


Figure 6.26-7 Counter Mode

CBC Ciphertext-Stealing 1 Mode (CBC-CS1):

Figure 6.26-8 illustrates the CBC-CS1-Encrypt algorithm for the case that P_n^* is a partial block. The cryptographic accelerator would append P_n^* with ‘0’ to form a complete block P_n.

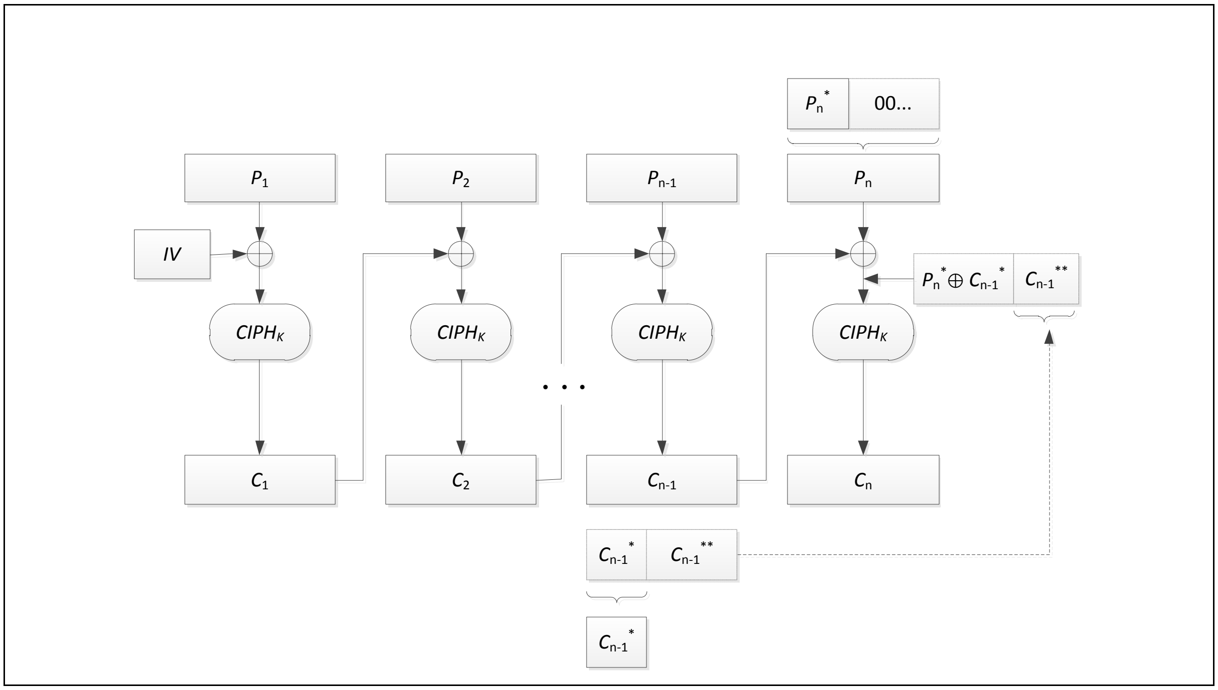


Figure 6.26-8 CBC-CS1 Encryption

Figure 6.26-9 illustrates the CBC-CS1-Decrypt algorithm for the case that C_{n-1}^* is a partial block.

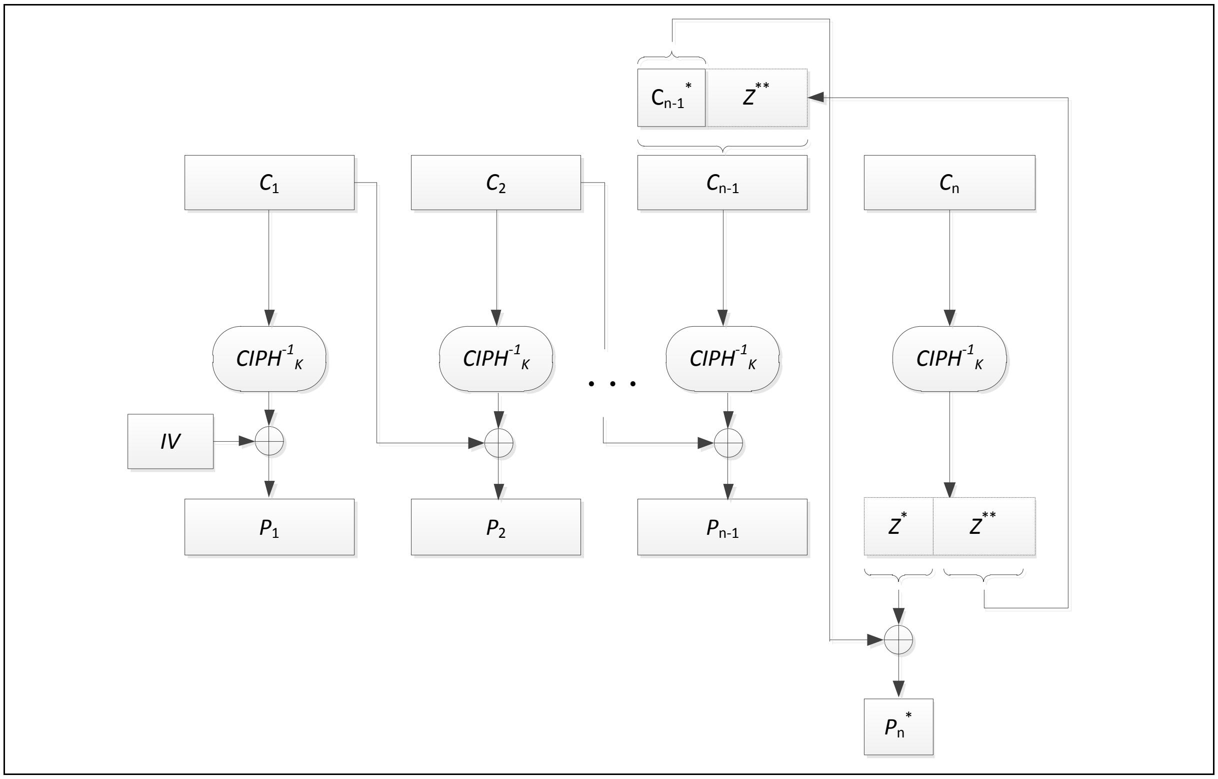


Figure 6.26-9 CBC-CS1 Decryption

CBC Ciphertext-Stealing 2 Mode (CBC-CS2):

When P_n^* is a partial block, then CBC-CS2-Encrypt and CBC-CS1-Encrypt differ only in the ordering of C_{n-1}^* and C_n..

CBC Ciphertext-Stealing 3 Mode (CBC-CS3):

C_{n-1}^* and C_n are unconditionally swapped, i.e., even when C_{n-1}^* is a complete block; therefore, CBC-CS3 is not strictly an extension of CBC mode. In the other case, i.e., when C_{n-1}^* is a nonempty partial block, CBC-CS3-Encrypt is equivalent to CBC-CS2-Encrypt.

Refer to the following programming steps for how to program the AES related registers.

AES DMA mode programming flow:

1. Write 1 to AESIEN (CRPT_INTEN[0]) to enable AES interrupt.
2. Program AES key to registers CRPT_AESn_KEY0 ~ CRPT_AESn_KEY7. (where n is 0 in NUC980)
3. Program initial vectors to registers CRPT_AESn_IV0 ~ CRPT_AESn_IV3.
4. Program DMA source address to register CRPT_AESn_SADDR.
5. Program DMA destination address to register CRPT_AESn_DADDR.
6. Program DMA byte count to register CRPT_AESn_CNT.
7. Configure AES control register CRPT_AES_CTL for encryption/decryption, operational mode, DMA mode, key size, and DMA input/output swap.
8. Write input data to DMA source address with selected DMA byte count.
9. Write 1 to START(CRPT_AES_CTL[0]) to start AES encryption/decryption.

10. Waits for the AES interrupt flag AESIF (CRPT_INTSTS[0]) be set.
11. Read output data from DMA destination address with selected DMA byte count.
12. Repeat step 9 to step 12 until all data processed.

AES Non-DMA mode programming flow:

1. Write 1 to AESIEN (CRPT_INTEN[0]) to enable AES interrupt.
2. Program AES key to register CRPT_AESn_KEY0 ~ CRPT_AESn_KEY7. (where n is 0 in NUC980)
3. Program initial vectors to register CRPT_AESn_IV0 ~ CRPT_AESn_IV3.
4. Configure AES control register (CRPT_AES_CTL) for encryption/decryption, operational mode, and key size.
5. Write 1 to START(CRPT_AES_CTL[0]) to start AES encryption/decryption.
6. Polling INBUFFULL(CRPT_AES_STS[9]) and OUTBUFEMPTY(CRPT_AES_STS[16]). If INBUFFULL(CRPT_AES_STS[9]) is 0, write 32 bits input data to CRPT_AES_DATIN. If OUTBUFEMPTY(CRPT_AES_STS[16]) is 0, read 32 bits data from CRPT_AES_DATOUT.
7. Repeat step 6 until 128 bits data (16 bytes) are written to and read from AES engine.
8. Write 1 to DMALAST(CRPT_AES_CTL[5]).
9. Repeat steps 6 to step 8 until all data processed.

6.26.5.4 SHA (Secure Hash Algorithm)

The Secure Hash Algorithm is a family of cryptographic hash functions published by the National Institute of Standards and Technology (NIST) as a U.S. Federal Information Processing Standard (FIPS).

User can refer to the following steps to understand how to program the SHA related registers.

SHA DMA mode programming flow:

1. Write 1 to HMACIEN(CRPT_INTEN[24]) to enable SHA/HMAC interrupt.
2. Configure SHA/HMAC control register CRPT_HMAC_CTL for SHA/HMAC engine input/output data swap, DMA mode, and SHA operation mode. Clear HMACEN(CRPT_HMAC_CTL[4]) to select SHA mode.
3. Program DMA source address to register CRPT_HMAC_SADDR.
4. Program DMA byte count to register CRPT_HMAC_DMACNT.
5. Write input data to DMA source address with selected DMA byte count.
6. Write 1 to START(CRPT_HMAC_CTL[0]) to start SHA encryption.
7. Waits for the SHA interrupt flag HMACIF(CRPT_INTSTS[24]) be set.
8. Read output digest (SHA160: CRPT_HMAC_DGST0 ~ CRPT_HMAC_DGST4, SHA224: CRPT_HMAC_DGST0 ~ CRPT_HMAC_DGST6, SHA256: CRPT_HMAC_DGST0 ~ CRPT_HMAC_DGST7, SHA384: CRPT_HMAC_DGST0 ~ CRPT_HMAC_DGST11, SHA512: CRPT_HMAC_DGST0 ~ CRPT_HMAC_DGST15).

SHA Non-DMA mode programming flow:

1. Configure SHA/HMAC control register CRPT_HMAC_CTL for SHA/HMAC engine input/output data swap and SHA operation mode. Clear HMACEN(CRPT_HMAC_CTL[4]) to select SHA mode.

2. If it's the last input word, set DMALAST(CRPT_HMAC_CTL[5]).
3. Write 1 to START(CRPT_HMAC_CTL[0]) to start SHA encryption.
4. Waits for the SHA data input request DATINREQ(CRPT_HMAC_STS[16]) be set.
5. Write one word of input data to CRPT_HMAC_DATIN.
6. Repeat step 2 to 5 until all inut words are written into SHA engine.
7. Waits for the BUSY (CRPT_HMAC_STS[0]) be cleared.
8. Read output digest (SHA160: CRPT_HMAC_DGST0 ~ CRPT_HMAC_DGST4, SHA224: CRPT_HMAC_DGST0 ~ CRPT_HMAC_DGST6, SHA256: CRPT_HMAC_DGST0 ~ CRPT_HMAC_DGST7, SHA384: CRPT_HMAC_DGST0 ~ CRPT_HMAC_DGST11, SHA512: CRPT_HMAC_DGST0 ~ CRPT_HMAC_DGST15).

6.26.5.5 HMAC (Keyed-Hash Message Authentication Code)

The Keyed-Hash Message Authentication Code is a specific construction for calculating a message authentication code involving a cryptographic hash function in combination with a secret cryptographic key. Any cryptographic hash function, such as SHA-1, may be used in the calculation of an HMAC; the resulting MAC algorithm is termed HMAC-SHA1 accordingly.

User can refer to the following steps to under

There are no sources in the current document.stand how to program the HMAC related registers.

HMAC DMA mode programming flow:

1. Write 1 to HMACIEN(CRPT_INTEN[24]) to enable HMAC interrupt.
2. Configure SHA/HMAC control register CRPT_HMAC_CTL for HMAC engine input/output data swap, DMA mode, and HMAC operation mode. Set HMACEN(CRPT_HMAC_CTL[4]) to select HMAC mode.
3. Program DMA source address to register CRPT_HMAC_SADDR.
4. Program DMA byte count to register CRPT_HMAC_DMACNT.
5. Write input data to DMA source address with selected DMA byte count.
6. Write 1 to START(CRPT_HMAC_CTL[0]) to start HMAC encryption.
7. Waits for the HMAC interrupt flag HMACIF(CRPT_INTSTS[24]) be set.
8. Read output digest (HMAC-SHA160: CRPT_HMAC_DGST0 ~ CRPT_HMAC_DGST4, HMAC-SHA224: CRPT_HMAC_DGST0 ~ CRPT_HMAC_DGST6, HMAC-SHA256: CRPT_HMAC_DGST0 ~ CRPT_HMAC_DGST7, HMAC-SHA384: CRPT_HMAC_DGST0 ~ CRPT_HMAC_DGST11, HMAC-SHA512: CRPT_HMAC_DGST0 ~ CRPT_HMAC_DGST15).

HMAC Non-DMA mode programming flow:

1. Configure SHA/HMAC control register CRPT_HMAC_CTL for SHA/HMAC engine input/output data swap and HMAC operation mode. Set HMACEN(CRPT_HMAC_CTL[4]) to select HMAC mode.
2. If it's the last input word, set DMALAST(CRPT_HMAC_CTL[5]).
3. Write 1 to START(CRPT_HMAC_CTL[0]) to start HMAC encryption.
4. Waits for the HMAC data input request DATINREQ(CRPT_HMAC_STS[16]) be set.
5. Write one word of input data to CRPT_HMAC_DATIN.
6. Repeat step 2 to 5 until all inut words are written into SHA engine.

7. Waits for the BUSY (CRPT_HMAC_STS[0]) be cleared.
8. Read output digest (HMAC-SHA160: CRPT_HMAC_DGST0 ~ CRPT_HMAC_DGST4, HMAC-SHA224: CRPT_HMAC_DGST0 ~ CRPT_HMAC_DGST6, HMAC-SHA256: CRPT_HMAC_DGST0 ~ CRPT_HMAC_DGST7, HMAC-SHA384: CRPT_HMAC_DGST0 ~ CRPT_HMAC_DGST11, HMAC-SHA512: CRPT_HMAC_DGST0 ~ CRPT_HMAC_DGST15).

6.26.5.6 ECC (Elliptic Curve Cryptography)

Elliptic Curve Cryptography (ECC) is a famous approach of public-key cryptosystems. Recently, many protocols and applications utilize the algebraic cyclic group characters of elliptic curves over finite field to build cryptographic systems. All points of an elliptic curve will follow the formula of elliptic curve : y2 ≡x3+A*x+B (mod N) in GF(p) and y2+x*y≡x3+A*x2+B (mod N) in GF(2^m). Figure 6.26-10 exhibits the main hierarchy chart of ECC applications. The often appeared parameters and corresponding registers are shown in Table 6.26-1.

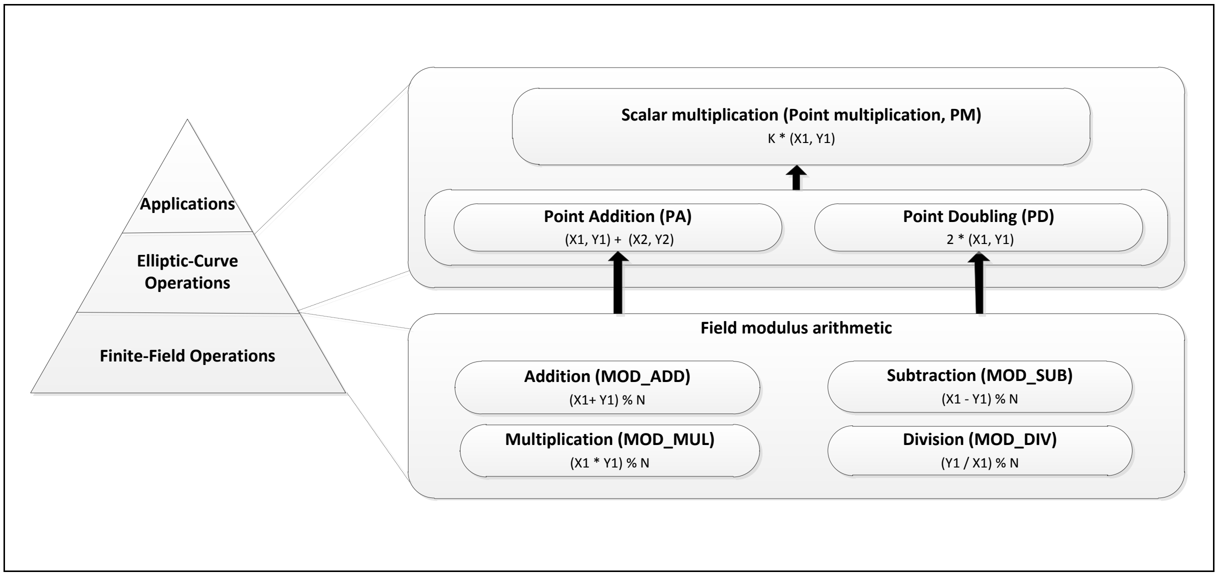


Figure 6.26-10 Main Hierarchy Chart of ECC

Parameter	Description	Corresponding Register
X1	The x-coordinate of point1	CRPT_ECC_X1_00~ CRPT_ECC_X1_17
Y1	The y-coordinate of point1	CRPT_ECC_Y1_00~ CRPT_ECC_Y1_17
X2	The x-coordinate of point2	CRPT_ECC_X2_00~ CRPT_ECC_X2_17
Y2	The y-coordinate of point2	CRPT_ECC_Y2_00~ CRPT_ECC_Y2_17
A	The curve parameter A	CRPT_ECC_A_00~ CRPT_ECC_A_17
B	The curve parameter B	CRPT_ECC_B_00~ CRPT_ECC_B_17
N	The curve parameter N	CRPT_ECC_N_00~ CRPT_ECC_N_17
M	The curve length	CRPT_ECC_CTL[31:22]
K	The scalar constant	CRPT_ECC_K_00~ CRPT_ECC_K_17

Table 6.26-1 ECC Parameters and Corresponding Registers Table

Scalar multiplication (point multiplication) is the core operation in ECC applications. The computation of scalar multiplication is composed of point addition and point doubling operations. Moreover, there are many finite field modulus arithmetic operations in the formula of point addition and doubling operation. To accelerate ECC applications, we propose an elliptic curve cryptographic accelerator that can process not only three point operations in both GF(p) and GF(2^m) but also four modulus operations in GF(p). Before starting ECC accelerator, user must provide the required input data of ECC operation include point coordinates (X1, Y1, X2, Y2), curve parameters (A, B, N, M) and scalar data (K) in Table 6.26-2.The mark "√" means that the input data is necessary for this operation. The detail definition of input data and the corresponding registers in the ECC accelerator are exhibited in the next section Register Map.

After ECC accelerator finished, all point operations will generate a output point includes x-coordinate in the registers from CRPT_ECC_X1_00 to CRPT_ECC_X1_17 and y-coordinate in the registers from CRPT_ECC_Y1_00 to CRPT_ECC_Y1_17. In all modulus operations, ECC accelerator will only produce a output result in the registers from CRPT_ECC_X1_00 to CRPT_ECC_X1_17.

Operation	PM	PA	PD	MOD_DIV	MOD_MUL	MOD_ADD	MOD_SUB
ECCOP[1:0]	00	10	11	01	01	01	01
MODOP[1:0]	XX	XX	XX	00	01	10	11
X1	√	√	√	√	√	√	√
Y1	√	√	√	√	√	√	√
X2		√					
Y2		√					
A	√	√	√				
B	√		√				
N	√	√	√	√	√	√	√
M	√	√	√		√		
K	√						

Table 6.26-2 Required Input Data of Various Operations

User can refer to the following steps to understand how to program the ECC related registers.

ECC DMA Mode Programming Flow

1. Write 1 to ECCIEN(CRPT_INTEN[22]) to enable ECC interrupt if needed.
2. Program DMA source address to register CRPT_ECC_SADDR.
3. Program DMA destination address to register CRPT_ECC_DADDR.
4. Program DMA word count to register CRPT_ECC_WORDCNT.
5. Program the starting register address of all input data in Table 6.26-2 that will update to the register CRPT_ECC_STARTREG.
6. Write input data to DMA source address with selected DMA word count.
7. Configure ECC control register CRPT_ECC_CTL for ECC accelerator, such as the start signal of ECC accelerator(START), DMA mode enable signal(DMAEN), field selection(FSEL), point operation mode(ECCOP), modulus operation mode(MODOP), the control signals fo all input data registers(LDA, LDB, LDN, LDK, LDP1, LDP2), and the key length of elliptic

curve(CURVEM).

8. Wait for the ECC interrupt flag ECCIF(CRPT_INTSTS[22]) be set.
9. Read output data and then clear ECC interrupt flag ECCIF.

ECC Non-DMA Mode Programming Flow

1. Write all necessary input data in the corresponding registers according to in Table 6.26-2, such as CURVEA, CURVEB, CURVEN, SCALARK.
2. Configure ECC control register CRPT_ECC_CTL for ECC accelerator, such as the start signal of ECC accelerator(START), field selection(FSEL), point operation mode(ECCOP), modulus operation mode(MODOP), the control signals fo all input data registers(LDA, LDB, LDN, LDK, LDP1, LDP2), and the key length of elliptic curve(CURVEM).
3. Wait for the BUSY (CRPT_ECC_STS[0]) be cleared.
4. Read output digest and then clear ECC interrupt flag ECCIF.

Some Notices of ECC Accelerator

1. The key length support of ECC accelerator is from 163 to 256 bits.
2. All input and output data must be positive. (If the input data is negative, it must be added N).
3. The irreducible polynomial of GF(2m) must adopt the smallest one from HP, please refer to Table 6.26-3 (Reference from HP, Table of Low-Weight Binary Irreducible Polynomials, HPL-98-135, August, 1998)

	2,1	**3,1**	**4,1**	**5,2**
6,1	7,1	8,4,3,1	9,1	10,3
11,2	12,3	13,4,3,1	14,5	15,1
16,5,3,1	17,3	18,3	19,5,2,1	20,3
21,2	22,1	23,5	24,4,3,1	25,3
26,4,3,1	27,5,2,1	28,1	29,2	30,1
31,3	32,7,3,2	33,10	34,7	35,2
36,9	37,6,4,1	38,6,5,1	39,4	40,5,4,3
41,3	42,7	43,6,4,3	44,5	45,4,3,1
46,1	47,5	48,5,3,2	49,9	50,4,3,2
51,6,3,1	52,3	53,6,2,1	54,9	55,7
56,7,4,2	57,4	58,19	59,7,4,2	60,1
61,5,2,1	62,29	63,1	64,4,3,1	65,18
66,3	67,5,2,1	68,9	69,6,5,2	70,5,3,1
71,6	72,10,9,3	73,25	74,35	75,6,3,1
76,21	77,6,5,2	78,6,5,3	79,9	80,9,4,2
81,4	82,8,3,1	83,7,4,2	84,5	85,8,2,1
86,21	87,13	88,7,6,2	89,38	90,27
91,8,5,1	92,21	93,2	94,21	95,11

96,10,9,6	97,6	98,11	99,6,3,1	100,15
101,7,6,1	102,29	103,9	104,4,3,1	105,4
106,15	107,9,7,4	108,17	109,5,4,2	110,33
111,10	112,5,4,3	113,9	114,5,3,2	115,8,7,5
116,4,2,1	117,5,2,1	118,33	119,8	120,4,3,1
121,18	122,6,2,1	123,2	124,19	125,7,6,5
126,21	127,1	128,7,2,1	129,5	130,3
131,8,3,2	132,17	133,9,8,2	134,57	135,11
136,5,3,2	137,21	138,8,7,1	139,8,5,3	140,15
141,10,4,1	142,21	143,5,3,2	144,7,4,2	145,52
146,71	147,14	148,27	149,10,9,7	150,53
151,3	152,6,3,2	153,1	154,15	155,62
156,9	157,6,5,2	158,8,6,5	159,31	160,5,3,2
161,18	162,27	163,7,6,3	164,10,8,7	165,9,8,3
166,37	167,6	168,15,3,2	169,34	170,11
171,6,5,2	172,1	173,8,5,2	174,13	175,6
176,11,3,2	177,8	178,31	179,4,2,1	180,3
181,7,6,1	182,81	183,56	184,9,8,7	185,24
186,11	187,7,6,5	188,6,5,2	189,6,5,2	190,8,7,6
191,9	192,7,2,1	193,15	194,87	195,8,3,2
196,3	197,9,4,2	198,9	199,34	200,5,3,2
201,14	202,55	203,8,7,1	204,27	205,9,5,2
206,10,9,5	207,43	208,9,3,1	209,6	210,7
211,11,10,8	212,105	213,6,5,2	214,73	215,23
216,7,3,1	217,45	218,11	219,8,4,1	220,7
221,8,6,2	222,5,4,2	223,33	224,9,8,3	225,32
226,10,7,3	227,10,9,4	228,113	229,10,4,1	230,8,7,6
231,26	232,9,4,2	233,74	234,31	235,9,6,1
236,5	237,7,4,1	238,73	239,36	240,8,5,3
241,70	242,95	243,8,5,1	244,111	245,6,4,1
246,11,2,1	247,82	248,15,14,10	249,35	250,103
251,7,4,2	252,15	253,46	254,7,2,1	255,52
256,10,5,2	257,12	258,71	259,10,6,2	260,15
261,7,6,4	262,9,8,4	263,93	264,9,6,2	265,42

266,47	267,8,6,3	268,25	269,7,6,1	270,53
271,58	272,9,3,2	273,23	274,67	275,11,10,9
276,63	277,12,6,3	278,5	279,5	280,9,5,2
281,93	282,35	283,12,7,5	284,53	285,10,7,5
286,69	287,71	288,11,10,1	289,21	290,5,3,2
291,12,11,5	292,37	293,11,6,1	294,33	295,48
296,7,3,2	297,5	298,11,8,4	299,11,6,4	300,5
301,9,5,2	302,41	303,1	304,11,2,1	305,102
306,7,3,1	307,8,4,2	308,15	309,10,6,4	310,93
311,7,5,3	312,9,7,4	313,79	314,15	315,10,9,1
316,63	317,7,4,2	318,45	319,36	320,4,3,1
321,31	322,67	323,10,3,1	324,51	325,10,5,2
326,10,3,1	327,34	328,8,3,1	329,50	330,99
331,10,6,2	332,89	333,2	334,5,2,1	335,10,7,2
336,7,4,1	337,55	338,4,3,1	339,16,10,7	340,45
341,10,8,6	342,125	343,75	344,7,2,1	345,22
346,63	347,11,10,3	348,103	349,6,5,2	350,53
351,34	352,13,11,6	353,69	354,99	355,6,5,1
356,10,9,7	357,11,10,2	358,57	359,68	360,5,3,2
361,7,4,1	362,63	363,8,5,3	364,9	365,9,6,5
366,29	367,21	368,7,3,2	369,91	370,139
371,8,3,2	372,111	373,8,7,2	374,8,6,5	375,16
376,8,7,5	377,41	378,43	379,10,8,5	380,47
381,5,2,1	382,81	383,90	384,12,3,2	385,6
386,83	387,8,7,1	388,159	389,10,9,5	390,9
391,28	392,13,10,6	393,7	394,135	395,11,6,5
396,25	397,12,7,6	398,7,6,2	399,26	400,5,3,2
401,152	402,171	403,9,8,5	404,65	405,13,8,2
406,141	407,71	408,5,3,2	409,87	410,10,4,3
411,12,10,3	412,147	413,10,7,6	414,13	415,102
416,9,5,2	417,107	418,199	419,15,5,4	420,7
421,5,4,2	422,149	423,25	424,9,7,2	425,12
426,63	427,11,6,5	428,105	429,10,8,7	430,14,6,1
431,120	432,13,4,3	433,33	434,12,11,5	435,12,9,5

436,165	437,6,2,1	438,65	439,49	440,4,3,1
441,7	442,7,5,2	443,10,6,1	444,81	445,7,6,4
446,105	447,73	448,11,6,4	449,134	450,47
451,16,10,1	452,6,5,4	453,15,6,4	454,8,6,1	455,38
456,18,9,6	457,16	458,203	459,13,5,2	460,19
461,7,6,1	462,73	463,93	464,19,18,13	465,31
466,14,11,6	467,11,6,1	468,27	469,9,5,2	470,9
471,1	472,11,3,2	473,200	474,191	475,9,8,4
476,9	477,16,15,7	478,121	479,104	480,15,9,6
481,138	482,9,6,5	483,9,6,4	484,105	485,17,16,6
486,81	487,94	488,4,3,1	489,83	490,219
491,11,6,3	492,7	493,10,5,3	494,17	495,76
496,16,5,2	497,78	498,155	499,11,6,5	500,27
501,5,4,2	502,8,5,4	503,3	504,15,14,6	505,156
506,23	507,13,6,3	508,9	509,8,7,3	510,69
511,10	512,8,5,2	513,26	514,67	515,14,7,4
516,21	517,12,10,2	518,33	519,79	520,15,11,2
521,32	522,39	523,13,6,2	524,167	525,6,4,1
526,97	527,47	528,11,6,2	529,42	530,10,7,3
531,10,5,4	532,1	533,4,3,2	534,161	535,8,6,2
536,7,5,3	537,94	538,195	539,10,5,4	540,9
541,13,10,4	542,8,6,1	543,16	544,8,3,1	545,122
546,8,2,1	547,13,7,4	548,10,5,3	549,16,4,3	550,193
551,135	552,19,16,9	553,39	554,10,8,7	555,10,9,4
556,153	557,7,6,5	558,73	559,34	560,11,9,6
561,71	562,11,4,2	563,14,7,3	564,163	565,11,6,1
566,153	567,28	568,15,7,6	569,77	570,67
571,10,5,2	572,12,8,1	573,10,6,4	574,13	575,146
576,13,4,3	577,25	578,23,22,16	579,12,9,7	580,237

Table 6.26-3 Low-Weight Binary Irreducible Polynomials

4. Only when START and DMAEN (CRPT_ECC_CTL[0] and [7]) are assigned to 1 simultaneously, ECC DMA mode will be active.
5. When ECC engine is active (i.e., BUSY is 1 and DMABUSY (CRPT_ECC_STS[1]) is 0), user can’t modify all input data registers (CRPT_ECC_X1_00 ~ CRPT_ECC_K_17).
6. If user wants to stop ECC accelerator, please configures the STOP (CRPT_ECC_CTL[1]) to 1.

Note that to avoid the transmission error of the next operation, BUSY signal will not be cleared immediately until the action of DMA is done.

7. The modulus operation is not supported for binary field.
8. The input data of modulus multiplication operation and division operation for PF must be less than N.
9. K is private key, so this register is write only.

The following describes the method of application about key pair generation, ECDSA and ECDH.

Key Pair Generation

Public key generation function: Q = dG (mod N)

1. Write the curve parameter A, B, N, and curve length M to corresponding registers according to Table 6.26-1.
2. Write the point G(x, y) to X1, Y1 registers according to Table 6.26-1.
3. Write the private key d to K register according to Table 6.26-1.
4. Set ECCOP(CRPT_ECC_CTL[10:9]) to 00
5. Set FSEL(CRPT_ECC_CTL[8]) according to used curve of prime field or binary field
6. Set START(CRPT_ECC_CTL[0]) to 1
7. Wait for BUSY(CRPT_ECC_STS[0]) be cleared
8. Read public key Q from X1, Y1 registers

Elliptic Curve Digital Signature Algorithm (ECDSA)

ECDSA signature generation steps:

1. Calculate e = HASH(m), where HASH is a cryptographic hashing algorithm, (i.e. SHA-1)
   1) Use SHA to calculate e
2. Select a random integer k form [1, n-1]
   1) Note that n is order, not prime modulus or irreducible polynomial function
3. Compute r = x1 (mod n), where (x1, y1) = k * G. If r = 0, go to step 2
   1) Write the curve parameter A, B, N and curve length M to corresponding registers according to Table 6.26-1
   2) Write the prime modulus or irreducible polynomial function to N registers according to Table 6.26-1
   3) Write the point G(x, y) to X1, Y1 registers according to Table 6.26-1
   4) Write the random integer k to K register according to Table 6.26-1
   5) Set ECCOP(CRPT_ECC_CTL[10:9]) to 00
   6) Set FSEL(CRPT_ECC_CTL[8]) according to used curve of prime field or binary field
   7) Set START(CRPT_ECC_CTL[0]) to 1
   8) Wait for BUSY(CRPT_ECC_STS[0]) be cleared
   9) Write the curve order and curve length to N, M registers according to Table 6.26-1

10) Write 0x0 to Y1 registers
11) Set ECCOP(CRPT_ECC_CTL[10:9]) to 01
12) Set MOPOP(CRPT_ECC_CTL[12:11]) to 10
13) Set START(CRPT_ECC_CTL[0]) to 1
14) Wait for BUSY(CRPT_ECC_STS[0]) be cleared
15) Read X1 registers to get r

4. Compute s = k − 1 × (e + d × r)(mod n). If s = 0, go to step 2

1) Write the curve order to N registers according to Table 6.26-1
2) Write the random integer k to X1 registers according to Table 6.26-1
3) Set ECCOP(CRPT_ECC_CTL[10:9]) to 01
4) Set MOPOP(CRPT_ECC_CTL[12:11]) to 00
5) Set START(CRPT_ECC_CTL[0]) to 1
6) Wait for BUSY(CRPT_ECC_STS[0]) be cleared
7) Read X1 registers to get k-1
8) Write the curve order and curve length to N, M registers according to Table 6.26-1
9) Write r, d to X1, Y1 registers
10) Set ECCOP(CRPT_ECC_CTL[10:9]) to 01
11) Set MOPOP(CRPT_ECC_CTL[12:11]) to 01
12) Set START(CRPT_ECC_CTL[0]) to 1
13) Wait for BUSY(CRPT_ECC_STS[0]) be cleared
14) Write the curve order to N registers according to Table 6.26-1
15) Write e to Y1 registers
16) Set ECCOP(CRPT_ECC_CTL[10:9]) to 01
17) Set MOPOP(CRPT_ECC_CTL[12:11]) to 10
18) Set START(CRPT_ECC_CTL[0]) to 1
19) Wait for BUSY(CRPT_ECC_STS[0]) be cleared
20) Write the curve order and curve length to N, M registers according to Table 6.26-1
21) Write k-1 to Y1 registers
22) Set ECCOP(CRPT_ECC_CTL[10:9]) to 01
23) Set MOPOP(CRPT_ECC_CTL[12:11]) to 01
24) Set START(CRPT_ECC_CTL[0]) to 1
25) Wait for BUSY(CRPT_ECC_STS[0]) be cleared
26) Read X1 registers to get s

5. The signature is the pair (r, s)

ECDSA signature varification steps:

1. Verify that r and s are integers in the interval [1, n-1]. If not, the signature is invalid

2. Compute e = HASH (m), where HASH is the hashing algorithm in signature generation
    1) Use SHA to calculate e
3. Compute w = s −1 (mod n)
    1) Write the curve order to N registers according to Table 6.26-1
    2) Write s to X1 registers according to Table 6.26-1
    3) Write 0x1 to Y1 registers according to Table 6.26-1
    4) Set ECCOP(CRPT_ECC_CTL[10:9]) to 01
    5) Set MOPOP(CRPT_ECC_CTL[12:11]) to 00
    6) Set FSEL(CRPT_ECC_CTL[8]) according to used curve of prime field or binary field
    7) Set START(CRPT_ECC_CTL[0]) to 1
    8) Wait for BUSY(CRPT_ECC_STS[0]) be cleared
    9) Read X1 registers to get w
4. Compute u1 = e × w (mod n) and u2 = r × w (mod n)
    1) Write the curve order and curve length to N, M registers according to Table 6.26-1
    2) Write e, w to X1, Y1 registers
    3) Set ECCOP(CRPT_ECC_CTL[10:9]) to 01
    4) Set MOPOP(CRPT_ECC_CTL[12:11]) to 01
    5) Set START(CRPT_ECC_CTL[0]) to 1
    6) Wait for BUSY(CRPT_ECC_STS[0]) be cleared
    7) Read X1 registers to get u1
    8) Write the curve order and curve length to N, M registers according to Table 6.26-1
    9) Write r, w to X1, Y1 registers
    10) Set ECCOP(CRPT_ECC_CTL[10:9]) to 01
    11) Set MOPOP(CRPT_ECC_CTL[12:11]) to 01
    12) Set START(CRPT_ECC_CTL[0]) to 1
    13) Wait for BUSY(CRPT_ECC_STS[0]) be cleared
    14) Read X1 registers to get u2
5. Compute X’ (x1’, y1’) = u1 * G + u2 * Q
    1) Write the curve parameter A, B, N, and curve length M to corresponding registers according to Table 6.26-1
    2) Write the point G(x, y) to X1, Y1 registers
    3) Write u1 to K registers
    4) Set ECCOP(CRPT_ECC_CTL[10:9]) to 00
    5) Set START(CRPT_ECC_CTL[0]) to 1
    6) Wait for BUSY(CRPT_ECC_STS[0]) be cleared
    7) Read X1, Y1 registers to get u1*G

8) Write the curve parameter A, B, N, and curve length M to corresponding registers according to Table 6.26-1

9) Write the publick key Q(x,y) to X1, Y1 registers

10) Write u2 to K registers

11) Set ECCOP(CRPT_ECC_CTL[10:9]) to 00

12) Set START(CRPT_ECC_CTL[0]) to 1

13) Wait for BUSY(CRPT_ECC_STS[0]) be cleared

14) Write the curve parameter A, B, N, and curve length M to corresponding registers according to Table 6.26-1

15) Write the result data u1*G to X2, Y2 registers

16) Set ECCOP(CRPT_ECC_CTL[10:9]) to 10

17) Set START(CRPT_ECC_CTL[0]) to 1

18) Wait for BUSY(CRPT_ECC_STS[0]) be cleared

19) Read X1, Y1 registers to get X'(x1', y1')

20) Write the curve order and curve length to N, M registers according to Table 6.26-1

21) Write x1' to X1 registers

22) Write 0x0 to Y1 registers

23) Set ECCOP(CRPT_ECC_CTL[10:9]) to 01

24) Set MOPOP(CRPT_ECC_CTL[12:11]) to 10

25) Set START(CRPT_ECC_CTL[0]) to 1

26) Wait for BUSY(CRPT_ECC_STS[0]) be cleared

27) Read X1 registers to get x1' (mod n)

6. The signature is valid if x1' = r, otherwise it is invalid.

Elliptic Curve Diffie-Hellman

Share secret generation function: Z is the x-coordinate of Q where Q = dG (mod N)

1. Write the curve parameter A, B, N, and curve length M to corresponding registers according to Table 6.26-1.
2. Write the public key of receiving party G(x, y) to X1, Y1 registers according to Table 6.26-1.
3. Write (cofactor h * my private key d) to K register according to Table 6.26-1.
   1) h=1 in P-192, P-224, P-256, P-384 and P-521
   2) h=2 in B-163, B-233, B-283, B-409, B-571 and K-163
   3) h=4 in K-233, K-283, K-409 and K-571
4. Set ECCOP(CRPT_ECC_CTL[10:9]) to 00
5. Set FSEL(CRPT_ECC_CTL[8]) according to used curve of prime field or binary field
6. Set START(CRPT_ECC_CTL[0]) to 1
7. Wait for BUSY(CRPT_ECC_STS[0]) be cleared
8. Read public key Q from X1, Y1 registers

Hash-based key derivation function: DerivedKeyingMaterial = KDF(Z, OtherInput)

Step1 For i = 1 to reps, where reps = ceil((the length of Z, OtherInput)/hash length)

1. Write the curve parameter A, B, N, and curve length M to corresponding registers according to Table 6.26-1.
2. Write the public key of receiving party G(x, y) to X1, Y1 registers according to Table 6.26-1.

Step 2 Return DerivedKeyingMaterial = $Hash_1$ || $Hash_2$ || … || $Hash_{reps}$

Note:

1. the details of OtherInput please refer to the page 46 in NIST SP 800-56A, recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography (http://csrc.nist.gov/groups/ST/toolkit/documents/SP800-56Arev1_3-8-07.pdf)
2. the details of cofactor please refer to “RECOMMENDED ELLIPTIC CURVES FOR FEDERAL GOVERNMENT USE” (http://csrc.nist.gov/groups/ST/toolkit/documents/dss/NISTReCur.pdf)

6.26.5.7 RSA (Rivest、Shamir and Adleman) Cryptography

RSA is the first asymmetric cryptosystem (public-key cryptosystems) designed and named by Ron Rivest、Adi Shamir and Leonard Adleman from MIT in 1977. In the past years, many protocols and applications utilize the complexity of the prime factoring problem in the large semi-prime number to build cryptographic systems. The main computation of encryption and description in RSA is modulus exponentiation operation with very large bit length (as least 2048 bits from NIST’s suggestion). To accelerate RSA applications, we propose an RSA accelerator that can quickly compute the complex modulus exponentiation operation. Before starting RSA accelerator, users must provide the required input data of modulus exponentiation operation include the base of exponentiation (abbreviated to M), the exponent of exponentiation (abbreviated to E) and the base of modulus operation (abbriviated to N). Besides, the constant value of Montgomery domain (abbreviated to C) is required in RSA accelerator. The formula of computing C is shown in the below equation:

$C = 2^{(key\_length+2)*2}$ % N

From above equation, we can obtain that the value of C is changed with the value of N. That is, users only need to compute the value of C once by the below algorithm RSA_CS after producing every key pair in RSA. After modulus exponentiation operation finished, RSA accelerator will output a final result (i.e., M^E % N) in the register CRPT_RSA_M_i for 0 ≦i ≦63.

```
Algorithm: RSA_CS( )
// RSA Compare and subtract algorithm
// Input: N
// Output: C ( $C = 2^{(key\_length+2)*2}$ %N )
$C = 2^{(key\_length)} - N$;
for i = 0 to key_length + 3 {
    C = C << 1;
    if (C >=N) C = C - N ;
}
```

6. return C;

Users can refer to the following steps to understand how to program the RSA related registers.

RSA DMA Mode Programming Flow

1. Write 1 to RSAIEN(CRPT_INTEN [30]) to enable RSA interrupt if needed.
2. Program DMA source address to register CRPT_RSA_SADDR.
3. Program DMA destination address to register CRPT_RSA_DADDR.
4. Program DMA word count to register CRPT_RSA_WORDCNT.
5. Program the starting register address of all input data in Table 6.26-2 that will update to the register CRPT_ECC_STARTREG.
6. Write input data to DMA source address with selected DMA word count.
7. Configure RSA control register CRPT_RSA_CTL for RSA accelerator, such as the start signal of RSA accelerator(START), DMA mode enable signal(DMAEN), the control signals fo all input data registers(LDM, LDE, LDN, LDC), and the key length of elliptic curve(KEYLENGTH).
8. Wait for the RSA interrupt flag RSAIF(CRPT_INTSTS[30]) be set.
9. Read output data and then clear RSA interrupt flag RSAIF.

RSA Non-DMA Mode Programming Flow

1. Write all necessary input data in the corresponding registers, such as CRYPTO _RSA_M, CRYPTO _RSA_E, CRYPTO _RSA_N, CRPT_RSA_C.
2. Configure RSA control register CRPT_RSA_CTL for RSA accelerator, such as the start signal of RSA accelerator(START), the control signals fo all input data registers(LDM, LDE, LDN, LDC), and the key length of elliptic curve(KEYLENGTH).
3. Wait for the BUSY (CRPT_RSA_STS[0]) be cleared.
4. Read output digest and then clear RSA interrupt flag RSAIF.

Some Notices of RSA Accelerator

1. The key length support of RSA accelerator is up to 2048 bits and must be a multiple to four.
2. All input and output data must be positive.
3. Only when START and DMAEN (CRPT_RSA_CTL[0] and [7]) are assigned to 1 simultaneously, RSA DMA mode will be active.
4. When RSA engine is active (i.e., BUSY is 1 and DMABUSY (CRPT_RSA_STS[1]) is 0), user can’t modify all input data registers (CRPT_RSA_M_i ~ CRPT_ECC_C_i).
5. If user wants to stop RSA accelerator, please configures the STOP (CRPT_RSA_CTL[1]) to 1. Note that to avoid the transmission error of the next operation, BUSY signal will not be cleared immediately until the action of DMA is done.

6.26.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
CRYP Base Address: **CRYP_BA = 0xB001_C000**				
CRPT_INTEN	CRYP_BA+0x000	R/W	Crypto Interrupt Enable Control Register	0x0000_0000
CRPT_INTSTS	CRYP_BA+0x004	R/W	Crypto Interrupt Flag	0x0000_0000
CRPT_PRNG_CTL	CRYP_BA+0x008	R/W	PRNG Control Register	0x0000_0000
CRPT_PRNG_SEED	CRYP_BA+0x00C	W	Seed for PRNG	Undefined
CRPT_PRNG_KEY0	CRYP_BA+0x010	R	PRNG Generated Key0	Undefined
CRPT_PRNG_KEY1	CRYP_BA+0x014	R	PRNG Generated Key1	Undefined
CRPT_PRNG_KEY2	CRYP_BA+0x018	R	PRNG Generated Key2	Undefined
CRPT_PRNG_KEY3	CRYP_BA+0x01C	R	PRNG Generated Key3	Undefined
CRPT_PRNG_KEY4	CRYP_BA+0x020	R	PRNG Generated Key4	Undefined
CRPT_PRNG_KEY5	CRYP_BA+0x024	R	PRNG Generated Key5	Undefined
CRPT_PRNG_KEY6	CRYP_BA+0x028	R	PRNG Generated Key6	Undefined
CRPT_PRNG_KEY7	CRYP_BA+0x02C	R	PRNG Generated Key7	Undefined
CRPT_AES_FDBCK0	CRYP_BA+0x050	R	AES Engine Output Feedback Data After Cryptographic Operation	0x0000_0000
CRPT_AES_FDBCK1	CRYP_BA+0x054	R	AES Engine Output Feedback Data After Cryptographic Operation	0x0000_0000
CRPT_AES_FDBCK2	CRYP_BA+0x058	R	AES Engine Output Feedback Data After Cryptographic Operation	0x0000_0000
CRPT_AES_FDBCK3	CRYP_BA+0x05C	R	AES Engine Output Feedback Data After Cryptographic Operation	0x0000_0000
CRPT_AES_CTL	CRYP_BA+0x100	R/W	AES Control Register	0x0000_0000
CRPT_AES_STS	CRYP_BA+0x104	R	AES Engine Flag	0x0001_0100
CRPT_AES_DATIN	CRYP_BA+0x108	R/W	AES Engine Data Input Port Register	0x0000_0000
CRPT_AES_DATOUT	CRYP_BA+0x10C	R	AES Engine Data Output Port Register	0x0000_0000
CRPT_AES0_KEY0	CRYP_BA+0x110	R/W	AES Key Word 0 Register for Channel 0	0x0000_0000
CRPT_AES0_KEY1	CRYP_BA+0x114	R/W	AES Key Word 1 Register for Channel 0	0x0000_0000
CRPT_AES0_KEY2	CRYP_BA+0x118	R/W	AES Key Word 2 Register for Channel 0	0x0000_0000
CRPT_AES0_KEY3	CRYP_BA+0x11C	R/W	AES Key Word 3 Register for Channel 0	0x0000_0000
CRPT_AES0_KEY4	CRYP_BA+0x120	R/W	AES Key Word 4 Register for Channel 0	0x0000_0000
CRPT_AES0_KEY5	CRYP_BA+0x124	R/W	AES Key Word 5 Register for Channel 0	0x0000_0000
CRPT_AES0_KEY6	CRYP_BA+0x128	R/W	AES Key Word 6 Register for Channel 0	0x0000_0000

CRPT_AES0_KEY7	CRYP_BA+0x12C	R/W	AES Key Word 7 Register for Channel 0	0x0000_0000
CRPT_AES0_IV0	CRYP_BA+0x130	R/W	AES Initial Vector Word 0 Register for Channel 0	0x0000_0000
CRPT_AES0_IV1	CRYP_BA+0x134	R/W	AES Initial Vector Word 1 Register for Channel 0	0x0000_0000
CRPT_AES0_IV2	CRYP_BA+0x138	R/W	AES Initial Vector Word 2 Register for Channel 0	0x0000_0000
CRPT_AES0_IV3	CRYP_BA+0x13C	R/W	AES Initial Vector Word 3 Register for Channel 0	0x0000_0000
CRPT_AES0_SADDR	CRYP_BA+0x140	R/W	AES DMA Source Address Register for Channel 0	0x0000_0000
CRPT_AES0_DADDR	CRYP_BA+0x144	R/W	AES DMA Destination Address Register for Channel 0	0x0000_0000
CRPT_AES0_CNT	CRYP_BA+0x148	R/W	AES Byte Count Register for Channel 0	0x0000_0000
CRPT_HMAC_CTL	CRYP_BA+0x300	R/W	SHA/HMAC Control Register	0x0000_0000
CRPT_HMAC_STS	CRYP_BA+0x304	R	SHA/HMAC Status Flag	0x0000_0000
CRPT_HMAC_DGST0	CRYP_BA+0x308	R	SHA/HMAC Digest Message 0	0x0000_0000
CRPT_HMAC_DGST1	CRYP_BA+0x30C	R	SHA/HMAC Digest Message 1	0x0000_0000
CRPT_HMAC_DGST2	CRYP_BA+0x310	R	SHA/HMAC Digest Message 2	0x0000_0000
CRPT_HMAC_DGST3	CRYP_BA+0x314	R	SHA/HMAC Digest Message 3	0x0000_0000
CRPT_HMAC_DGST4	CRYP_BA+0x318	R	SHA/HMAC Digest Message 4	0x0000_0000
CRPT_HMAC_DGST5	CRYP_BA+0x31C	R	SHA/HMAC Digest Message 5	0x0000_0000
CRPT_HMAC_DGST6	CRYP_BA+0x320	R	SHA/HMAC Digest Message 6	0x0000_0000
CRPT_HMAC_DGST7	CRYP_BA+0x324	R	SHA/HMAC Digest Message 7	0x0000_0000
CRPT_HMAC_DGST8	CRYP_BA+0x328	R	SHA/HMAC Digest Message 8	0x0000_0000
CRPT_HMAC_DGST9	CRYP_BA+0x32C	R	SHA/HMAC Digest Message 9	0x0000_0000
CRPT_HMAC_DGST10	CRYP_BA+0x330	R	SHA/HMAC Digest Message 10	0x0000_0000
CRPT_HMAC_DGST11	CRYP_BA+0x334	R	SHA/HMAC Digest Message 11	0x0000_0000
CRPT_HMAC_DGST12	CRYP_BA+0x338	R	SHA/HMAC Digest Message 12	0x0000_0000
CRPT_HMAC_DGST13	CRYP_BA+0x33C	R	SHA/HMAC Digest Message 13	0x0000_0000
CRPT_HMAC_DGST14	CRYP_BA+0x340	R	SHA/HMAC Digest Message 14	0x0000_0000
CRPT_HMAC_DGST15	CRYP_BA+0x344	R	SHA/HMAC Digest Message 15	0x0000_0000
CRPT_HMAC_KEYCNT	CRYP_BA+0x348	R/W	SHA/HMAC Key Byte Count Register	0x0000_0000
CRPT_HMAC_SADDR	CRYP_BA+0x34C	R/W	SHA/HMAC DMA Source Address Register	0x0000_0000
CRPT_HMAC_DMACNT	CRYP_BA+0x350	R/W	SHA/HMAC Byte Count Register	0x0000_0000
CRPT_HMAC_DATIN	CRYP_BA+0x354	R/W	SHA/HMAC Engine Non-dMA Mode Data Input Port Register	0x0000_0000
CRPT_ECC_CTL	CRYP_BA+0x800	R/W	ECC Control Register	0x0000_0000
CRPT_ECC_STS	CRYP_BA+0x804	R	ECC Status Register	0x0000_0000
CRPT_ECC_X1_00	CRYP_BA+0x808	R/W	ECC the X-coordinate Word0 of the First Point	0x0000_0000

CRPT_ECC_X1_01	CRYP_BA+0x80C	R/W	ECC the X-coordinate Word1 of the First Point	0x0000_0000
CRPT_ECC_X1_02	CRYP_BA+0x810	R/W	ECC the X-coordinate Word2 of the First Point	0x0000_0000
CRPT_ECC_X1_03	CRYP_BA+0x814	R/W	ECC the X-coordinate Word3 of the First Point	0x0000_0000
CRPT_ECC_X1_04	CRYP_BA+0x818	R/W	ECC the X-coordinate Word4 of the First Point	0x0000_0000
CRPT_ECC_X1_05	CRYP_BA+0x81C	R/W	ECC the X-coordinate Word5 of the First Point	0x0000_0000
CRPT_ECC_X1_06	CRYP_BA+0x820	R/W	ECC the X-coordinate Word6 of the First Point	0x0000_0000
CRPT_ECC_X1_07	CRYP_BA+0x824	R/W	ECC the X-coordinate Word7 of the First Point	0x0000_0000
CRPT_ECC_X1_08	CRYP_BA+0x828	R/W	ECC the X-coordinate Word8 of the First Point	0x0000_0000
CRPT_ECC_X1_09	CRYP_BA+0x82C	R/W	ECC the X-coordinate Word9 of the First Point	0x0000_0000
CRPT_ECC_X1_10	CRYP_BA+0x830	R/W	ECC the X-coordinate Word10 of the First Point	0x0000_0000
CRPT_ECC_X1_11	CRYP_BA+0x834	R/W	ECC the X-coordinate Word11 of the First Point	0x0000_0000
CRPT_ECC_X1_12	CRYP_BA+0x838	R/W	ECC the X-coordinate Word12 of the First Point	0x0000_0000
CRPT_ECC_X1_13	CRYP_BA+0x83C	R/W	ECC the X-coordinate Word13 of the First Point	0x0000_0000
CRPT_ECC_X1_14	CRYP_BA+0x840	R/W	ECC the X-coordinate Word14 of the First Point	0x0000_0000
CRPT_ECC_X1_15	CRYP_BA+0x844	R/W	ECC the X-coordinate Word15 of the First Point	0x0000_0000
CRPT_ECC_X1_16	CRYP_BA+0x848	R/W	ECC the X-coordinate Word16 of the First Point	0x0000_0000
CRPT_ECC_X1_17	CRYP_BA+0x84C	R/W	ECC the X-coordinate Word17 of the First Point	0x0000_0000
CRPT_ECC_Y1_00	CRYP_BA+0x850	R/W	ECC the Y-coordinate Word0 of the First Point	0x0000_0000
CRPT_ECC_Y1_01	CRYP_BA+0x854	R/W	ECC the Y-coordinate Word1 of the First Point	0x0000_0000
CRPT_ECC_Y1_02	CRYP_BA+0x858	R/W	ECC the Y-coordinate Word2 of the First Point	0x0000_0000
CRPT_ECC_Y1_03	CRYP_BA+0x85C	R/W	ECC the Y-coordinate Word3 of the First Point	0x0000_0000
CRPT_ECC_Y1_04	CRYP_BA+0x860	R/W	ECC the Y-coordinate Word4 of the First Point	0x0000_0000
CRPT_ECC_Y1_05	CRYP_BA+0x864	R/W	ECC the Y-coordinate Word5 of the First Point	0x0000_0000
CRPT_ECC_Y1_06	CRYP_BA+0x868	R/W	ECC the Y-coordinate Word6 of the First Point	0x0000_0000
CRPT_ECC_Y1_07	CRYP_BA+0x86C	R/W	ECC the Y-coordinate Word7 of the First Point	0x0000_0000
CRPT_ECC_Y1_08	CRYP_BA+0x870	R/W	ECC the Y-coordinate Word8 of the First Point	0x0000_0000
CRPT_ECC_Y1_09	CRYP_BA+0x874	R/W	ECC the Y-coordinate Word9 of the First Point	0x0000_0000
CRPT_ECC_Y1_10	CRYP_BA+0x878	R/W	ECC the Y-coordinate Word10 of the First Point	0x0000_0000
CRPT_ECC_Y1_11	CRYP_BA+0x87C	R/W	ECC the Y-coordinate Word11 of the First Point	0x0000_0000
CRPT_ECC_Y1_12	CRYP_BA+0x880	R/W	ECC the Y-coordinate Word12 of the First Point	0x0000_0000
CRPT_ECC_Y1_13	CRYP_BA+0x884	R/W	ECC the Y-coordinate Word13 of the First Point	0x0000_0000
CRPT_ECC_Y1_14	CRYP_BA+0x888	R/W	ECC the Y-coordinate Word14 of the First Point	0x0000_0000
CRPT_ECC_Y1_15	CRYP_BA+0x88C	R/W	ECC the Y-coordinate Word15 of the First Point	0x0000_0000
CRPT_ECC_Y1_16	CRYP_BA+0x890	R/W	ECC the Y-coordinate Word16 of the First Point	0x0000_0000

CRPT_ECC_Y1_17	CRYP_BA+0x894	R/W	ECC the Y-coordinate Word17 of the First Point	0x0000_0000
CRPT_ECC_X2_00	CRYP_BA+0x898	R/W	ECC the X-coordinate Word0 of the Second Point	0x0000_0000
CRPT_ECC_X2_01	CRYP_BA+0x89C	R/W	ECC the X-coordinate Word1 of the Second Point	0x0000_0000
CRPT_ECC_X2_02	CRYP_BA+0x8A0	R/W	ECC the X-coordinate Word2 of the Second Point	0x0000_0000
CRPT_ECC_X2_03	CRYP_BA+0x8A4	R/W	ECC the X-coordinate Word3 of the Second Point	0x0000_0000
CRPT_ECC_X2_04	CRYP_BA+0x8A8	R/W	ECC the X-coordinate Word4 of the Second Point	0x0000_0000
CRPT_ECC_X2_05	CRYP_BA+0x8AC	R/W	ECC the X-coordinate Word5 of the Second Point	0x0000_0000
CRPT_ECC_X2_06	CRYP_BA+0x8B0	R/W	ECC the X-coordinate Word6 of the Second Point	0x0000_0000
CRPT_ECC_X2_07	CRYP_BA+0x8B4	R/W	ECC the X-coordinate Word7 of the Second Point	0x0000_0000
CRPT_ECC_X2_08	CRYP_BA+0x8B8	R/W	ECC the X-coordinate Word8 of the Second Point	0x0000_0000
CRPT_ECC_X2_09	CRYP_BA+0x8BC	R/W	ECC the X-coordinate Word9 of the Second Point	0x0000_0000
CRPT_ECC_X2_10	CRYP_BA+0x8C0	R/W	ECC the X-coordinate Word10 of the Second Point	0x0000_0000
CRPT_ECC_X2_11	CRYP_BA+0x8C4	R/W	ECC the X-coordinate Word11 of the Second Point	0x0000_0000
CRPT_ECC_X2_12	CRYP_BA+0x8C8	R/W	ECC the X-coordinate Word12 of the Second Point	0x0000_0000
CRPT_ECC_X2_13	CRYP_BA+0x8CC	R/W	ECC the X-coordinate Word13 of the Second Point	0x0000_0000
CRPT_ECC_X2_14	CRYP_BA+0x8D0	R/W	ECC the X-coordinate Word14 of the Second Point	0x0000_0000
CRPT_ECC_X2_15	CRYP_BA+0x8D4	R/W	ECC the X-coordinate Word15 of the Second Point	0x0000_0000
CRPT_ECC_X2_16	CRYP_BA+0x8D8	R/W	ECC the X-coordinate Word16 of the Second Point	0x0000_0000
CRPT_ECC_X2_17	CRYP_BA+0x8DC	R/W	ECC the X-coordinate Word17 of the Second Point	0x0000_0000
CRPT_ECC_Y2_00	CRYP_BA+0x8E0	R/W	ECC the Y-coordinate Word0 of the Second Point	0x0000_0000
CRPT_ECC_Y2_01	CRYP_BA+0x8E4	R/W	ECC the Y-coordinate Word1 of the Second Point	0x0000_0000
CRPT_ECC_Y2_02	CRYP_BA+0x8E8	R/W	ECC the Y-coordinate Word2 of the Second Point	0x0000_0000
CRPT_ECC_Y2_03	CRYP_BA+0x8EC	R/W	ECC the Y-coordinate Word3 of the Second Point	0x0000_0000
CRPT_ECC_Y2_04	CRYP_BA+0x8F0	R/W	ECC the Y-coordinate Word4 of the Second Point	0x0000_0000
CRPT_ECC_Y2_05	CRYP_BA+0x8F4	R/W	ECC the Y-coordinate Word5 of the Second Point	0x0000_0000
CRPT_ECC_Y2_06	CRYP_BA+0x8F8	R/W	ECC the Y-coordinate Word6 of the Second Point	0x0000_0000
CRPT_ECC_Y2_07	CRYP_BA+0x8FC	R/W	ECC the Y-coordinate Word7 of the Second Point	0x0000_0000
CRPT_ECC_Y2_08	CRYP_BA+0x900	R/W	ECC the Y-coordinate Word8 of the Second Point	0x0000_0000
CRPT_ECC_Y2_09	CRYP_BA+0x904	R/W	ECC the Y-coordinate Word9 of the Second Point	0x0000_0000
CRPT_ECC_Y2_10	CRYP_BA+0x908	R/W	ECC the Y-coordinate Word10 of the Second Point	0x0000_0000
CRPT_ECC_Y2_11	CRYP_BA+0x90C	R/W	ECC the Y-coordinate Word11 of the Second Point	0x0000_0000
CRPT_ECC_Y2_12	CRYP_BA+0x910	R/W	ECC the Y-coordinate Word12 of the Second Point	0x0000_0000
CRPT_ECC_Y2_13	CRYP_BA+0x914	R/W	ECC the Y-coordinate Word13 of the Second Point	0x0000_0000
CRPT_ECC_Y2_14	CRYP_BA+0x918	R/W	ECC the Y-coordinate Word14 of the Second Point	0x0000_0000

CRPT_ECC_Y2_15	CRYP_BA+0x91C	R/W	ECC the Y-coordinate Word15 of the Second Point	0x0000_0000
CRPT_ECC_Y2_16	CRYP_BA+0x920	R/W	ECC the Y-coordinate Word16 of the Second Point	0x0000_0000
CRPT_ECC_Y2_17	CRYP_BA+0x924	R/W	ECC the Y-coordinate Word17 of the Second Point	0x0000_0000
CRPT_ECC_A_00	CRYP_BA+0x928	R/W	ECC the Parameter CURVEA Word0 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_01	CRYP_BA+0x92C	R/W	ECC the Parameter CURVEA Word1 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_02	CRYP_BA+0x930	R/W	ECC the Parameter CURVEA Word2 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_03	CRYP_BA+0x934	R/W	ECC the Parameter CURVEA Word3 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_04	CRYP_BA+0x938	R/W	ECC the Parameter CURVEA Word4 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_05	CRYP_BA+0x93C	R/W	ECC the Parameter CURVEA Word5 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_06	CRYP_BA+0x940	R/W	ECC the Parameter CURVEA Word6 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_07	CRYP_BA+0x944	R/W	ECC the Parameter CURVEA Word7 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_08	CRYP_BA+0x948	R/W	ECC the Parameter CURVEA Word8 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_09	CRYP_BA+0x94C	R/W	ECC the Parameter CURVEA Word9 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_10	CRYP_BA+0x950	R/W	ECC the Parameter CURVEA Word10 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_11	CRYP_BA+0x954	R/W	ECC the Parameter CURVEA Word11 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_12	CRYP_BA+0x958	R/W	ECC the Parameter CURVEA Word12 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_13	CRYP_BA+0x95C	R/W	ECC the Parameter CURVEA Word13 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_14	CRYP_BA+0x960	R/W	ECC the Parameter CURVEA Word14 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_15	CRYP_BA+0x964	R/W	ECC the Parameter CURVEA Word15 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_16	CRYP_BA+0x968	R/W	ECC the Parameter CURVEA Word16 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_17	CRYP_BA+0x96C	R/W	ECC the Parameter CURVEA Word17 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_00	CRYP_BA+0x970	R/W	ECC the Parameter CURVEB Word0 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_01	CRYP_BA+0x974	R/W	ECC the Parameter CURVEB Word1 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_02	CRYP_BA+0x978	R/W	ECC the Parameter CURVEB Word2 of Elliptic Curve	0x0000_0000

NUC980 SERIES TECHNICAL REFERENCE MANUAL

CRPT_ECC_B_03	CRYP_BA+0x97C	R/W	ECC the Parameter CURVEB Word3 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_04	CRYP_BA+0x980	R/W	ECC the Parameter CURVEB Word4 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_05	CRYP_BA+0x984	R/W	ECC the Parameter CURVEB Word5 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_06	CRYP_BA+0x988	R/W	ECC the Parameter CURVEB Word6 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_07	CRYP_BA+0x98C	R/W	ECC the Parameter CURVEB Word7 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_08	CRYP_BA+0x990	R/W	ECC the Parameter CURVEB Word8 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_09	CRYP_BA+0x994	R/W	ECC the Parameter CURVEB Word9 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_10	CRYP_BA+0x998	R/W	ECC the Parameter CURVEB Word10 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_11	CRYP_BA+0x99C	R/W	ECC the Parameter CURVEB Word11 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_12	CRYP_BA+0x9A0	R/W	ECC the Parameter CURVEB Word12 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_13	CRYP_BA+0x9A4	R/W	ECC the Parameter CURVEB Word13 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_14	CRYP_BA+0x9A8	R/W	ECC the Parameter CURVEB Word14 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_15	CRYP_BA+0x9AC	R/W	ECC the Parameter CURVEB Word15 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_16	CRYP_BA+0x9B0	R/W	ECC the Parameter CURVEB Word16 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_17	CRYP_BA+0x9B4	R/W	ECC the Parameter CURVEB Word17 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_00	CRYP_BA+0x9B8	R/W	ECC the Parameter CURVEN Word0 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_01	CRYP_BA+0x9BC	R/W	ECC the Parameter CURVEN Word1 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_02	CRYP_BA+0x9C0	R/W	ECC the Parameter CURVEN Word2 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_03	CRYP_BA+0x9C4	R/W	ECC the Parameter CURVEN Word3 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_04	CRYP_BA+0x9C8	R/W	ECC the Parameter CURVEN Word4 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_05	CRYP_BA+0x9CC	R/W	ECC the Parameter CURVEN Word5 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_06	CRYP_BA+0x9D0	R/W	ECC the Parameter CURVEN Word6 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_07	CRYP_BA+0x9D4	R/W	ECC the Parameter CURVEN Word7 of Elliptic Curve	0x0000_0000

CRPT_ECC_N_08	CRYP_BA+0x9D8	R/W	ECC the Parameter CURVEN Word8 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_09	CRYP_BA+0x9DC	R/W	ECC the Parameter CURVEN Word9 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_10	CRYP_BA+0x9E0	R/W	ECC the Parameter CURVEN Word10 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_11	CRYP_BA+0x9E4	R/W	ECC the Parameter CURVEN Word11 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_12	CRYP_BA+0x9E8	R/W	ECC the Parameter CURVEN Word12 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_13	CRYP_BA+0x9EC	R/W	ECC the Parameter CURVEN Word13 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_14	CRYP_BA+0x9F0	R/W	ECC the Parameter CURVEN Word14 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_15	CRYP_BA+0x9F4	R/W	ECC the Parameter CURVEN Word15 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_16	CRYP_BA+0x9F8	R/W	ECC the Parameter CURVEN Word16 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_17	CRYP_BA+0x9FC	R/W	ECC the Parameter CURVEN Word17 of Elliptic Curve	0x0000_0000
CRPT_ECC_K_00	CRYP_BA+0xA00	W	ECC the Scalar SCALARK Word0 of Point Multiplication	0x0000_0000
CRPT_ECC_K_01	CRYP_BA+0xA04	W	ECC the Scalar SCALARK Word1 of Point Multiplication	0x0000_0000
CRPT_ECC_K_02	CRYP_BA+0xA08	W	ECC the Scalar SCALARK Word2 of Point Multiplication	0x0000_0000
CRPT_ECC_K_03	CRYP_BA+0xA0C	W	ECC the Scalar SCALARK Word3 of Point Multiplication	0x0000_0000
CRPT_ECC_K_04	CRYP_BA+0xA10	W	ECC the Scalar SCALARK Word4 of Point Multiplication	0x0000_0000
CRPT_ECC_K_05	CRYP_BA+0xA14	W	ECC the Scalar SCALARK Word5 of Point Multiplication	0x0000_0000
CRPT_ECC_K_06	CRYP_BA+0xA18	W	ECC the Scalar SCALARK Word6 of Point Multiplication	0x0000_0000
CRPT_ECC_K_07	CRYP_BA+0xA1C	W	ECC the Scalar SCALARK Word7 of Point Multiplication	0x0000_0000
CRPT_ECC_K_08	CRYP_BA+0xA20	W	ECC the Scalar SCALARK Word8 of Point Multiplication	0x0000_0000
CRPT_ECC_K_09	CRYP_BA+0xA24	W	ECC the Scalar SCALARK Word9 of Point Multiplication	0x0000_0000
CRPT_ECC_K_10	CRYP_BA+0xA28	W	ECC the Scalar SCALARK Word10 of Point Multiplication	0x0000_0000
CRPT_ECC_K_11	CRYP_BA+0xA2C	W	ECC the Scalar SCALARK Word11 of Point Multiplication	0x0000_0000
CRPT_ECC_K_12	CRYP_BA+0xA30	W	ECC the Scalar SCALARK Word12 of Point Multiplication	0x0000_0000

CRPT_ECC_K_13	CRYP_BA+0xA34	W	ECC the Scalar SCALARK Word13 of Point Multiplication	0x0000_0000
CRPT_ECC_K_14	CRYP_BA+0xA38	W	ECC the Scalar SCALARK Word14 of Point Multiplication	0x0000_0000
CRPT_ECC_K_15	CRYP_BA+0xA3C	W	ECC the Scalar SCALARK Word15 of Point Multiplication	0x0000_0000
CRPT_ECC_K_16	CRYP_BA+0xA40	W	ECC the Scalar SCALARK Word16 of Point Multiplication	0x0000_0000
CRPT_ECC_K_17	CRYP_BA+0xA44	W	ECC the Scalar SCALARK Word17 of Point Multiplication	0x0000_0000
CRPT_ECC_SADDR	CRYP_BA+0xA48	R/W	ECC DMA Source Address Register	0x0000_0000
CRPT_ECC_DADDR	CRYP_BA+0xA4C	R/W	ECC DMA Destination Address Register	0x0000_0000
CRPT_ECC_STARTREG	CRYP_BA+0xA50	R/W	ECC Starting Address of Updated Registers	0x0000_0000
CRPT_ECC_WORDCNT	CRYP_BA+0xA54	R/W	ECC DMA Word Count	0x0000_0000
CRPT_RSA_CTL	CRYP_BA+0x1000	R/W	RSA Control Register	0x0000_0000
CRPT_RSA_STS	CRYP_BA+0x1004	R	RSA Status Register	0x0000_0000
CRPT_RSA_M_i **i=0,1..63**	CRYP_BA+0x1008+ 0x4*i	R/W	RSA the Base of Exponentiation Word i	0x0000_0000
CRPT_RSA_E_i **i=0,1..63**	CRYP_BA+0x1208+ 0x4*i	W	RSA the Exponent of Exponentiation Word i	0x0000_0000
CRPT_RSA_N_i **i=0,1..63**	CRYP_BA+0x1408+ 0x4*i	R/W	RSA the Base of Modulus Operation Word i	0x0000_0000
CRPT_RSA_C_i **i=0,1..63**	CRYP_BA+0x1608+ 0x4*i	R/W	RSA the Constant Value of Montgomery Domain Word i	0x0000_0000
CRPT_RSA_SADDR	CRYP_BA+0x1808	R/W	RSA DMA Source Address Register	0x0000_0000
CRPT_RSA_DADDR	CRYP_BA+0x180C	R/W	RSA DMA Destination Address Register	0x0000_0000
CRPT_RSA_STARTREG	CRYP_BA+0x1810	R/W	RSA Starting Address of Updated Registers	0x0000_0000
CRPT_RSA_WORDCNT	CRYP_BA+0x1814	R/W	RSA DMA Word Count	0x0000_0000

6.26.7 Register Description

6.26.7.1 Crypto Register

CRYPTO Interrupt Enable Control Register (CRPT_INTEN)

Register	Offset	R/W	Description	Reset Value
CRPT_INTEN	CRYP_BA+0x000	R/W	Crypto Interrupt Enable Control Register	0x0000_0000

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

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25]	**HMACEIEN**	**SHA/HMAC Error Interrupt Enable Control** 0 = SHA/HMAC error interrupt flag Disabled. 1 = SHA/HMAC error interrupt flag Enabled.
[24]	**HMACIEN**	**SHA/HMAC Interrupt Enable Control** 0 = SHA/HMAC interrupt Disabled. 1 = SHA/HMAC interrupt Enabled. In DMA mode, an interrupt will be triggered when amount of data set in SHA _DMA_CNT is fed into the SHA/HMAC engine. In Non-DMA mode, an interrupt will be triggered when the SHA/HMAC engine finishes the operation.
[23:17]	**Reserved**	Reserved.
[16]	**PRNGIEN**	**PRNG Interrupt Enable Control** 0 = PRNG interrupt Disabled. 1 = PRNG interrupt Enabled.
[15:8]	**Reserved**	Reserved.
[7:2]	**Reserved**	Reserved.
[1]	**AESEIEN**	**AES Error Flag Enable Control** 0 = AES error interrupt flag Disabled. 1 = AES error interrupt flag Enabled.

[0]	**AESIEN**	**AES Interrupt Enable Control** 0 = AES interrupt Disabled. 1 = AES interrupt Enabled. In DMA mode, an interrupt will be triggered when amount of data set in AES_DMA_CNT is fed into the AES engine. In Non-DMA mode, an interrupt will be triggered when the AES engine finishes the operation.

CRYPTO Interrupt Flag Register (CRPT_INTSTS)

Register	Offset	R/W	Description	Reset Value
CRPT_INTSTS	CRYP_BA+0x004	R/W	Crypto Interrupt Flag	0x0000_0000

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

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25]	**HMACEIF**	**SHA/HMAC Error Flag** This register includes operating and setting error. The detail flag is shown in SHA _FLAG register. This bit is cleared by writing 1, and it has no effect by writing 0. 0 = No SHA/HMAC error. 1 = SHA/HMAC error interrupt.
[24]	**HMACIF**	**SHA/HMAC Finish Interrupt Flag** This bit is cleared by writing 1, and it has no effect by writing 0. 0 = No SHA/HMAC interrupt. 1 = SHA/HMAC operation done interrupt.
[23:17]	**Reserved**	Reserved.
[16]	**PRNGIF**	**PRNG Finish Interrupt Flag** This bit is cleared by writing 1, and it has no effect by writing 0. 0 = No PRNG interrupt. 1 = PRNG key generation done interrupt.
[15:8]	**Reserved**	Reserved.
[7:2]	**Reserved**	Reserved.
[1]	**AESEIF**	**AES Error Flag** This bit is cleared by writing 1, and it has no effect by writing 0. 0 = No AES error. 1 = AES encryption/decryption done interrupt.

[0]	**AESIF**	**AES Finish Interrupt Flag** This bit is cleared by writing 1, and it has no effect by writing 0. 0 = No AES interrupt. 1 = AES encryption/decryption done interrupt.

6.26.7.2

6.26.7.3 *PRNG Register*

PRNG Control Register (CRPT_PRNG_CTL)

Register	Offset	R/W	Description	Reset Value
CRPT_PRNG_CTL	CRYP_BA+0x008	R/W	PRNG Control Register	0x0000_0000

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

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**BUSY**	**PRNG Busy (Read Only)** 0 = PRNG engine is idle. 1 = Indicate that the PRNG engine is generating CRPT_PRNG_KEYx.
[7:4]	**Reserved**	Reserved.
[3:2]	**KEYSZ**	**PRNG Generate Key Size** 00 = 64 bits. 01 = 128 bits. 10 = 192 bits. 11 = 256 bits.
[1]	**SEEDRLD**	**Reload New Seed for PRNG Engine** 0 = Generating key based on the current seed. 1 = Reload new seed.
[0]	**START**	**Start PRNG Engine** 0 = Stop PRNG engine. 1 = Generate new key and store the new key to register CRPT_PRNG_KEYx, which will be cleared when the new key is generated.

PRNG Seed Register (CRPT_PRNG_SEED)

Register	Offset	R/W	Description	Reset Value
CRPT_PRNG_SEED	CRYP_BA+0x00C	W	Seed for PRNG	Undefined

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

Bits	Description	
[31:0]	**SEED**	**Seed for PRNG (Write Only)** The bits store the seed for PRNG engine.

PRNG Key x Register (CRPT_PRNG_KEYx)

Register	Offset	R/W	Description	Reset Value
CRPT_PRNG_KEY0	CRYP_BA+0x010	R	PRNG Generated Key0	Undefined
CRPT_PRNG_KEY1	CRYP_BA+0x014	R	PRNG Generated Key1	Undefined
CRPT_PRNG_KEY2	CRYP_BA+0x018	R	PRNG Generated Key2	Undefined
CRPT_PRNG_KEY3	CRYP_BA+0x01C	R	PRNG Generated Key3	Undefined
CRPT_PRNG_KEY4	CRYP_BA+0x020	R	PRNG Generated Key4	Undefined
CRPT_PRNG_KEY5	CRYP_BA+0x024	R	PRNG Generated Key5	Undefined
CRPT_PRNG_KEY6	CRYP_BA+0x028	R	PRNG Generated Key6	Undefined
CRPT_PRNG_KEY7	CRYP_BA+0x02C	R	PRNG Generated Key7	Undefined

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

Bits	Description	
[31:0]	**KEY**	**Store PRNG Generated Key (Read Only)** The bits store the key that is generated by PRNG.

6.26.7.4 AES Register

AES Control Register (CRPT_AES_CTL)

Register	Offset	R/W	Description	Reset Value
CRPT_AES_CTL	CRYP_BA+0x100	R/W	AES Control Register	0x0000_0000

31	30	29	28	27	26	25	24
KEYPRT	KEYUNPRT					Reserved	
23	22	21	20	19	18	17	16
INSWAP	OUTSWAP	Reserved					ENCRPT
15	14	13	12	11	10	9	8
OPMODE							
7	6	5	4	3	2	1	0
DMAEN	DMACSCAD	DMALAST	EXTKEY	KEYSZ		STOP	START

Bits	Description	
[31]	**KEYPRT**	**Protect Key** Read as a flag to reflect KEYPRT. 0 = No effect. 1 = Protect the content of the AES key from reading. The return value for reading CRPT_AESn_KEYx is not the content of the registers CRPT_AESn_KEYx. Once it is set, it can be cleared by asserting KEYUNPRT. And the key content would be cleared as well.
[30:26]	**KEYUNPRT**	**Unprotect Key** Writing 0 to CRPT_AES_CTL[31] and “10110” to CRPT_AES_CTL[30:26] is to unprotect the AES key. The KEYUNPRT can be read and written. When it is written as the AES engine is operating, BUSY flag is 1, there would be no effect on KEYUNPRT.
[25:24]	**Reserved**	Reserved.
[23]	**INSWAP**	**AES Engine Input Data Swap** 0 = Keep the original order. 1 = The order that CPU feeds data to the accelerator will be changed from {byte3, byte2, byte1, byte0} to {byte0, byte1, byte2, byte3}.
[22]	**OUTSWAP**	**AES Engine Output Data Swap** 0 = Keep the original order. 1 = The order that CPU outputs data from the accelerator will be changed from {byte3, byte2, byte1, byte0} to {byte0, byte1, byte2, byte3}.
[16]	**ENCRPT**	**AES Encryption/Decryption** 0 = AES engine executes decryption operation. 1 = AES engine executes encryption operation.

[15:8]	**OPMODE**	**AES Engine Operation Modes** 0x00 = ECB (Electronic Codebook Mode) 0x01 = CBC (Cipher Block Chaining Mode). 0x02 = CFB (Cipher Feedback Mode). 0x03 = OFB (Output Feedback Mode). 0x04 = CTR (Counter Mode). 0x10 = CBC-CS1 (CBC Ciphertext-Stealing 1 Mode). 0x11 = CBC-CS2 (CBC Ciphertext-Stealing 2 Mode). 0x12 = CBC-CS3 (CBC Ciphertext-Stealing 3 Mode).
[7]	**DMAEN**	**AES Engine DMA Enable Control** 0 = AES DMA engine Disabled. The AES engine operates in Non-DMA mode, and gets data from the port CRPT_AES_DATIN. 1 = AES_DMA engine Enabled. The AES engine operates in DMA mode, and data movement from/to the engine is done by DMA logic.
[6]	**DMACSCAD**	**AES Engine DMA with Cascade Mode** 0 = DMA cascade function Disabled. 1 = In DMA cascade mode, software can update DMA source address register, destination address register, and byte count register during a cascade operation, without finishing the accelerator operation.
[5]	**DMALAST**	**AES Last Block** In DMA mode, this bit must be set as beginning the last DMA cascade round. In Non-DMA mode, this bit must be set when feeding in the last block of data in ECB, CBC, CTR, OFB, and CFB mode, and feeding in the (last-1) block of data at CBC-CS1, CBC-CS2, and CBC-CS3 mode. This bit is always 0 when it's read back. Must be written again once START is triggered.
[4]	**EXTKEY**	**External Key** 0 = AES key is from CRPT_AESn_KEYx. 1 = AES uses external key. If the AES accelerator is operating and the corresponding flag BUSY is 1, updating this register has no effect.
[3:2]	**KEYSZ**	**AES Key Size** This bit defines three different key size for AES operation. 2'b00 = 128 bits key. 2'b01 = 192 bits key. 2'b10 = 256 bits key. 2'b11 = Reserved. If the AES accelerator is operating and the corresponding flag BUSY is 1, updating this register has no effect.
[1]	**STOP**	**AES Engine Stop** 0 = No effect. 1 = Stop AES engine. **Note:** This bit is always 0 when it's read back.

[0]	**START**	**AES Engine Start** 0 = No effect. 1 = Start AES engine. BUSY flag will be set. **Note:** This bit is always 0 when it’s read back.

AES Status Flag Register (CRPT_AES_STS)

Register	Offset	R/W	Description	Reset Value
CRPT_AES_STS	CRYP_BA+0x104	R	AES Engine Flag	0x0001_0100

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved			**BUSERR**	**Reserved**	**OUTBUFERR**	**OUTBUFFULL**	**OUTBUFEMPTY**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			**CNTERR**	**Reserved**	**INBUFERR**	**INBUFFULL**	**INBUFEMPTY**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**BUSY**

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20]	**BUSERR**	**AES DMA Access Bus Error Flag** 0 = No error. 1 = Bus error will stop DMA operation and AES engine.
[19]	**Reserved**	Reserved.
[18]	**OUTBUFERR**	**AES Out Buffer Error Flag** 0 = No error. 1 = Error happens during getting the result from AES engine.
[17]	**OUTBUFFULL**	**AES Out Buffer Full Flag** 0 = AES output buffer is not full. 1 = AES output buffer is full, and software needs to get data from CRPT_AES_DATOUT. Otherwise, the AES engine will be pending since the output buffer is full.
[16]	**OUTBUFEMPTY**	**AES Out Buffer Empty** 0 = AES output buffer is not empty. There are some valid data kept in output buffer. 1 = AES output buffer is empty. Software cannot get data from CRPT_AES_DATOUT. Otherwise, the flag OUTBUFERR will be set to 1 since the output buffer is empty.
[15:13]	**Reserved**	Reserved.
[12]	**CNTERR**	**CRPT_AESn_CNT Setting Error** 0 = No error in CRPT_AESn_CNT setting. 1 = CRPT_AESn_CNT is not a multiply of 16 in ECB, CBC, CFB, OFB, and CTR mode.
[11]	**Reserved**	Reserved.

[10]	**INBUFERR**	**AES Input Buffer Error Flag** 0 = No error. 1 = Error happens during feeding data to the AES engine.
[9]	**INBUFFULL**	**AES Input Buffer Full Flag** 0 = AES input buffer is not full. Software can feed the data into the AES engine. 1 = AES input buffer is full. Software cannot feed data to the AES engine. Otherwise, the flag INBUFERR will be set to 1.
[8]	**INBUFEMPTY**	**AES Input Buffer Empty** 0 = There are some data in input buffer waiting for the AES engine to process. 1 = AES input buffer is empty. Software needs to feed data to the AES engine. Otherwise, the AES engine will be pending to wait for input data.
[7:1]	**Reserved**	Reserved.
[0]	**BUSY**	**AES Engine Busy** 0 = The AES engine is idle or finished. 1 = The AES engine is under processing.

AES Data Input Port Register (CRPT_AES_DATIN)

Register	Offset	R/W	Description	Reset Value
CRPT_AES_DATIN	CRYP_BA+0x108	R/W	AES Engine Data Input Port Register	0x0000_0000

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

Bits	Description	
[31:0]	**DATIN**	**AES Engine Input Port** CPU feeds data to AES engine through this port by checking CRPT_AES_STS. Feed data as INBUFFULL is 0.

NUC980 SERIES TECHNICAL REFERENCE MANUAL

AES Data Output Port Register (CRPT_AES_DATOUT)

Register	Offset	R/W	Description	Reset Value
CRPT_AES_DATOUT	CRYP_BA+0x10C	R	AES Engine Data Output Port Register	0x0000_0000

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

Bits	Description	
[31:0]	**DATOUT**	**AES Engine Output Port** CPU gets results from the AES engine through this port by checking CRPT_AES_STS. Get data as OUTBUFEMPTY is 0.

AES Key Word x Register (CRPT_AES0_KEYx)

Register	Offset	R/W	Description	Reset Value
CRPT_AES0_KEY0	CRYP_BA+0x110	R/W	AES Key Word 0 Register for Channel 0	0x0000_0000
CRPT_AES0_KEY1	CRYP_BA+0x114	R/W	AES Key Word 1 Register for Channel 0	0x0000_0000
CRPT_AES0_KEY2	CRYP_BA+0x118	R/W	AES Key Word 2 Register for Channel 0	0x0000_0000
CRPT_AES0_KEY3	CRYP_BA+0x11C	R/W	AES Key Word 3 Register for Channel 0	0x0000_0000
CRPT_AES0_KEY4	CRYP_BA+0x120	R/W	AES Key Word 4 Register for Channel 0	0x0000_0000
CRPT_AES0_KEY5	CRYP_BA+0x124	R/W	AES Key Word 5 Register for Channel 0	0x0000_0000
CRPT_AES0_KEY6	CRYP_BA+0x128	R/W	AES Key Word 6 Register for Channel 0	0x0000_0000
CRPT_AES0_KEY7	CRYP_BA+0x12C	R/W	AES Key Word 7 Register for Channel 0	0x0000_0000

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

Bits	Description	
[31:0]	**KEY**	**CRPT_AES0_KEYx** The KEY keeps the security key for AES operation. n = 0. x = 0, 1..7. The security key for AES accelerator can be 128, 192, or 256 bits and four, six, or eight 32-bit registers are to store each security key. {CRPT_AESn_KEY3, CRPT_AESn_KEY2, CRPT_AESn_KEY1, CRPT_AESn_KEY0} stores the 128-bit security key for AES operation. {CRPT_AESn_KEY5, CRPT_AESn_KEY4, CRPT_AESn_KEY3, CRPT_AESn_KEY2, CRPT_AESn_KEY1, CRPT_AESn_KEY0} stores the 192-bit security key for AES operation. {CRPT_AESn_KEY7, CRPT_AESn_KEY6, CRPT_AESn_KEY5, CRPT_AESn_KEY4, CRPT_AESn_KEY3, CRPT_AESn_KEY2, CRPT_AESn_KEY1, CRPT_AESn_KEY0} stores the 256-bit security key for AES operation.

AES Initial Vector Word x Register (CRPT_AES0_IVx)

Register	Offset	R/W	Description	Reset Value
CRPT_AES0_IV0	CRYP_BA+0x130	R/W	AES Initial Vector Word 0 Register for Channel 0	0x0000_0000
CRPT_AES0_IV1	CRYP_BA+0x134	R/W	AES Initial Vector Word 1 Register for Channel 0	0x0000_0000
CRPT_AES0_IV2	CRYP_BA+0x138	R/W	AES Initial Vector Word 2 Register for Channel 0	0x0000_0000
CRPT_AES0_IV3	CRYP_BA+0x13C	R/W	AES Initial Vector Word 3 Register for Channel 0	0x0000_0000

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

Bits	Description	

[31:0]	**IV**	**AES Initial Vectors**
		n = 0.
		x = 0, 1..3.
		Four initial vectors (CRPT_AESn_IV0, CRPT_AESn_IV1, CRPT_AESn_IV2, and CRPT_AESn_IV3) are for AES operating in CBC, CFB, and OFB mode. Four registers (CRPT_AESn_IV0, CRPT_AESn_IV1, CRPT_AESn_IV2, and CRPT_AESn_IV3) act as Nonce counter when the AES engine is operating in CTR mode.

AES DMA Source Address Register (CRPT_AES0_SADDR)

Register	Offset	R/W	Description	Reset Value
CRPT_AES0_SADDR	CRYP_BA+0x140	R/W	AES DMA Source Address Register for Channel 0	0x0000_0000

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

Bits	Description	
[31:0]	**SADDR**	**AES DMA Source Address** The AES accelerator supports DMA function to transfer the plain text between system memory and embedded FIFO. The SADDR keeps the source address of the data buffer where the source text is stored. Based on the source address, the AES accelerator can read the plain text from system memory and do AES operation. The start of source address should be located at word boundary. In other words, bit 1 and 0 of SADDR are ignored. SADDR can be read and written. Writing to SADDR while the AES accelerator is operating doesn't affect the current AES operation. But the value of SADDR will be updated later on. Consequently, software can prepare the DMA source address for the next AES operation. In DMA mode, software can update the next CRPT_AESn_SADDR before triggering START. The value of CRPT_AESn_SADDR and CRPT_AESn_DADDR can be the same.

AES DMA Destination Address Register (CRPT_AES0_DADDR)

Register	Offset	R/W	Description	Reset Value
CRPT_AES0_DADDR	CRYP_BA+0x144	R/W	AES DMA Destination Address Register for Channel 0	0x0000_0000

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

Bits	Description	
[31:0]	**DADDR**	**AES DMA Destination Address** The AES accelerator supports DMA function to transfer the cipher text between system memory and embedded FIFO. The DADDR keeps the destination address of the data buffer where the engine output's text will be stored. Based on the destination address, the AES accelerator can write the cipher text back to system memory after the AES operation is finished. The start of destination address should be located at word boundary. In other words, bit 1 and 0 of DADDR are ignored. DADDR can be read and written. Writing to DADDR while the AES accelerator is operating doesn't affect the current AES operation. But the value of DADDR will be updated later on. Consequently, software can prepare the destination address for the next AES operation. In DMA mode, software can update the next CRPT_AESn_DADDR before triggering START. The value of CRPT_AESn_SADDR and CRPT_AESn_DADDR can be the same.

AES Byte Count Register (CRPT_AES0_CNT)

Register	Offset	R/W	Description	Reset Value
CRPT_AES0_CNT	CRYP_BA+0x148	R/W	AES Byte Count Register for Channel 0	0x0000_0000

31	30	29	28	27	26	25	24
CNT							
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	
[31:0]	**CNT**	**AES Byte Count** The CRPT_AESn_CNT keeps the byte count of source text that is for the AES engine operating in DMA mode. The CRPT_AESn_CNT is 32-bit and the maximum of byte count is 4G bytes. CRPT_AESn_CNT can be read and written. Writing to CRPT_AESn_CNT while the AES accelerator is operating doesn't affect the current AES operation. But the value of CRPT_AESn_CNT will be updated later on. Consequently, software can prepare the byte count of data for the next AES operation. According to CBC-CS1, CBC-CS2, and CBC-CS3 standard, the count of operation data must be at least one block. Operations that are less than one block will output unexpected result. In Non-DMA ECB, CBC, CFB, OFB, and CTR mode, CRPT_AESn_CNT must be set as byte count for the last block of data before feeding in the last block of data. In Non-DMA CBC-CS1, CBC-CS2, and CBC-CS3 mode, CRPT_AESn_CNT must be set as byte count for the last two blocks of data before feeding in the last two blocks of data.

AES Feedback x Register (CRPT_AES_FDBCKx)

Register	Offset	R/W	Description	Reset Value
CRPT_AES_FDBCK0	CRYP_BA+0x050	R	AES Engine Output Feedback Data After Cryptographic Operation	0x0000_0000
CRPT_AES_FDBCK1	CRYP_BA+0x054	R	AES Engine Output Feedback Data After Cryptographic Operation	0x0000_0000
CRPT_AES_FDBCK2	CRYP_BA+0x058	R	AES Engine Output Feedback Data After Cryptographic Operation	0x0000_0000
CRPT_AES_FDBCK3	CRYP_BA+0x05C	R	AES Engine Output Feedback Data After Cryptographic Operation	0x0000_0000

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

Bits	Description	
[31:0]	**FDBCK**	**AES Feedback Information** The feedback value is 128 bits in size. The AES engine uses the data from CRPT_AES_FDBCKx as the data inputted to CRPT_AESn_IVx for the next block in DMA cascade mode. The AES engine outputs feedback information for IV in the next block's operation. Software can use this feedback information to implement more than one DMA channels. Software can store that feedback value temporarily. After switching back, fill the stored feedback value to this register in the same channel operation, and then continue the operation with the original setting.

6.26.7.5 SHA/HMAC Register

SHA/HMAC Control Register (CRPT_HMAC_CTL)

Register	Offset	R/W	Description	Reset Value
CRPT_HMAC_CTL	CRYP_BA+0x300	R/W	SHA/HMAC Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
INSWAP	OUTSWAP	Reserved					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CMPEN	Reserved				OPMODE		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DMAEN	Reserved	DMALAST	HMACEN	Reserved		STOP	START

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23]	**INSWAP**	**SHA/HMAC Engine Input Data Swap** 0 = Keep the original order. 1 = The order that CPU feeds data to the accelerator will be changed from {byte3, byte2, byte1, byte0} to {byte0, byte1, byte2, byte3}.
[22]	**OUTSWAP**	**SHA/HMAC Engine Output Data Swap** 0 = Keep the original order. 1 = The order that CPU feeds data to the accelerator will be changed from {byte3, byte2, byte1, byte0} to {byte0, byte1, byte2, byte3}.
[21:16]	**Reserved**	Reserved.
[15]	**CMPEN**	**Compare SHA/HMAC Output Digest with MTP Key** 0 = Don't compare with MTP key. 1 = The SHA/HMAC output digest would be compared with MTP key. Since MTP is 256 bits in size, HMAC digest comparing is not supported in HMAC-SHA-384 and HMAC-SHA-512.
[14:11]	**Reserved**	Reserved.
[10:8]	**OPMODE**	**SHA/HMAC Engine Operation Modes** 0x0xx: SHA160 0x100: SHA256 0x101: SHA224 0x110: SHA512 0x111: SHA384 These bits can be read and written. But writing to them wouldn't take effect as BUSY is 1.

[7]	**DMAEN**	**SHA/HMAC Engine DMA Enable Control** 0 = SHA/HMAC DMA engine Disabled. SHA/HMAC engine operates in Non-DMA mode, and gets data from the port CRPT_HMAC_DATIN. 1 = SHA/HMAC DMA engine Enabled. SHA/HMAC engine operates in DMA mode, and data movement from/to the engine is done by DMA logic.
[6]	**Reserved**	Reserved.
[5]	**DMALAST**	**SHA/HMAC Last Block** In DMA mode, this bit must be set as beginning the last DMA cascade round. In Non-DMA mode, this bit must be set as feeding in last byte of data.
[4]	**HMACEN**	**HMAC_SHA Engine Operating Mode** 0 = execute SHA function. 1 = execute HMAC function.
[3:2]	**Reserved**	Reserved.
[1]	**STOP**	**SHA/HMAC Engine Stop** 0 = No effect. 1 = Stop SHA/HMAC engine. This bit is always 0 when it’s read back.
[0]	**START**	**SHA/HMAC Engine Start** 0 = No effect. 1 = Start SHA/HMAC engine. BUSY flag will be set. This bit is always 0 when it’s read back.

SHA/HMAC Status Register (CRPT_HMAC_STS)

Register	Offset	R/W	Description	Reset Value
CRPT_HMAC_STS	CRYP_BA+0x304	R	SHA/HMAC Status Flag	0x0000_0000

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

Bits	Description	
[31:16]	**Reserved**	Reserved.
[16]	**DATINREQ**	**SHA/HMAC Non-dMA Mode Data Input Request** 0 = No effect. 1 = Request SHA/HMAC Non-DMA mode data input.
[15]	**CMPSTS**	**SHA/HMAC Output Digest Compare Result with MTP Key** 0 = SHA/HMAC output digest doesn’t match MTP key. 1 = SHA/HMAC output digest matches MTP key.
[14:9]	**Reserved**	Reserved.
[8]	**DMAERR**	**SHA/HMAC Engine DMA Error Flag** 0 = Show the SHA/HMAC engine access normal. 1 = Show the SHA/HMAC engine access error.
[7:2]	**Reserved**	Reserved.
[1]	**DMABUSY**	**SHA/HMAC Engine DMA Busy Flag** 0 = SHA/HMAC DMA engine is idle or finished. 1 = SHA/HMAC DMA engine is busy.
[0]	**BUSY**	**SHA/HMAC Engine Busy** 0 = SHA/HMAC engine is idle or finished. 1 = SHA/HMAC engine is busy.

SHA/HMAC Outputs Digest Word Register (CRPT_HMAC_DGSTx)

Register	Offset	R/W	Description	Reset Value
CRPT_HMAC_DGST0	CRYP_BA+0x308	R	SHA/HMAC Digest Message 0	0x0000_0000
CRPT_HMAC_DGST1	CRYP_BA+0x30C	R	SHA/HMAC Digest Message 1	0x0000_0000
CRPT_HMAC_DGST2	CRYP_BA+0x310	R	SHA/HMAC Digest Message 2	0x0000_0000
CRPT_HMAC_DGST3	CRYP_BA+0x314	R	SHA/HMAC Digest Message 3	0x0000_0000
CRPT_HMAC_DGST4	CRYP_BA+0x318	R	SHA/HMAC Digest Message 4	0x0000_0000
CRPT_HMAC_DGST5	CRYP_BA+0x31C	R	SHA/HMAC Digest Message 5	0x0000_0000
CRPT_HMAC_DGST6	CRYP_BA+0x320	R	SHA/HMAC Digest Message 6	0x0000_0000
CRPT_HMAC_DGST7	CRYP_BA+0x324	R	SHA/HMAC Digest Message 7	0x0000_0000
CRPT_HMAC_DGST8	CRYP_BA+0x328	R	SHA/HMAC Digest Message 8	0x0000_0000
CRPT_HMAC_DGST9	CRYP_BA+0x32C	R	SHA/HMAC Digest Message 9	0x0000_0000
CRPT_HMAC_DGST10	CRYP_BA+0x330	R	SHA/HMAC Digest Message 10	0x0000_0000
CRPT_HMAC_DGST11	CRYP_BA+0x334	R	SHA/HMAC Digest Message 11	0x0000_0000
CRPT_HMAC_DGST12	CRYP_BA+0x338	R	SHA/HMAC Digest Message 12	0x0000_0000
CRPT_HMAC_DGST13	CRYP_BA+0x33C	R	SHA/HMAC Digest Message 13	0x0000_0000
CRPT_HMAC_DGST14	CRYP_BA+0x340	R	SHA/HMAC Digest Message 14	0x0000_0000
CRPT_HMAC_DGST15	CRYP_BA+0x344	R	SHA/HMAC Digest Message 15	0x0000_0000

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

DGST

Bits	Description	
[31:0]	**DGST**	**SHA/HMAC Digest Message Output Register** For SHA-160, the digest is stored in CRPT_HMAC_DGST0 ~ CRPT_HMAC_DGST4. For SHA-224, the digest is stored in CRPT_HMAC_DGST0 ~ CRPT_HMAC_DGST6. For SHA-256, the digest is stored in CRPT_HMAC_DGST0 ~ CRPT_HMAC_DGST7. For SHA-384, the digest is stored in CRPT_HMAC_DGST0 ~ CRPT_HMAC_DGST11. For SHA-512, the digest is stored in CRPT_HMAC_DGST0 ~ CRPT_HMAC_DGST15.

SHA/HMAC Key Byte Count Register (CRPT_HMAC_KEYCNT)

Register	Offset	R/W	Description	Reset Value
CRPT_HMAC_KEYCNT	CRYP_BA+0x348	R/W	SHA/HMAC Key Byte Count Register	0x0000_0000

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

Bits	Description	
[31:0]	**KEYCNT**	**SHA/HMAC Key Byte Count** The CRPT_HMAC_KEYCNT keeps the byte count of key that SHA/HMAC engine operates. The register is 32-bit and the maximum byte count is 4G bytes. It can be read and written. Writing to the register CRPT_HMAC_KEYCNT as the SHA/HMAC accelerator operating doesn’t affect the current SHA/HMAC operation. But the value of CRPT_SHA _KEYCNT will be updated later on. Consequently, software can prepare the key count for the next SHA/HMAC operation.

SHA/HMAC DMA Source Address Register (CRPT_HMAC_SADDR)

Register	Offset	R/W	Description	Reset Value
CRPT_HMAC_SADDR	CRYP_BA+0x34C	R/W	SHA/HMAC DMA Source Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**SADDR**	**SHA/HMAC DMA Source Address** The SHA/HMAC accelerator supports DMA function to transfer the plain text between system memory and embedded FIFO. The CRPT_HMAC_SADDR keeps the source address of the data buffer where the source text is stored. Based on the source address, the SHA/HMAC accelerator can read the plain text from system memory and do SHA/HMAC operation. The start of source address should be located at word boundary. In other words, bit 1 and 0 of CRPT_HMAC_SADDR are ignored. CRPT_HMAC_SADDR can be read and written. Writing to CRPT_HMAC_SADDR while the SHA/HMAC accelerator is operating doesn’t affect the current SHA/HMAC operation. But the value of CRPT_HMAC_SADDR will be updated later on. Consequently, software can prepare the DMA source address for the next SHA/HMAC operation. In DMA mode, software can update the next CRPT_HMAC_SADDR before triggering START. CRPT_HMAC_SADDR and CRPT_HMAC_DADDR can be the same in the value.

SHA/HMAC Byte Count Register (CRPT_HMAC_DMACNT)

Register	Offset	R/W	Description	Reset Value
CRPT_HMAC_DMACNT	CRYP_BA+0x350	R/W	SHA/HMAC Byte Count Register	0x0000_0000

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

Bits	Description	
[31:0]	**DMACNT**	**SHA/HMAC Operation Byte Count** The CRPT_HMAC_DMACNT keeps the byte count of source text that is for the SHA/HMAC engine operating in DMA mode. The CRPT_HMAC_DMACNT is 32-bit and the maximum of byte count is 4G bytes. CRPT_HMAC_DMACNT can be read and written. Writing to CRPT_HMAC_DMACNT while the SHA/HMAC accelerator is operating doesn’t affect the current SHA/HMAC operation. But the value of CRPT_HMAC_DMACNT will be updated later on. Consequently, software can prepare the byte count of data for the next SHA/HMAC operation. In Non-DMA mode, CRPT_HMAC_DMACNT must be set as the byte count of the last block before feeding in the last block of data.

SHA/HMAC Data Input Port Register (CRPT_HMAC_DATIN)

Register	Offset	R/W	Description	Reset Value
CRPT_HMAC_DATIN	CRYP_BA+0x354	R/W	SHA/HMAC Engine Non-dMA Mode Data Input Port Register	0x0000_0000

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

Bits	Description	
[31:0]	**DATIN**	**SHA/HMAC Engine Input Port** CPU feeds data to SHA/HMAC engine through this port by checking CRPT_HMAC_STS. Feed data as DATINREQ is 1.

6.26.7.6 ECC Register

ECC Control Register (CRPT_ECC_CTL)

Register	Offset	R/W	Description	Reset Value
CRPT_ECC_CTL	CRYP_BA+0x800	R/W	ECC Control Register	0x0000_0000

31	30	29	28	27	26	25	24
CURVEM							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CURVEM		**LDK**	**LDN**	**LDB**	**LDA**	**LDP2**	**LDP1**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			**MODOP**		**ECCOP**		**FSEL**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DMAEN	**Reserved**					**STOP**	**START**

Bits	Description	
[31:22]	**CURVEM**	The key length of elliptic curve.
[21]	**LDK**	**The Control Signal of Register for SCALARK** 0 = The register for SCALARK is not modified by DMA or user. 1 = The register for SCALARK is modified by DMA or user.
[20]	**LDN**	**The Control Signal of Register for the Parameter CURVEN of Elliptic Curve** 0 = The register for CURVEN is not modified by DMA or user. 1 = The register for CURVEN is modified by DMA or user.
[19]	**LDB**	**The Control Signal of Register for the Parameter CURVEB of Elliptic Curve** 0 = The register for CURVEB is not modified by DMA or user. 1 = The register for CURVEB is modified by DMA or user.
[18]	**LDA**	**The Control Signal of Register for the Parameter CURVEA of Elliptic Curve** 0 = The register for CURVEA is not modified by DMA or user. 1 = The register for CURVEA is modified by DMA or user.
[17]	**LDP2**	**The Control Signal of Register for the x and Y Coordinate of the Second Point (POINTX2, POINTY2)** 0 = The register for POINTX2 and POINTY2 is not modified by DMA or user. 1 = The register for POINTX2 and POINTY2 is modified by DMA or user.
[16]	**LDP1**	**The Control Signal of Register for the x and Y Coordinate of the First Point (POINTX1, POINTY1)** 0 = The register for POINTX1 and POINTY1 is not modified by DMA or user. 1 = The register for POINTX1 and POINTY1 is modified by DMA or user.

[15:13]	**Reserved**	Reserved.
[12:11]	**MODOP**	**Modulus Operation for PF** 00 = Division :. POINTX1 = (POINTY1 / POINTX1) % CURVEN. 01 = Multiplication :. POINTX1 = (POINTX1 * POINTY1) % CURVEN. 10 = Addition :. POINTX1 = (POINTX1 + POINTY1) % CURVEN. 11 = Subtraction :. POINTX1 = (POINTX1 - POINTY1) % CURVEN. MODOP is active only when ECCOP = 01.
[10:9]	**ECCOP**	**Point Operation for BF and PF** 00 = Point multiplication :. (POINTX1, POINTY1) = SCALARK * (POINTX1, POINTY1). 01 = Modulus operation : choose by MODOP (CRPT_ECC_CTL[12:11]). 10 = Point addition :. (POINTX1, POINTY1) = (POINTX1, POINTY1) +. (POINTX2, POINTY2) 11 = Point doubling :. (POINTX1, POINTY1) = 2 * (POINTX1, POINTY1). Besides above three input data, point operations still need the parameters of elliptic curve (CURVEA, CURVEB, CURVEN and CURVEM) as shown in Figure 6.27-11.
[8]	**FSEL**	**Field Selection** 0 = Binary Field (GF(2^m)). 1 = Prime Field (GF(p)).
[7]	**DMAEN**	**ECC Accelerator DMA Enable Bit** 0 = ECC DMA engine Disabled. 1 = ECC DMA engine Enabled. Only when START and DMAEN are 1, ECC DMA engine will be active
[6:2]	**Reserved**	Reserved.
[1]	**STOP**	**ECC Accelerator Stop** 0 = No effect. 1 = Abort ECC accelerator and make it into idle state. This bit is always 0 when it’s read back. Remember to clear ECC interrupt flag after stopping ECC accelerator.
[0]	**START**	**ECC Accelerator Start** 0 = No effect. 1 = Start ECC accelerator. BUSY flag will be set. This bit is always 0 when it’s read back. ECC accelerator will ignore this START signal when BUSY flag is 1.

ECC Status Register (CRPT_ECC_STS)

Register	Offset	R/W	Description	Reset Value
CRPT_ECC_STS	CRYP_BA+0x804	R	ECC Status Register	0x0000_0000

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

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**BUSERR**	**ECC DMA Access Bus Error Flag** 0 = No error. 1 = Bus error will stop DMA operation and ECC accelerator.
[15:2]	**Reserved**	Reserved.
[1]	**DMABUSY**	**ECC DMA Busy Flag** 0 = ECC DMA is idle or finished. 1 = ECC DMA is busy.
[0]	**BUSY**	**ECC Accelerator Busy Flag** 0 = The ECC accelerator is idle or finished. 1 = The ECC accelerator is under processing and protects all registers. Remember to clear ECC interrupt flag after ECC accelerator finished

ECC the X-coordinate Value of the First Point Register (CRPT_ECC_X1)

Register	Offset	R/W	Description	Reset Value
CRPT_ECC_X1_00	CRYP_BA+0x808	R/W	ECC the X-coordinate Word0 of the First Point	0x0000_0000
CRPT_ECC_X1_01	CRYP_BA+0x80C	R/W	ECC the X-coordinate Word1 of the First Point	0x0000_0000
CRPT_ECC_X1_02	CRYP_BA+0x810	R/W	ECC the X-coordinate Word2 of the First Point	0x0000_0000
CRPT_ECC_X1_03	CRYP_BA+0x814	R/W	ECC the X-coordinate Word3 of the First Point	0x0000_0000
CRPT_ECC_X1_04	CRYP_BA+0x818	R/W	ECC the X-coordinate Word4 of the First Point	0x0000_0000
CRPT_ECC_X1_05	CRYP_BA+0x81C	R/W	ECC the X-coordinate Word5 of the First Point	0x0000_0000
CRPT_ECC_X1_06	CRYP_BA+0x820	R/W	ECC the X-coordinate Word6 of the First Point	0x0000_0000
CRPT_ECC_X1_07	CRYP_BA+0x824	R/W	ECC the X-coordinate Word7 of the First Point	0x0000_0000
CRPT_ECC_X1_08	CRYP_BA+0x828	R/W	ECC the X-coordinate Word8 of the First Point	0x0000_0000
CRPT_ECC_X1_09	CRYP_BA+0x82C	R/W	ECC the X-coordinate Word9 of the First Point	0x0000_0000
CRPT_ECC_X1_10	CRYP_BA+0x830	R/W	ECC the X-coordinate Word10 of the First Point	0x0000_0000
CRPT_ECC_X1_11	CRYP_BA+0x834	R/W	ECC the X-coordinate Word11 of the First Point	0x0000_0000
CRPT_ECC_X1_12	CRYP_BA+0x838	R/W	ECC the X-coordinate Word12 of the First Point	0x0000_0000
CRPT_ECC_X1_13	CRYP_BA+0x83C	R/W	ECC the X-coordinate Word13 of the First Point	0x0000_0000
CRPT_ECC_X1_14	CRYP_BA+0x840	R/W	ECC the X-coordinate Word14 of the First Point	0x0000_0000
CRPT_ECC_X1_15	CRYP_BA+0x844	R/W	ECC the X-coordinate Word15 of the First Point	0x0000_0000
CRPT_ECC_X1_16	CRYP_BA+0x848	R/W	ECC the X-coordinate Word16 of the First Point	0x0000_0000
CRPT_ECC_X1_17	CRYP_BA+0x84C	R/W	ECC the X-coordinate Word17 of the First Point	0x0000_0000

31	30	29	28	27	26	25	24
POINTX1							
23	22	21	20	19	18	17	16
POINTX1							
15	14	13	12	11	10	9	8
POINTX1							
7	6	5	4	3	2	1	0
POINTX1							

Bits	Description

[31:0]	**POINTX1**	**ECC the X-coordinate Value of the First Point (POINTX1)** For B-163 or K-163, POINTX1 is stored in CRPT_ECC_X1_00~CRPT_ECC_X1_05 For B-233 or K-233, POINTX1 is stored in CRPT_ECC_X1_00~CRPT_ECC_X1_07 For B-283 or K-283, POINTX1 is stored in CRPT_ECC_X1_00~CRPT_ECC_X1_08 For B-409 or K-409, POINTX1 is stored in CRPT_ECC_X1_00~CRPT_ECC_X1_12 For B-571 or K-571, POINTX1 is stored in CRPT_ECC_X1_00~CRPT_ECC_X1_17 For P-192, POINTX1 is stored in CRPT_ECC_X1_00~CRPT_ECC_X1_05 For P-224, POINTX1 is stored in CRPT_ECC_X1_00~CRPT_ECC_X1_06 For P-256, POINTX1 is stored in CRPT_ECC_X1_00~CRPT_ECC_X1_07 For P-384, POINTX1 is stored in CRPT_ECC_X1_00~CRPT_ECC_X1_11 For P-521, POINTX1 is stored in CRPT_ECC_X1_00~CRPT_ECC_X1_16

ECC the Y-coordinate Value of the First Point Register (CRPT_ECC_Y1)

Register	Offset	R/W	Description	Reset Value
CRPT_ECC_Y1_00	CRYP_BA+0x850	R/W	ECC the Y-coordinate Word0 of the First Point	0x0000_0000
CRPT_ECC_Y1_01	CRYP_BA+0x854	R/W	ECC the Y-coordinate Word1 of the First Point	0x0000_0000
CRPT_ECC_Y1_02	CRYP_BA+0x858	R/W	ECC the Y-coordinate Word2 of the First Point	0x0000_0000
CRPT_ECC_Y1_03	CRYP_BA+0x85C	R/W	ECC the Y-coordinate Word3 of the First Point	0x0000_0000
CRPT_ECC_Y1_04	CRYP_BA+0x860	R/W	ECC the Y-coordinate Word4 of the First Point	0x0000_0000
CRPT_ECC_Y1_05	CRYP_BA+0x864	R/W	ECC the Y-coordinate Word5 of the First Point	0x0000_0000
CRPT_ECC_Y1_06	CRYP_BA+0x868	R/W	ECC the Y-coordinate Word6 of the First Point	0x0000_0000
CRPT_ECC_Y1_07	CRYP_BA+0x86C	R/W	ECC the Y-coordinate Word7 of the First Point	0x0000_0000
CRPT_ECC_Y1_08	CRYP_BA+0x870	R/W	ECC the Y-coordinate Word8 of the First Point	0x0000_0000
CRPT_ECC_Y1_09	CRYP_BA+0x874	R/W	ECC the Y-coordinate Word9 of the First Point	0x0000_0000
CRPT_ECC_Y1_10	CRYP_BA+0x878	R/W	ECC the Y-coordinate Word10 of the First Point	0x0000_0000
CRPT_ECC_Y1_11	CRYP_BA+0x87C	R/W	ECC the Y-coordinate Word11 of the First Point	0x0000_0000
CRPT_ECC_Y1_12	CRYP_BA+0x880	R/W	ECC the Y-coordinate Word12 of the First Point	0x0000_0000
CRPT_ECC_Y1_13	CRYP_BA+0x884	R/W	ECC the Y-coordinate Word13 of the First Point	0x0000_0000
CRPT_ECC_Y1_14	CRYP_BA+0x888	R/W	ECC the Y-coordinate Word14 of the First Point	0x0000_0000
CRPT_ECC_Y1_15	CRYP_BA+0x88C	R/W	ECC the Y-coordinate Word15 of the First Point	0x0000_0000
CRPT_ECC_Y1_16	CRYP_BA+0x890	R/W	ECC the Y-coordinate Word16 of the First Point	0x0000_0000
CRPT_ECC_Y1_17	CRYP_BA+0x894	R/W	ECC the Y-coordinate Word17 of the First Point	0x0000_0000

31	30	29	28	27	26	25	24
POINTY1							
23	22	21	20	19	18	17	16
POINTY1							
15	14	13	12	11	10	9	8
POINTY1							
7	6	5	4	3	2	1	0
POINTY1							

Bits	Description

[31:0]	**POINTY1**	**ECC the Y-coordinate Value of the First Point (POINTY1)** For B-163 or K-163, POINTY1 is stored in CRPT_ECC_Y1_00~CRPT_ECC_Y1_05 For B-233 or K-233, POINTY1 is stored in CRPT_ECC_Y1_00~CRPT_ECC_Y1_07 For B-283 or K-283, POINTY1 is stored in CRPT_ECC_Y1_00~CRPT_ECC_Y1_08 For B-409 or K-409, POINTY1 is stored in CRPT_ECC_Y1_00~CRPT_ECC_Y1_12 For B-571 or K-571, POINTY1 is stored in CRPT_ECC_Y1_00~CRPT_ECC_Y1_17 For P-192, POINTY1 is stored in CRPT_ECC_Y1_00~CRPT_ECC_Y1_05 For P-224, POINTY1 is stored in CRPT_ECC_Y1_00~CRPT_ECC_Y1_06 For P-256, POINTY1 is stored in CRPT_ECC_Y1_00~CRPT_ECC_Y1_07 For P-384, POINTY1 is stored in CRPT_ECC_Y1_00~CRPT_ECC_Y1_11 For P-521, POINTY1 is stored in CRPT_ECC_Y1_00~CRPT_ECC_Y1_16

ECC the X-coordinate Value of the Second Point Register (CRPT_ECC_X2)

Register	Offset	R/W	Description	Reset Value
CRPT_ECC_X2_00	CRYP_BA+0x898	R/W	ECC the X-coordinate Word0 of the Second Point	0x0000_0000
CRPT_ECC_X2_01	CRYP_BA+0x89C	R/W	ECC the X-coordinate Word1 of the Second Point	0x0000_0000
CRPT_ECC_X2_02	CRYP_BA+0x8A0	R/W	ECC the X-coordinate Word2 of the Second Point	0x0000_0000
CRPT_ECC_X2_03	CRYP_BA+0x8A4	R/W	ECC the X-coordinate Word3 of the Second Point	0x0000_0000
CRPT_ECC_X2_04	CRYP_BA+0x8A8	R/W	ECC the X-coordinate Word4 of the Second Point	0x0000_0000
CRPT_ECC_X2_05	CRYP_BA+0x8AC	R/W	ECC the X-coordinate Word5 of the Second Point	0x0000_0000
CRPT_ECC_X2_06	CRYP_BA+0x8B0	R/W	ECC the X-coordinate Word6 of the Second Point	0x0000_0000
CRPT_ECC_X2_07	CRYP_BA+0x8B4	R/W	ECC the X-coordinate Word7 of the Second Point	0x0000_0000
CRPT_ECC_X2_08	CRYP_BA+0x8B8	R/W	ECC the X-coordinate Word8 of the Second Point	0x0000_0000
CRPT_ECC_X2_09	CRYP_BA+0x8BC	R/W	ECC the X-coordinate Word9 of the Second Point	0x0000_0000
CRPT_ECC_X2_10	CRYP_BA+0x8C0	R/W	ECC the X-coordinate Word10 of the Second Point	0x0000_0000
CRPT_ECC_X2_11	CRYP_BA+0x8C4	R/W	ECC the X-coordinate Word11 of the Second Point	0x0000_0000
CRPT_ECC_X2_12	CRYP_BA+0x8C8	R/W	ECC the X-coordinate Word12 of the Second Point	0x0000_0000
CRPT_ECC_X2_13	CRYP_BA+0x8CC	R/W	ECC the X-coordinate Word13 of the Second Point	0x0000_0000
CRPT_ECC_X2_14	CRYP_BA+0x8D0	R/W	ECC the X-coordinate Word14 of the Second Point	0x0000_0000
CRPT_ECC_X2_15	CRYP_BA+0x8D4	R/W	ECC the X-coordinate Word15 of the Second Point	0x0000_0000
CRPT_ECC_X2_16	CRYP_BA+0x8D8	R/W	ECC the X-coordinate Word16 of the Second Point	0x0000_0000
CRPT_ECC_X2_17	CRYP_BA+0x8DC	R/W	ECC the X-coordinate Word17 of the Second Point	0x0000_0000

<table>
<tr><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th></tr>
<tr><td colspan="8">POINTX2</td></tr>
<tr><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th></tr>
<tr><td colspan="8">POINTX2</td></tr>
<tr><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th></tr>
<tr><td colspan="8">POINTX2</td></tr>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">POINTX2</td></tr>
</table>

Bits	Description

[31:0]	**POINTX2**	**ECC the X-coordinate Value of the Second Point (POINTX2)** For B-163 or K-163, POINTX2 is stored in CRPT_ECC_X2_00~CRPT_ECC_X2_05 For B-233 or K-233, POINTX2 is stored in CRPT_ECC_X2_00~CRPT_ECC_X2_07 For B-283 or K-283, POINTX2 is stored in CRPT_ECC_X2_00~CRPT_ECC_X2_08 For B-409 or K-409, POINTX2 is stored in CRPT_ECC_X2_00~CRPT_ECC_X2_12 For B-571 or K-571, POINTX2 is stored in CRPT_ECC_X2_00~CRPT_ECC_X2_17 For P-192, POINTX2 is stored in CRPT_ECC_X2_00~CRPT_ECC_X2_05 For P-224, POINTX2 is stored in CRPT_ECC_X2_00~CRPT_ECC_X2_06 For P-256, POINTX2 is stored in CRPT_ECC_X2_00~CRPT_ECC_X2_07 For P-384, POINTX2 is stored in CRPT_ECC_X2_00~CRPT_ECC_X2_11 For P-521, POINTX2 is stored in CRPT_ECC_X2_00~CRPT_ECC_X2_16

ECC the Y-coordinate Value of the Second Point Register (CRPT_ECC_Y2)

Register	Offset	R/W	Description	Reset Value
CRPT_ECC_Y2_00	CRYP_BA+0x8E0	R/W	ECC the Y-coordinate Word0 of the Second Point	0x0000_0000
CRPT_ECC_Y2_01	CRYP_BA+0x8E4	R/W	ECC the Y-coordinate Word1 of the Second Point	0x0000_0000
CRPT_ECC_Y2_02	CRYP_BA+0x8E8	R/W	ECC the Y-coordinate Word2 of the Second Point	0x0000_0000
CRPT_ECC_Y2_03	CRYP_BA+0x8EC	R/W	ECC the Y-coordinate Word3 of the Second Point	0x0000_0000
CRPT_ECC_Y2_04	CRYP_BA+0x8F0	R/W	ECC the Y-coordinate Word4 of the Second Point	0x0000_0000
CRPT_ECC_Y2_05	CRYP_BA+0x8F4	R/W	ECC the Y-coordinate Word5 of the Second Point	0x0000_0000
CRPT_ECC_Y2_06	CRYP_BA+0x8F8	R/W	ECC the Y-coordinate Word6 of the Second Point	0x0000_0000
CRPT_ECC_Y2_07	CRYP_BA+0x8FC	R/W	ECC the Y-coordinate Word7 of the Second Point	0x0000_0000
CRPT_ECC_Y2_08	CRYP_BA+0x900	R/W	ECC the Y-coordinate Word8 of the Second Point	0x0000_0000
CRPT_ECC_Y2_09	CRYP_BA+0x904	R/W	ECC the Y-coordinate Word9 of the Second Point	0x0000_0000
CRPT_ECC_Y2_10	CRYP_BA+0x908	R/W	ECC the Y-coordinate Word10 of the Second Point	0x0000_0000
CRPT_ECC_Y2_11	CRYP_BA+0x90C	R/W	ECC the Y-coordinate Word11 of the Second Point	0x0000_0000
CRPT_ECC_Y2_12	CRYP_BA+0x910	R/W	ECC the Y-coordinate Word12 of the Second Point	0x0000_0000
CRPT_ECC_Y2_13	CRYP_BA+0x914	R/W	ECC the Y-coordinate Word13 of the Second Point	0x0000_0000
CRPT_ECC_Y2_14	CRYP_BA+0x918	R/W	ECC the Y-coordinate Word14 of the Second Point	0x0000_0000
CRPT_ECC_Y2_15	CRYP_BA+0x91C	R/W	ECC the Y-coordinate Word15 of the Second Point	0x0000_0000
CRPT_ECC_Y2_16	CRYP_BA+0x920	R/W	ECC the Y-coordinate Word16 of the Second Point	0x0000_0000
CRPT_ECC_Y2_17	CRYP_BA+0x924	R/W	ECC the Y-coordinate Word17 of the Second Point	0x0000_0000

31	30	29	28	27	26	25	24
POINTY2							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
POINTY2							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
POINTY2							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
POINTY2							

Bits	Description

[31:0]	**POINTY2**	**ECC the Y-coordinate Value of the Second Point (POINTY2)** For B-163 or K-163, POINTY2 is stored in CRPT_ECC_Y2_00~CRPT_ECC_Y2_05 For B-233 or K-233, POINTY2 is stored in CRPT_ECC_Y2_00~CRPT_ECC_Y2_07 For B-283 or K-283, POINTY2 is stored in CRPT_ECC_Y2_00~CRPT_ECC_Y2_08 For B-409 or K-409, POINTY2 is stored in CRPT_ECC_Y2_00~CRPT_ECC_Y2_12 For B-571 or K-571, POINTY2 is stored in CRPT_ECC_Y2_00~CRPT_ECC_Y2_17 For P-192, POINTY2 is stored in CRPT_ECC_Y2_00~CRPT_ECC_Y2_05 For P-224, POINTY2 is stored in CRPT_ECC_Y2_00~CRPT_ECC_Y2_06 For P-256, POINTY2 is stored in CRPT_ECC_Y2_00~CRPT_ECC_Y2_07 For P-384, POINTY2 is stored in CRPT_ECC_Y2_00~CRPT_ECC_Y2_11 For P-521, POINTY2 is stored in CRPT_ECC_Y2_00~CRPT_ECC_Y2_16

ECC the Parameter CURVEA Value of Elliptic Curve Register (CRPT_ECC_A)

Register	Offset	R/W	Description	Reset Value
CRPT_ECC_A_00	CRYP_BA+0x928	R/W	ECC the Parameter CURVEA Word0 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_01	CRYP_BA+0x92C	R/W	ECC the Parameter CURVEA Word1 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_02	CRYP_BA+0x930	R/W	ECC the Parameter CURVEA Word2 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_03	CRYP_BA+0x934	R/W	ECC the Parameter CURVEA Word3 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_04	CRYP_BA+0x938	R/W	ECC the Parameter CURVEA Word4 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_05	CRYP_BA+0x93C	R/W	ECC the Parameter CURVEA Word5 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_06	CRYP_BA+0x940	R/W	ECC the Parameter CURVEA Word6 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_07	CRYP_BA+0x944	R/W	ECC the Parameter CURVEA Word7 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_08	CRYP_BA+0x948	R/W	ECC the Parameter CURVEA Word8 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_09	CRYP_BA+0x94C	R/W	ECC the Parameter CURVEA Word9 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_10	CRYP_BA+0x950	R/W	ECC the Parameter CURVEA Word10 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_11	CRYP_BA+0x954	R/W	ECC the Parameter CURVEA Word11 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_12	CRYP_BA+0x958	R/W	ECC the Parameter CURVEA Word12 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_13	CRYP_BA+0x95C	R/W	ECC the Parameter CURVEA Word13 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_14	CRYP_BA+0x960	R/W	ECC the Parameter CURVEA Word14 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_15	CRYP_BA+0x964	R/W	ECC the Parameter CURVEA Word15 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_16	CRYP_BA+0x968	R/W	ECC the Parameter CURVEA Word16 of Elliptic Curve	0x0000_0000
CRPT_ECC_A_17	CRYP_BA+0x96C	R/W	ECC the Parameter CURVEA Word17 of Elliptic Curve	0x0000_0000

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

Bits	Description

[31:0]	**CURVEA**	**ECC the Parameter CURVEA Value of Elliptic Curve (CURVEA)** The formula of elliptic curve is $y^2=x^3+CURVEA*x+CURVEB$ in GF(p) and $y^2+x*y=x^3+CURVEA*x^2+CURVEB$ in GF(2^m). For B-163 or K-163, CURVEA is stored in CRPT_ECC_A_00~CRPT_ECC_A_05 For B-233 or K-233, CURVEA is stored in CRPT_ECC_A_00~CRPT_ECC_A_07 For B-283 or K-283, CURVEA is stored in CRPT_ECC_A_00~CRPT_ECC_A_08 For B-409 or K-409, CURVEA is stored in CRPT_ECC_A_00~CRPT_ECC_A_12 For B-571 or K-571, CURVEA is stored in CRPT_ECC_A_00~CRPT_ECC_A_17 For P-192, CURVEA is stored in CRPT_ECC_A_00~CRPT_ECC_A_05 For P-224, CURVEA is stored in CRPT_ECC_A_00~CRPT_ECC_A_06 For P-256, CURVEA is stored in CRPT_ECC_A_00~CRPT_ECC_A_07 For P-384, CURVEA is stored in CRPT_ECC_A_00~CRPT_ECC_A_11 For P-521, CURVEA is stored in CRPT_ECC_A_00~CRPT_ECC_A_16

ECC the Parameter CURVEB Value of Elliptic Curve Register (CRPT_ECC_B)

Register	Offset	R/W	Description	Reset Value
CRPT_ECC_B_00	CRYP_BA+0x970	R/W	ECC the Parameter CURVEB Word0 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_01	CRYP_BA+0x974	R/W	ECC the Parameter CURVEB Word1 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_02	CRYP_BA+0x978	R/W	ECC the Parameter CURVEB Word2 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_03	CRYP_BA+0x97C	R/W	ECC the Parameter CURVEB Word3 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_04	CRYP_BA+0x980	R/W	ECC the Parameter CURVEB Word4 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_05	CRYP_BA+0x984	R/W	ECC the Parameter CURVEB Word5 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_06	CRYP_BA+0x988	R/W	ECC the Parameter CURVEB Word6 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_07	CRYP_BA+0x98C	R/W	ECC the Parameter CURVEB Word7 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_08	CRYP_BA+0x990	R/W	ECC the Parameter CURVEB Word8 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_09	CRYP_BA+0x994	R/W	ECC the Parameter CURVEB Word9 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_10	CRYP_BA+0x998	R/W	ECC the Parameter CURVEB Word10 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_11	CRYP_BA+0x99C	R/W	ECC the Parameter CURVEB Word11 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_12	CRYP_BA+0x9A0	R/W	ECC the Parameter CURVEB Word12 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_13	CRYP_BA+0x9A4	R/W	ECC the Parameter CURVEB Word13 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_14	CRYP_BA+0x9A8	R/W	ECC the Parameter CURVEB Word14 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_15	CRYP_BA+0x9AC	R/W	ECC the Parameter CURVEB Word15 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_16	CRYP_BA+0x9B0	R/W	ECC the Parameter CURVEB Word16 of Elliptic Curve	0x0000_0000
CRPT_ECC_B_17	CRYP_BA+0x9B4	R/W	ECC the Parameter CURVEB Word17 of Elliptic Curve	0x0000_0000

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

Bits	Description

[31:0]	**CURVEB**	**ECC the Parameter CURVEB Value of Elliptic Curve (CURVEA)** The formula of elliptic curve is $y^2=x^3$+CURVEA*x+CURVEB in GF(p) and $y^2+x*y=x^3$+CURVEA*x^2+CURVEB in GF(2^m). For B-163 or K-163, CURVEB is stored in CRPT_ECC_B_00~CRPT_ECC_B_05 For B-233 or K-233, CURVEB is stored in CRPT_ECC_B_00~CRPT_ECC_B_07 For B-283 or K-283, CURVEB is stored in CRPT_ECC_B_00~CRPT_ECC_B_08 For B-409 or K-409, CURVEB is stored in CRPT_ECC_B_00~CRPT_ECC_B_12 For B-521 or K-521, CURVEB is stored in CRPT_ECC_B_00~CRPT_ECC_B_17 For P-192, CURVEB is stored in CRPT_ECC_B_00~CRPT_ECC_B_05 For P-224, CURVEB is stored in CRPT_ECC_B_00~CRPT_ECC_B_06 For P-256, CURVEB is stored in CRPT_ECC_B_00~CRPT_ECC_B_07 For P-384, CURVEB is stored in CRPT_ECC_B_00~CRPT_ECC_B_11 For P-521, CURVEB is stored in CRPT_ECC_B_00~CRPT_ECC_B_16

ECC the Parameter CURVEN Value of Elliptic Curve Register (CRPT_ECC_N)

Register	Offset	R/W	Description	Reset Value
CRPT_ECC_N_00	CRYP_BA+0x9B8	R/W	ECC the Parameter CURVEN Word0 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_01	CRYP_BA+0x9BC	R/W	ECC the Parameter CURVEN Word1 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_02	CRYP_BA+0x9C0	R/W	ECC the Parameter CURVEN Word2 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_03	CRYP_BA+0x9C4	R/W	ECC the Parameter CURVEN Word3 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_04	CRYP_BA+0x9C8	R/W	ECC the Parameter CURVEN Word4 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_05	CRYP_BA+0x9CC	R/W	ECC the Parameter CURVEN Word5 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_06	CRYP_BA+0x9D0	R/W	ECC the Parameter CURVEN Word6 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_07	CRYP_BA+0x9D4	R/W	ECC the Parameter CURVEN Word7 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_08	CRYP_BA+0x9D8	R/W	ECC the Parameter CURVEN Word8 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_09	CRYP_BA+0x9DC	R/W	ECC the Parameter CURVEN Word9 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_10	CRYP_BA+0x9E0	R/W	ECC the Parameter CURVEN Word10 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_11	CRYP_BA+0x9E4	R/W	ECC the Parameter CURVEN Word11 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_12	CRYP_BA+0x9E8	R/W	ECC the Parameter CURVEN Word12 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_13	CRYP_BA+0x9EC	R/W	ECC the Parameter CURVEN Word13 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_14	CRYP_BA+0x9F0	R/W	ECC the Parameter CURVEN Word14 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_15	CRYP_BA+0x9F4	R/W	ECC the Parameter CURVEN Word15 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_16	CRYP_BA+0x9F8	R/W	ECC the Parameter CURVEN Word16 of Elliptic Curve	0x0000_0000
CRPT_ECC_N_17	CRYP_BA+0x9FC	R/W	ECC the Parameter CURVEN Word17 of Elliptic Curve	0x0000_0000

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

Bits	Description

[31:0]	**CURVEN**	**ECC the Parameter CURVEN Value of Elliptic Curve (CURVEN)** In GF(p), CURVEN is the prime p. In GF(2^m), CURVEN is the irreducible polynomial. For B-163 or K-163, CURVEN is stored in CRPT_ECC_N_00~CRPT_ECC_N_05 For B-233 or K-233, CURVEN is stored in CRPT_ECC_N_00~CRPT_ECC_N_07 For B-283 or K-283, CURVEN is stored in CRPT_ECC_N_00~CRPT_ECC_N_08 For B-409 or K-409, CURVEN is stored in CRPT_ECC_N_00~CRPT_ECC_N_12 For B-571 or K-571, CURVEN is stored in CRPT_ECC_N_00~CRPT_ECC_N_17 For P-192, CURVEN is stored in CRPT_ECC_N_00~CRPT_ECC_N_05 For P-224, CURVEN is stored in CRPT_ECC_N_00~CRPT_ECC_N_06 For P-256, CURVEN is stored in CRPT_ECC_N_00~CRPT_ECC_N_07 For P-384, CURVEN is stored in CRPT_ECC_N_00~CRPT_ECC_N_11 For P-521, CURVEN is stored in CRPT_ECC_N_00~CRPT_ECC_N_16

ECC the Scalar K Value of Elliptic Curve Register (CRPT_ECC_K)

Register	Offset	R/W	Description	Reset Value
CRPT_ECC_K_00	CRYP_BA+0xA00	W	ECC the Scalar SCALARK Word0 of Point Multiplication	0x0000_0000
CRPT_ECC_K_01	CRYP_BA+0xA04	W	ECC the Scalar SCALARK Word1 of Point Multiplication	0x0000_0000
CRPT_ECC_K_02	CRYP_BA+0xA08	W	ECC the Scalar SCALARK Word2 of Point Multiplication	0x0000_0000
CRPT_ECC_K_03	CRYP_BA+0xA0C	W	ECC the Scalar SCALARK Word3 of Point Multiplication	0x0000_0000
CRPT_ECC_K_04	CRYP_BA+0xA10	W	ECC the Scalar SCALARK Word4 of Point Multiplication	0x0000_0000
CRPT_ECC_K_05	CRYP_BA+0xA14	W	ECC the Scalar SCALARK Word5 of Point Multiplication	0x0000_0000
CRPT_ECC_K_06	CRYP_BA+0xA18	W	ECC the Scalar SCALARK Word6 of Point Multiplication	0x0000_0000
CRPT_ECC_K_07	CRYP_BA+0xA1C	W	ECC the Scalar SCALARK Word7 of Point Multiplication	0x0000_0000
CRPT_ECC_K_08	CRYP_BA+0xA20	W	ECC the Scalar SCALARK Word8 of Point Multiplication	0x0000_0000
CRPT_ECC_K_09	CRYP_BA+0xA24	W	ECC the Scalar SCALARK Word9 of Point Multiplication	0x0000_0000
CRPT_ECC_K_10	CRYP_BA+0xA28	W	ECC the Scalar SCALARK Word10 of Point Multiplication	0x0000_0000
CRPT_ECC_K_11	CRYP_BA+0xA2C	W	ECC the Scalar SCALARK Word11 of Point Multiplication	0x0000_0000
CRPT_ECC_K_12	CRYP_BA+0xA30	W	ECC the Scalar SCALARK Word12 of Point Multiplication	0x0000_0000
CRPT_ECC_K_13	CRYP_BA+0xA34	W	ECC the Scalar SCALARK Word13 of Point Multiplication	0x0000_0000
CRPT_ECC_K_14	CRYP_BA+0xA38	W	ECC the Scalar SCALARK Word14 of Point Multiplication	0x0000_0000
CRPT_ECC_K_15	CRYP_BA+0xA3C	W	ECC the Scalar SCALARK Word15 of Point Multiplication	0x0000_0000
CRPT_ECC_K_16	CRYP_BA+0xA40	W	ECC the Scalar SCALARK Word16 of Point Multiplication	0x0000_0000
CRPT_ECC_K_17	CRYP_BA+0xA44	W	ECC the Scalar SCALARK Word17 of Point Multiplication	0x0000_0000

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

Bits	Description

[31:0]	**SCALARK**	**ECC the Scalar SCALARK Value of Point Multiplication(SCALARK)** Because the SCALARK usually stores the private key, ECC accelerator do not allow to read the register SCALARK. For B-163 or K-163, SCALARK is stored in CRPT_ECC_K_00~CRPT_ECC_K_05 For B-233 or K-233, SCALARK is stored in CRPT_ECC_K_00~CRPT_ECC_K_07 For B-283 or K-283, SCALARK is stored in CRPT_ECC_K_00~CRPT_ECC_K_08 For B-409 or K-409, SCALARK is stored in CRPT_ECC_K_00~CRPT_ECC_K_12 For B-571 or K-571, SCALARK is stored in CRPT_ECC_K_00~CRPT_ECC_K_17 For P-192, SCALARK is stored in CRPT_ECC_K_00~CRPT_ECC_K_05 For P-224, SCALARK is stored in CRPT_ECC_K_00~CRPT_ECC_K_06 For P-256, SCALARK is stored in CRPT_ECC_K_00~CRPT_ECC_K_07 For P-384, SCALARK is stored in CRPT_ECC_K_00~CRPT_ECC_K_11 For P-521, SCALARK is stored in CRPT_ECC_K_00~CRPT_ECC_K_16

ECC DMA Source Address Register (CRPT_ECC_SADDR)

Register	Offset	R/W	Description	Reset Value
CRPT_ECC_SADDR	CRYP_BA+0xA48	R/W	ECC DMA Source Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**Reserved**	**ECC DMA Source Address** The ECC accelerator supports DMA function to transfer the DATA and PARAMETER between SRAM memory space and ECC accelarator. The SADDR keeps the source address of the data buffer where the source text is stored. Based on the source address, the ECC accelerator can read the DATA and PARAMETER from SRAM memory space and do ECC operation. The start of source address should be located at word boundary. That is, bit 1 and 0 of SADDR are ignored. SADDR can be read and written. In DMA mode, software must update the CRPT_ECC_SADDR before triggering START.

ECC DMA Destination Address Register (CRPT_ECC_DADDR)

Register	Offset	R/W	Description	Reset Value
CRPT_ECC_DADDR	CRYP_BA+0xA4C	R/W	ECC DMA Destination Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**DADDR**	**ECC DMA Destination Address** The ECC accelerator supports DMA function to transfer the DATA and PARAMETER between SRAM memory and ECC accelerator. The DADDR keeps the destination address of the data buffer where output data of ECC engine will be stored. Based on the destination address, the ECC accelerator can write the result data back to SRAM memory space after the ECC operation is finished. The start of destination address should be located at word boundary. That is, bit 1 and 0 of DADDR are ignored. DADDR can be read and written. In DMA mode, software must update the CRPT_ECC_DADDR before triggering START.

ECC Starting Address of Updated Registers (CRPT_ECC_STARTREG)

Register	Offset	R/W	Description	Reset Value
CRPT_ECC_STARTREG	CRYP_BA+0xA50	R/W	ECC Starting Address of Updated Registers	0x0000_0000

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

Bits	Description	
[31:0]	**STARTREG**	**ECC Starting Address of Updated Registers** The address of the updated registers that DMA feeds the first data or parameter to ECC engine. When ECC engine is active, ECC accelerator does not allow users to modify STARTREG, for example, to update input data from register CRPT_ECC POINTX1. Thus, the value of STARTREG is 0x808.

ECC DMA Word Count r (CRPT_ECC_WORDCNT)

Register	Offset	R/W	Description	Reset Value
CRPT_ECC_WORDCNT	CRYP_BA+0xA54	R/W	ECC DMA Word Count	0x0000_0000

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

Bits	Description	
[31:0]	**WORDCNT**	**ECC DMA Word Count** The CRPT_ECC_WORDCNT keeps the word count of source data that is for the required input data of ECC accelerator with various operations in DMA mode. Although CRPT_ECC_WORDCNT is 32-bit, the maximum of word count in ECC accelerator is 144 words. CRPT_ECC_WORDCNT can be read and written.

6.26.7.7 RSA Register

RSA Control Register (CRPT_RSA_CTL)

Register	Offset	R/W	Description	Reset Value
CRPT_RSA_CTL	CRYP_BA+0x1000	R/W	RSA Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			KEYLENGTH				
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
KEYLENGTH							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				LDC	LDN	LDE	LDM
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DMAEN	Reserved					STOP	START

Bits	Description	
[31:29]	**Reserved**	Reserved.
[28:16]	**KEYLENGTH**	The key length of RSA operation.
[15:12]	**Reserved**	Reserved.
[11]	**LDC**	**The Control Signal of Register for Input Data C of RSA** 0 = The register for C is not modified by DMA. 1 = The register for C is modified by DMA.
[10]	**LDN**	**The Control Signal of Register for Input Data n of RSA** 0 = The register for N is not modified by DMA. 1 = The register for N is modified by DMA.
[9]	**LDE**	**The Control Signal of Register for Input Data E of RSA** 0 = The register for E is not modified by DMA. 1 = The register for E is modified by DMA.
[8]	**LDM**	**The Control Signal of Register for Input Data m of RSA** 0 = The register for M is not modified by DMA. 1 = The register for M is modified by DMA.
[7]	**DMAEN**	**RSA Accelerator DMA Enable Bit** 0 = RSA DMA engine Disabled. 1 = RSA DMA engine Enabled. Only when START and DMAEN are 1, RSA DMA engine will be active
[6:2]	**Reserved**	Reserved.

[1]	**STOP**	**RSA Accelerator Stop** 0 = No effect. 1 = Abort RSA accelerator and make it into idle state. This bit is always 0 when it’s read back. Remember to clear RSA interrupt flag after stopping RSA accelerator.
[0]	**START**	**RSA Accelerator Start** 0 = No effect. 1 = Start RSA accelerator. BUSY flag will be set. This bit is always 0 when it’s read back. RSA accelerator will ignore this START signal when BUSY flag is 1.

RSA Status Register (CRPT_RSA_STS)

Register	Offset	R/W	Description	Reset Value
CRPT_RSA_STS	CRYP_BA+0x1004	R	RSA Status Register	0x0000_0000

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

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**BUSERR**	**RSA DMA Access Bus Error Flag** 0 = No error. 1 = Bus error will stop DMA operation and RSA accelerator.
[15:2]	**Reserved**	Reserved.
[1]	**DMABUSY**	**RSA DMA Busy Flag** 0 = RSA DMA is idle or finished. 1 = RSA DMA is busy.
[0]	**BUSY**	**RSA Accelerator Busy Flag** 0 = The RSA accelerator is idle or finished. 1 = The RSA accelerator is under processing and protects all registers. Remember to clear RSA interrupt flag after RSA accelerator finished

NUC980 SERIES TECHNICAL REFERENCE MANUAL

RSA the Base of the Exponentiation Register (CRPT_RSA_M)

Register	Offset	R/W	Description	Reset Value
CRPT_RSA_M_i	CRYP_BA+0x1008+0x4*i	R/W	RSA the Base of Exponentiation Word i	0x0000_0000

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

Bits	Description	
[31:0]	**M**	**RSA the Base of the Exponentiation Register (M)** For 1024-bit RSA, M is stored in CRPT_RSA_M_0 ~ CRPT_RSA_M_31 For 2048-bit RSA, M is stored in CRPT_RSA_M_0 ~ CRPT_RSA_M_63

RSA the Exponent of the Exponentiation Register (CRPT_RSA_E)

Register	Offset	R/W	Description	Reset Value
CRPT_RSA_E_i	CRYP_BA+0x1208+0x4*i	W	RSA the Exponent of Exponentiation Word i	0x0000_0000

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

Bits	Description	
[31:0]	**E**	**RSA the Exponent of the Exponentiation Register Register (E)** For 1024-bit RSA, E is stored in CRPT_RSA_E_0 ~ CRPT_RSA_E_31 For 2048-bit RSA, E is stored in CRPT_RSA_E_0 ~ CRPT_RSA_E_63

RSA the Base of the Modulud Operation Register (CRPT_RSA_N)

Register	Offset	R/W	Description	Reset Value
CRPT_RSA_N_i	CRYP_BA+0x140 8+0x4*i	R/W	RSA the Base of Modulus Operation Word i	0x0000_0000

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

Bits	Description	
[31:0]	**N**	**RSA the Base of the Modulud Operatio Register (n)** For 1024-bit RSA, N is stored in CRPT_RSA_N_0 ~ CRPT_RSA_N_31 For 2048-bit RSA, N is stored in CRPT_RSA_N_0 ~ CRPT_RSA_N_63

RSA the Constant Value of the Montgomery Domain Register (CRPT_RSA_C)

Register	Offset	R/W	Description	Reset Value
CRPT_RSA_C_i	CRYP_BA+0x160 8+0x4*i	R/W	RSA the Constant Value of Montgomery Domain Word i	0x0000_0000

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

Bits	Description	
[31:0]	**C**	**RSA the Constant Value of the Montgomery Domain (C)** For 1024-bit RSA, C is stored in CRPT_RSA_C_0 ~ CRPT_RSA_C_31 For 2048-bit RSA, C is stored in CRPT_RSA_C_0 ~ CRPT_RSA_C_63

RSA DMA Source Address Register (CRPT_RSA_SADDR)

Register	Offset	R/W	Description	Reset Value
CRPT_RSA_SADDR	CRYP_BA+0x1808	R/W	RSA DMA Source Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**SADDR**	**RSA DMA Source Address** The RSA accelerator supports DMA function to transfer the DATA and PARAMETER between SRAM memory space and RSA accelarator. The SADDR keeps the source address of the data buffer where the source text is stored. Based on the source address, the RSA accelerator can read the DATA and PARAMETER from SRAM memory space and do RSA operation. The start of source address should be located at word boundary. That is, bit 1 and 0 of SADDR are ignored. SADDR can be read and written. In DMA mode, software must update the CRPT_RSA_SADDR before triggering START.

NUC980 SERIES TECHNICAL REFERENCE MANUAL

RSA DMA Destination Address Register (CRPT_RSA_DADDR)

Register	Offset	R/W	Description	Reset Value
CRPT_RSA_DADDR	CRYP_BA+0x180C	R/W	RSA DMA Destination Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**DADDR**	**RSA DMA Destination Address** The RSA accelerator supports DMA function to transfer the DATA and PARAMETER between SRAM memory and RSA accelerator. The DADDR keeps the destination address of the data buffer where output data of RSA engine will be stored. Based on the destination address, the RSA accelerator can write the result data back to SRAM memory space after the RSA operation is finished. The start of destination address should be located at word boundary. That is, bit 1 and 0 of DADDR are ignored. DADDR can be read and written. In DMA mode, software must update the CRPT_RSA_DADDR before triggering START.

RSA Starting Address of Updated Registers (CRPT_RSA_STARTREG)

Register	Offset	R/W	Description	Reset Value
CRPT_RSA_STARTREG	CRYP_BA+0x1810	R/W	RSA Starting Address of Updated Registers	0x0000_0000

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

Bits	Description	
[31:0]	**STARTREG**	**RSA Starting Address of Updated Registers** The address of the updated registers that DMA feeds the first data or parameter to RSA engine. When RSA engine is active, RSA accelerator does not allow users to modify STARTREG, for example, to update input data from register CRPT_RSA M. Thus, the value of STARTREG is 0x1008.

NUC980 SERIES TECHNICAL REFERENCE MANUAL

RSA DMA Word Count r (CRPT_RSA_WORDCNT)

Register	Offset	R/W	Description	Reset Value
CRPT_RSA_WORDCNT	CRYP_BA+0x1814	R/W	RSA DMA Word Count	0x0000_0000

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

Bits	Description	
[31:0]	**WORDCNT**	**RSA DMA Word Count** The CRPT_RSA_WORDCNT keeps the word count of source data that is for the required input data of RSA accelerator with various operations in DMA mode. Although CRPT_RSA_WORDCNT is 32-bit, the maximum of word count in RSA accelerator is 256 words. CRPT_RSA_WORDCNT can be read and written.

6.27 Capture Sensor Interface Controller (CAP)

6.27.1 Overview

The Image Capture Interface is designed to capture image data from a sensor. After capturing or fetching image data, it will process the image data, and then FIFO outputs them into frame buffer.

6.27.2 Features

- 8-bit RGB565 sensor
- 8-bit YUV422 sensor
- Supports CCIR601 YCbCr color range scale to full YUV color range
- Supports 4 packaging format for packet data output: YUYV, Y only, RGB565, RGB555
- Supports YUV422 planar data output
- Supports the CROP function to crop input image to the required size for digital application.
- Supports the down scaling function to scale input image to the required size for digital application.
- Supports frame rate control
- Supports field detection and even/odd field skip mechanism
- Supports packet output dual buffer control through hardware buffer controller
- Supports negative/sepia/posterization color effect

6.27.3 Block Diagram

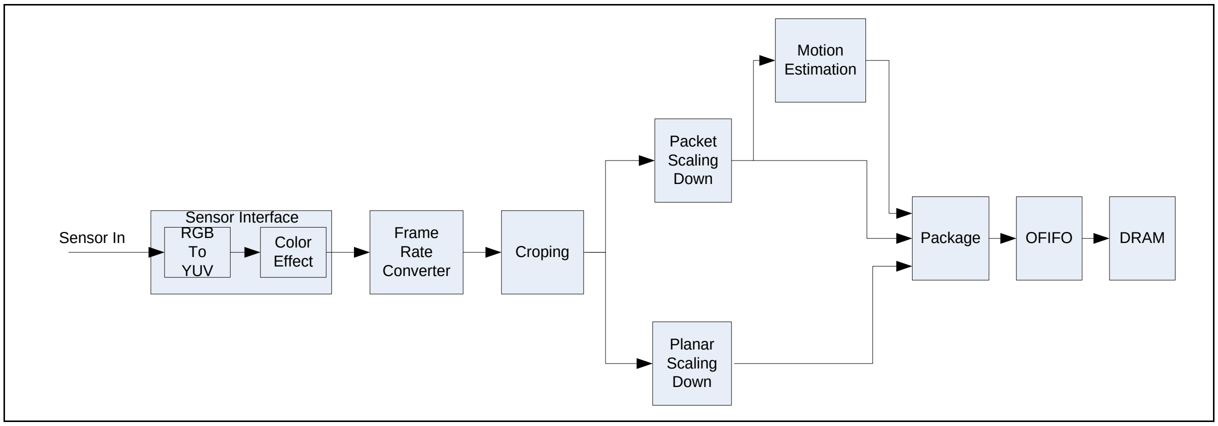


Figure 6.27-1 Image Capture Functional Block Diagram

6.27.4 Basic Configuration

- Clock Source Configuration
  - Enable the CMOS sensor reference clock on SENSOR (CLK_HCLKEN[27])
  - Enable the CAP0 peripheral clock on VCAP0 (CLK_HCLKEN[26])
  - Enable the CAP1 peripheral clock on VCAP1 (CLK_HCLKEN[31])
  - Select the CAP0 sensor clock source on SENSOR0_S (CLK_DIVCTL3[20:19])

- Select the CAP0 sensor clock source PLL divider on SENSOR0_SDIV (CLK_DIVCTL3[18:16])
- Select the CAP0 sensor clock divider on SENSOR0_N (CLK_DIVCTL3[27:24])
- Select the CAP1 sensor clock source on SENSOR1_S (CLK_DIVCTL2[20:19])
- Select the CAP1 sensor clock source PLL divider on SENSOR1_SDIV (CLK_DIVCTL2[18:16])
- Select the CAP1 sensor clock divider on SENSOR1_N (CLK_DIVCTL2[27:24])

● • Reset Configuration

- Reset CAP0 controller on VCAP0RST (SYS_AHBIPRST[10])
- Reset CAP1 controller on VCAP1RST (SYS_AHBIPRST[11])

Before using Capture Sensor interface, it’s necessary to configure related pins as the capture function and enable CAP’s clock. CMOS Sensor Reference Clock Enable (CLK_HCLKEN[27]) should be high.

For Capture Sensor interface related pin configuration, please refer to the register SYS_GPC _MFP and SYS_GPE _MFP to know how to configure related pins as the Capture Sensor interface function.

The CAP0 peripheral clock can be enabled by setting CAP0 HCLKEN (CLK_HCLKEN[26]) high. The CAP0 sensor clock source is selected by SENSOR0_S (CLK_DIVCTL3[20:19]), along with the extra PLL divider SENSOR0_SDIV (CLK_DIVCTL3[18:16]), and CAP0 sensor clock divider is determined by SENSOR0_N (CLK_DIVCTL3[27:24]).

The CAP1 peripheral clock can be enabled by setting CAP1 HCLKEN (CLK_HCLKEN[31]) high. The CAP1 sensor clock source is selected by SENSOR1_S (CLK_DIVCTL2[20:19]), along with the extra PLL divider SENSOR1_SDIV (CLK_DIVCTL2[18:16]), and CAP1 sensor clock divider is determined by SENSOR1_N (CLK_DIVCTL2[27:24]).

● • VCAP0 and VCAP1 Pin Configurations

Group	Pin Name	GPIO	MFP
VCAP0	VCAP0_CLKO	PC.3	MFP2
	VCAP0_DATA0	PC.8	MFP2
	VCAP0_DATA1	PC.9	MFP2
	VCAP0_DATA2	PC.10	MFP2
	VCAP0_DATA3	PC.11	MFP2
	VCAP0_DATA4	PC.12	MFP2
	VCAP0_DATA5	PC.13	MFP2
	VCAP0_DATA6	PC.14	MFP2
	VCAP0_DATA7	PC.15	MFP2
	VCAP0_FIELD	PC.7	MFP2
	VCAP0_HSYNC	PC.5	MFP2
	VCAP0_PCLK	PC.4	MFP2
	VCAP0_VSYNC	PC.6	MFP2

VCAP1	VCAP1_CLKO	PE.12	MFP7
	VCAP1_DATA0	PE.2	MFP7
	VCAP1_DATA1	PE.3	MFP7
	VCAP1_DATA2	PE.4	MFP7
	VCAP1_DATA3	PE.5	MFP7
	VCAP1_DATA4	PE.6	MFP7
	VCAP1_DATA5	PE.7	MFP7
	VCAP1_DATA6	PE.8	MFP7
	VCAP1_DATA7	PE.9	MFP7
	VCAP1_FIELD	PE.10	MFP7
	VCAP1_HSYNC	PE.0	MFP7
	VCAP1_PCLK	PF.10	MFP7
	VCAP1_VSYNC	PE.1	MFP7

6.27.5 Functional Description

6.27.5.1 *Image Capture Flow Chart*

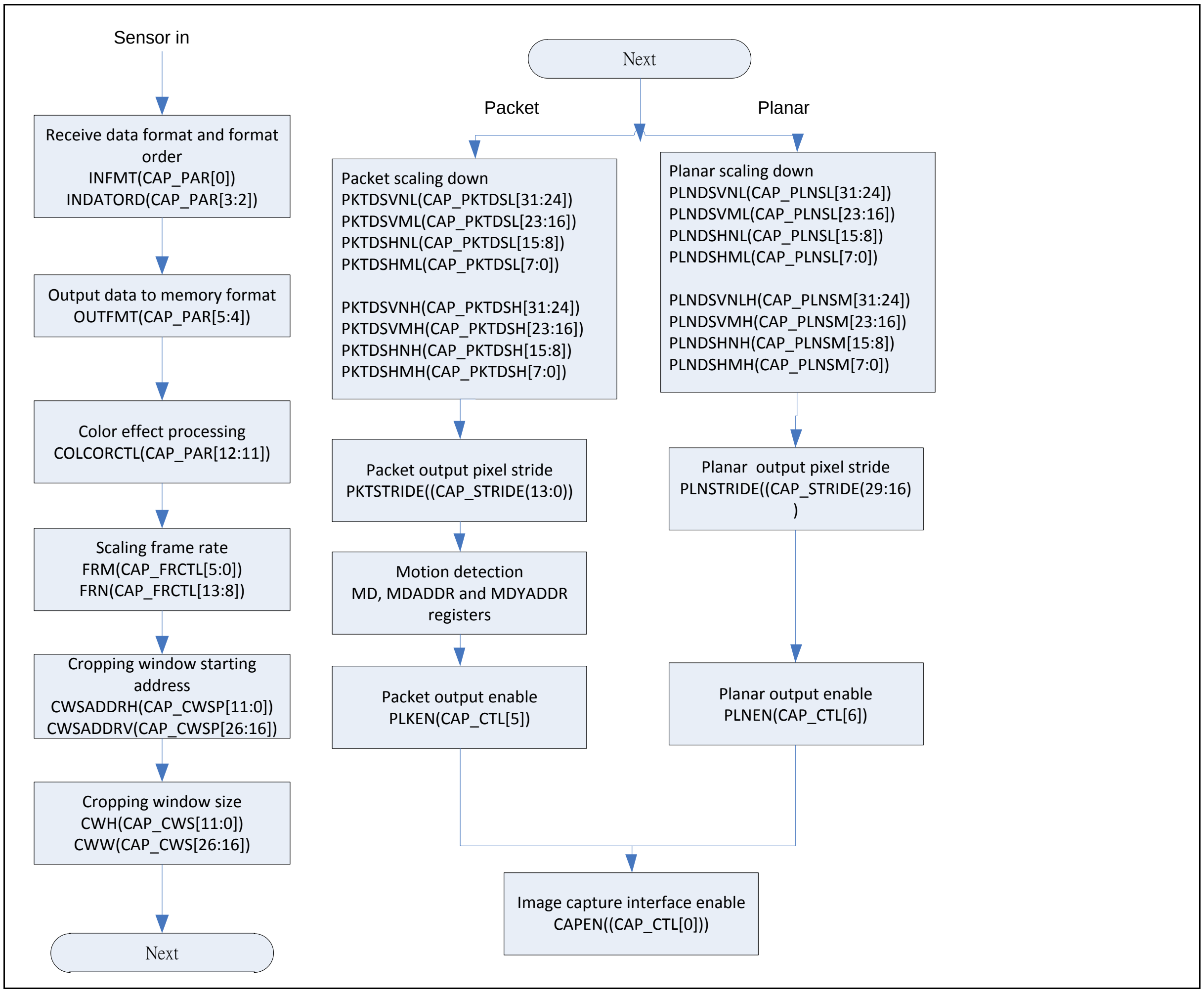

Sensor in
Receive data format and format order
INFMT(CAP_PAR[0])
INDATORD(CAP_PAR[3:2])
Output data to memory format
OUTFMT(CAP_PAR[5:4])
Color effect processing
COLCORCTL(CAP_PAR[12:11])
Scaling frame rate
FRM(CAP_FRCTL[5:0])
FRN(CAP_FRCTL[13:8])
Cropping window starting address
CWSADDRH(CAP_CWSP[11:0])
CWSADDRV(CAP_CWSP[26:16])
Cropping window size
CWH(CAP_CWS[11:0])
CWW(CAP_CWS[26:16])
Next
Next
Packet
Planar
Packet scaling down
PKTDSVNL(CAP_PKTDSL[31:24])
PKTDSVML(CAP_PKTDSL[23:16])
PKTDSHNL(CAP_PKTDSL[15:8])
PKTDSHML(CAP_PKTDSL[7:0])
PKTDSVNH(CAP_PKTDSH[31:24])
PKTDSVMH(CAP_PKTDSH[23:16])
PKTDSHNH(CAP_PKTDSH[15:8])
PKTDSHMH(CAP_PKTDSH[7:0])
Planar scaling down
PLNDSVNL(CAP_PLNSL[31:24])
PLNDSVML(CAP_PLNSL[23:16])
PLNDSHNL(CAP_PLNSL[15:8])
PLNDSHML(CAP_PLNSL[7:0])
PLNDSVNLH(CAP_PLNSM[31:24])
PLNDSVMH(CAP_PLNSM[23:16])
PLNDSHNH(CAP_PLNSM[15:8])
PLNDSHMH(CAP_PLNSM[7:0])
Packet output pixel stride
PKTSTRIDE((CAP_STRIDE(13:0))
Planar output pixel stride
PLNSTRIDE((CAP_STRIDE(29:16)
)
Motion detection
MD, MDADDR and MDYADDR registers
Packet output enable
PLKEN(CAP_CTL[5])
Planar output enable
PLNEN(CAP_CTL[6])
Image capture interface enable
CAPEN((CAP_CTL[0]))

Figure 6.27-2 Image Capture Flow Chart

6.27.5.2 *Cropping Feature*

The capture interface can select a window from the received image. The size of the window is specified by the number of pixel clocks (horizontal dimension) and the number of lines (vertical dimension). The start (left upper corner) coordinates can be specified by the CAP_CWSP register. The size (vertical dimension in number of lines and horizontal dimension in number of pixel clocks) can be specified by the CAP_CWS register.

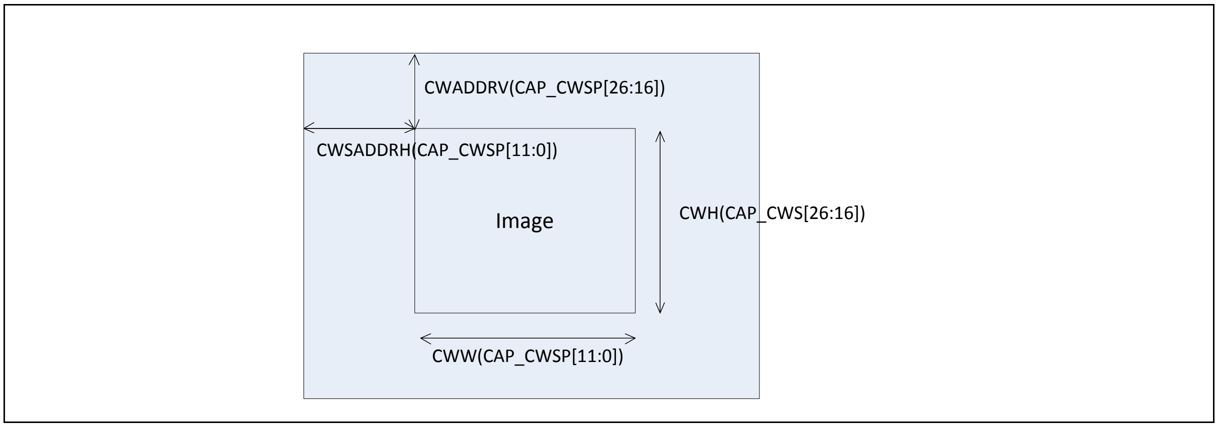


Figure 6.27-3 Image Start and Size of the Window after Cropping Block

6.27.5.3 One Shutter Mode (Single Frame)

In this mode, a single frame is captured. After the SHUTTER (CAP_CTL[16]) bit is set, the Image Capture interface automatically disables the capture interface after a frame is captured.

6.27.5.4 Motion Detection

The feature is used to detect object movement. MDSM (CAP_MD[9]) and MDBS (CAP_MD[8]) are determined using the method of storage. If the difference between the center of BASEADDR(CAP_PKTBA0[31:0]) block and the center of MDYADDR(CAP_MDYADDR[31:0]) block is greater than MDTHR (CAP_MD[20:16]) the first bit will change to 1, and other bits will be different, as shown in Figure 6.27-4.

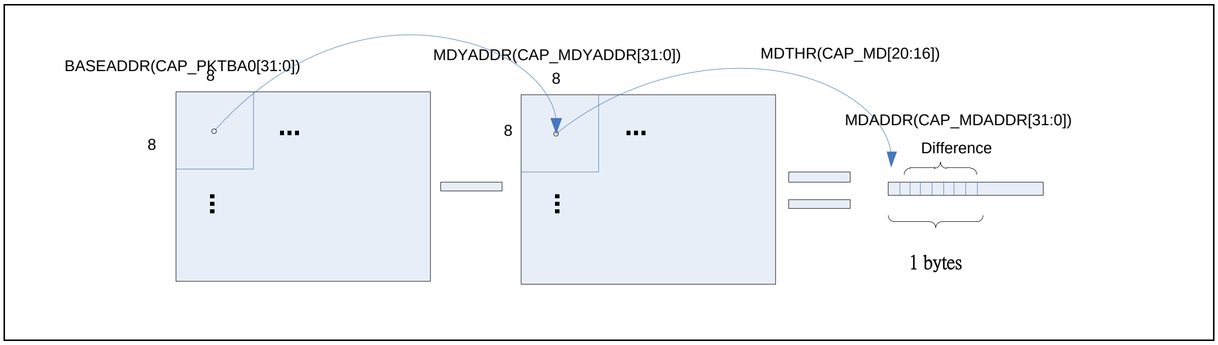


Figure 6.27-4 MDSM (MD[9]) is set to 0 and MDBS (MD[8]) is set to 1

If the difference between the center of BASEADDR(CAP_PKTBA0[31:0]) block and the center of CAP_MDYADDR block is greater than MDTHR (CAP_MD[20:16]) the first bit is set to 1, and if the next difference between the center of PACBA0 block and the center of MDYADDR(CAP_MDYADDR[31:0]) block is greater than MDTHR (CAP_MD[20:16]) the second bit will change to 1, as shown in Figure 6.27-5.

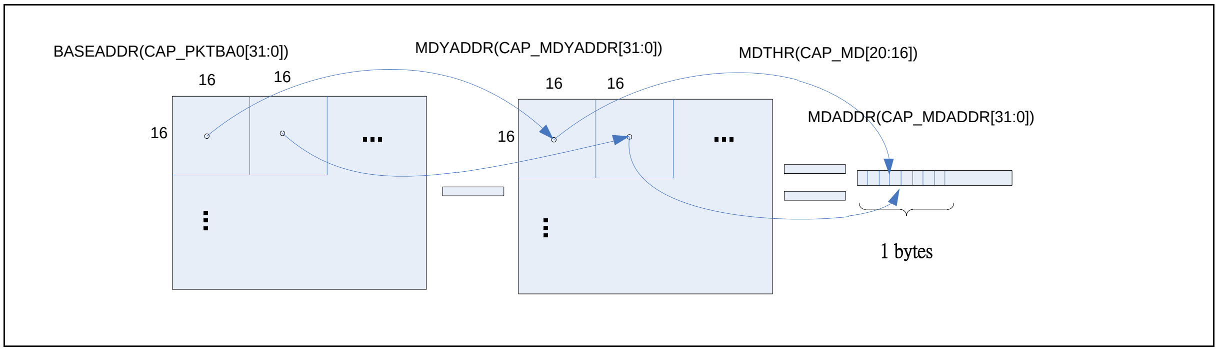


Figure 6.27-5 MDSM (MD[9]) is set to 1 and MDBS (MD[8]) is set to 0

6.27.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
CAP Base Address: **CAPx_BA = 0xB002_4000 - (0x1_0000 * x)** **x=0,1**				
CAP_CTL **x=0,1**	CAPx_BA+0x00	R/W	Image Capture Interface Control Register	0x0000_0040
CAP_PAR **x=0,1**	CAPx_BA+0x04	R/W	Image Capture Interface Parameter Register	0x0000_0000
CAP_INT **x=0,1**	CAPx_BA+0x08	R/W	Image Capture Interface Interrupt Register	0x0000_0000
CAP_POSTERIZE **x=0,1**	CAPx_BA+0x0C	R/W	YUV Component Posterizing Factor Register	0x0000_0000
CAP_MD **x=0,1**	CAPx_BA+0x10	R/W	Motion Detection Register	0x0000_0000
CAP_MDADDR **x=0,1**	CAPx_BA+0x14	R/W	Motion Detection Output Address Register	0x0000_0000
CAP_MDYADDR **x=0,1**	CAPx_BA+0x18	R/W	Motion Detection Temp Y Output Address Register	0x0000_0000
CAP_SEPIA **x=0,1**	CAPx_BA+0x1C	R/W	Sepia Effect Control Register	0x0000_0000
CAP_CWSP **x=0,1**	CAPx_BA+0x20	R/W	Cropping Window Starting Address Register	0x0000_0000
CAP_CWS **x=0,1**	CAPx_BA+0x24	R/W	Cropping Window Size Register	0x0000_0000
CAP_PKTSL **x=0,1**	CAPx_BA+0x28	R/W	Packet Scaling Vertical/Horizontal Factor Register (LSB)	0x0000_0000
CAP_PLNSL **x=0,1**	CAPx_BA+0x2C	R/W	Planar Scaling Vertical/Horizontal Factor Register (LSB)	0x0000_0000
CAP_FRCTL **x=0,1**	CAPx_BA+0x30	R/W	Scaling Frame Rate Factor Register	0x0000_0000
CAP_STRIDE **x=0,1**	CAPx_BA+0x34	R/W	Frame Output Pixel Stride Width Register	0x0000_0000
CAP_FIFOTH **x=0,1**	CAPx_BA+0x3C	R/W	FIFO Threshold Register	0x070D_0507
CAP_CMPADDR **x=0,1**	CAPx_BA+0x40	R/W	Compare Memory Base Address Register	0xFFFF_FFFC
CAP_PKTSM	CAPx_BA+0x48	R/W	Packet Scaling Vertical/Horizontal Factor Register (MSB)	0x0000_0000

x=0,1				
CAP_PLNSM **x=0,1**	CAPx_BA+0x4C	R/W	Planar Scaling Vertical/Horizontal Factor Register (MSB)	0x0000_0000
CAP_CURADDRP **x=0,1**	CAPx_BA+0x50	R	Current Packet System Memory Address Register	0x0000_0000
CAP_CURADDRY **x=0,1**	CAPx_BA+0x54	R	Current Planar Y System Memory Address Register	0x0000_0000
CAP_CURADDRU **x=0,1**	CAPx_BA+0x58	R	Current Planar U System Memory Address Register	0x0000_0000
CAP_CURADDRV **x=0,1**	CAPx_BA+0x5C	R	Current Planar V System Memory Address Register	0x0000_0000
CAP_PKTBA0 **x=0,1**	CAPx_BA+0x60	R/W	System Memory Packet Base Address 0 Register	0x0000_0000
CAP_YBA **x=0,1**	CAPx_BA+0x80	R/W	System Memory Planar Y Base Address Register	0x0000_0000
CAP_UBA **x=0,1**	CAPx_BA+0x84	R/W	System Memory Planar U Base Address Register	0x0000_0000
CAP_VBA **x=0,1**	CAPx_BA+0x88	R/W	System Memory Planar V Base Address Register	0x0000_0000

6.27.7 Register Description

Image Capture Interface Control Register (CAP_CTL)

Register	Offset	R/W	Description	Reset Value
CAP_CTL **x=0,1**	CAPx_BA+0x00	R/W	Image Capture Interface Control Register	0x0000_0040

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

Bits	Description	
[31:25]	**Reserved**	Reserved.
[24]	**VPRST**	**Capture Interface Reset** 0 = Capture interface reset Disabled. 1 = Capture interface reset Enabled.
[23:21]	**Reserved**	Reserved.
[20]	**UPDATE**	**Update Register at New Frame** 0 = Update register at new frame Disabled. 1 = Update register at new frame Enabled (Auto clear to 0 when register updated).
[19:17]	**Reserved**	Reserved.
[16]	**SHUTTER**	**Image Capture Interface Automatically Disable the Capture Inteface After a Frame Had Been Captured** 0 = Shutter Disabled. 1 = Shutter Enabled.
[15:7]	**Reserved**	Reserved.
[6]	**PKTEN**	**Packet Output Enable Bit** 0 = Packet output Disabled. 1 = Packet output Enabled.
[5]	**PLNEN**	**Planar Output Enable Bit** 0 = Planar output Disabled. 1 = Planar output Enabled.
[4]	**Reserved**	Reserved.

[3]	**Reserved**	Reserved.
[2:1]	**Reserved**	Reserved.
[0]	**CAPEN**	**Image Capture Interface Enable Bit** 0 = Image Capture Interface Disabled. 1 = Image Capture Interface Enabled.

Image Capture Interface Parameter Register (CAP_PAR)

Register	Offset	R/W	Description	Reset Value
CAP_PAR **x=0,1**	CAPx_BA+0x04	R/W	Image Capture Interface Parameter Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved					**FBB**	**Reserved**	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			**COLORCTL**		**VSP**	**HSP**	**PCLKP**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PLNFMT	**RANGE**	**OUTFMT**		**INDATORD**		**SENTYPE**	**INFMT**

Bits	Description	
[31:19]	**Reserved**	Reserved.
[18]	**FBB**	**Field by Blank** Hardware will tag field0 or field1 by vertical blanking instead of FIELD flag in ccir-656 mode. 0 = Field by blank Disabled. 1 = Field by blank Enabled.
[17:13]	**Reserved**	Reserved.
[12:11]	**COLORCTL**	**Special COLORCTL Processing** 00 = Normal Color. 01 = Sepia effect, corresponding U,V component value is set at register CAP_SEPIA. 10 = Negative picture. 11 = Posterize image, the Y, U, V components posterizing factor are set at register CAP_POSTERIZE.
[10]	**VSP**	**Sensor Vsync Polarity** 0 = Sync Low. 1 = Sync High.
[9]	**HSP**	**Sensor Hsync Polarity** 0 = Sync Low. 1 = Sync High.
[8]	**PCLKP**	**Sensor Pixel Clock Polarity** 0 = Input video data and signals are latched by falling edge of Pixel Clock. 1 = Input video data and signals are latched by rising edge of Pixel Clock.
[7]	**PLNFMT**	**Planar Output YUV Format** 0 = YUV422.

		1 = YUV420.
[6]	**RANGE**	**Scale Input YUV CCIR601 Color Range to Full Range** 0 = Default. 1 = Scale to full range.
[5:4]	**OUTFMT**	**Image Data Format Output to System Memory** 00 = YCbCr422. 01 = Only output Y. 10 = RGB555. 11 = RGB565.
[3:2]	**INDATORD**	**Sensor Input Data Order** If INFMT (CAP_PAR[0]) = 0 (YCbCr). 00 = Sensor input data (Byte 0 1 2 3) is Y0 U0 Y1 V0. 01 = Sensor input data (Byte 0 1 2 3) is Y0 V0 Y1 U0. 10 = Sensor input data (Byte 0 1 2 3) is U0 Y0 V0 Y1. 11 = Sensor input data (Byte 0 1 2 3) is V0 Y0 U0 Y1. If INFMT (CAP_PAR[0]) = 1 (RGB565). 00 = Sensor input data (Byte 0) is {R[4:0],G[5:3]},. Sensor input data (Byte 1) is {G[2:0],B[4:0]}. 01 = Sensor input data (Byte 0) is {B[4:0],G[5:3]},. Sensor input data (Byte 1) is {G[2:0], R[4:0]}. 10 = Sensor input data (Byte 0) is {G[2:0],B[4:0]},. Sensor input data (Byte 1) is {R[4:0], G[5:3]}. 11 = Sensor input data (Byte 0) is {G[2:0],R[4:0]},. Sensor input data (Byte 1) is {B[4:0], G[5:3]}.
[1]	**SENTYPE**	**Sensor Input Type** 0 = CCIR601. 1 = CCIR656, Vsync & Hsync embedded in the data signal.
[0]	**INFMT**	**Sensor Input Data Format** 0 = YCbCr422. 1 = RGB565.

Image Capture Interface Interrupt Register (CAP_INT)

Register	Offset	R/W	Description	Reset Value
CAP_INT x=0,1	CAPx_BA+0x08	R/W	Image Capture Interface Interrupt Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved			MDIEN	ADDRMIEN	Reserved	MEIEN	VIEN
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			MDINTF	ADDRMINTF	Reserved	MEINTF	VINTF

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20]	**MDIEN**	**Motion Detection Output Finish Interrupt Enable Bit** 0 = CAP_MD finish interrupt Disabled. 1 = CAP_MD finish interrupt Enabled.
[19]	**ADDRMIEN**	**Address Match Interrupt Enable Bit** 0 = Address match interrupt Disabled. 1 = Address match interrupt Enabled.
[18]	**Reserved**	Reserved.
[17]	**MEIEN**	**System Memory Error Interrupt Enable Bit** 0 = System memory error interrupt Disabled. 1 = System memory error interrupt Enabled.
[16]	**VIEN**	**Video Frame End Interrupt Enable Bit** 0 = Video frame end interrupt Disabled. 1 = Video frame end interrupt Enabled.
[15:5]	**Reserved**	Reserved.
[4]	**MDINTF**	**Motion Detection Output Finish Interrupt** 0 = Motion Detection Output Finish Interrupt did not occur. 1 = Motion Detection Output Finish Interrupt occurred. **Note:** This bit is cleared by writing 1 to it.
[3]	**ADDRMINTF**	**Memory Address Match Interrupt** 0 = Memory Address Match Interrupt did not occur. 1 = Memory Address Match Interrupt occurred. **Note:** This bit is cleared by writing 1 to it.

[2]	**Reserved**	Reserved.
[1]	**MEINTF**	**Bus Master Transfer Error Interrupt** 0 = Transfer Error did not occur. 1 = Transfer Error occurred. **Note:** This bit is cleared by writing 1 to it.
[0]	**VINTF**	**Video Frame End Interrupt** 0 = Did not receive a frame completely. 1 = Received a frame completely. **Note:** This bit is cleared by writing 1 to it.

YUV Component Posterizing Factor Register (CAP_POSTERIZE)

Register	Offset	R/W	Description	Reset Value
CAP_POSTERIZE **x=0,1**	CAPx_BA+0x0C	R/W	YUV Component Posterizing Factor Register	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**YCOMP**	**Y Component Posterizing Factor** Specify the 8-bit value of YCOMP for Y_Posterizing_Factor[7:0]. Final_Y_Out[7:0] = Original_Y[7:0] & Y_Posterizing_Factor[7:0].
[15:8]	**UCOMP**	**U Component Posterizing Factor** Specify the 8-bit value of UCOMP for U_Posterizing_Factor[7:0]. Final_U_Out[7:0] = Original_U[7:0] & U_Posterizing_Factor[7:0].
[7:0]	**VCOMP**	**V Component Posterizing Factor** Specify the 8-bit value of VCOMP for V_Posterizing_Factor[7:0]. Final_V_Out[7:0] = Original_V[7:0] & V_Posterizing_Factor[7:0].

Motion Detection Register (CAP_MD)

Register	Offset	R/W	Description	Reset Value
CAP_MD **x=0,1**	CAPx_BA+0x10	R/W	Motion Detection Register	0x0000_0000

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

Bits	Description	
[31:21]	**Reserved**	Reserved.
[20:16]	**MDTHR**	**Motion Detection Differential Threshold** Specify the 5-bit value of MDTHR for the motion detection differential threshold.
[15:12]	**Reserved**	Reserved.
[11:10]	**MDDF**	**Motion Detection Detect Frequency** 00 = Each frame. 01 = Every 2 frame. 10 = Every 3 frame. 11 = Every 4 frame.
[9]	**MDSM**	**Motion Detection Save Mode** 0 = 1 bit DIFF + 7 bit Y Differential. 1 = 1 bit DIFF only.
[8]	**MDBS**	**Motion Detection Block Size** 0 = 16x16. 1 = 8x8.
[7:1]	**Reserved**	Reserved.
[0]	**MDEN**	**Motion Detection Enable Bit** 0 = CAP_MD Disabled. 1 = CAP_MD Enabled.

Motion Detection Output Address Register (CAP_MDADDR)

Register	Offset	R/W	Description	Reset Value
CAP_MDADDR **x=0,1**	CAPx_BA+0x14	R/W	Motion Detection Output Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**MDADDR**	**Motion Detection Output Address Register (Word Alignment)** Specify the 32-bit value of MDADDR for the motion detection output address.

Motion Detection Temp Y Output Address Register (CAP_MDYADDR)

Register	Offset	R/W	Description	Reset Value
CAP_MDYADDR **x=0,1**	CAPx_BA+0x18	R/W	Motion Detection Temp Y Output Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**MDYADDR**	**Motion Detection Temp Y Output Address Register (Word Alignment)** Specify the 32-bit value of MDYADDR for the motion detection temporary Y output address.

Sepia Effect Control Register (CAP_SEPIA)

Register	Offset	R/W	Description	Reset Value
CAP_SEPIA **x=0,1**	CAPx_BA+0x1C	R/W	Sepia Effect 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**
UCOMP							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
VCOMP							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:8]	**UCOMP**	**U Component Value for Sepia Color Effect** Define the constant U component while the “Sepia” color effect is turned on.
[7:0]	**VCOMP**	**V Component Value for Sepia Color Effect** Define the constant V component while the “Sepia” color effect is turned on.

Cropping Window Starting Address Register (CAP_CWSP)

Register	Offset	R/W	Description	Reset Value
CAP_CWSP **x=0,1**	CAPx_BA+0x20	R/W	Cropping Window Starting Address Register	0x0000_0000

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

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:16]	**CWSADDRV**	**Cropping Window Vertical Starting Address** Specify the value of the cropping window vertical start address.
[15:12]	**Reserved**	Reserved.
[11:0]	**CWSADDRH**	**Cropping Window Horizontal Starting Address** Specify the value of the cropping window horizontal start address.

Cropping Window Size Register (CAP_CWS)

Register	Offset	R/W	Description	Reset Value
CAP_CWS **x=0,1**	CAPx_BA+0x24	R/W	Cropping Window Size Register	0x0000_0000

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

Bits	Description	
[31:27]	**Reserved**	Reserved.
[26:16]	**CWH**	**Cropping Window Height** Specify the szie of the cropping window height.
[15:12]	**Reserved**	Reserved.
[11:0]	**CWW**	**Cropping Window Width** Specify the szie of the cropping window width.

Packet Scaling Vertical/Horizontal Factor Register (LSB) (CAP_PKTSL)

Register	Offset	R/W	Description	Reset Value
CAP_PKTSL **x=0,1**	CAPx_BA+0x28	R/W	Packet Scaling Vertical/Horizontal Factor Register (LSB)	0x0000_0000

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

Bits	Description	
[31:24]	**PKTSVNL**	**Packet Scaling Vertical Factor n (Lower 8-bit)** Specify the lower 8-bit of numerator part (N) of the vertical scaling factor. The lower 8-bit will be cascaded with higher 8-bit (PKDSVNH) to form a 16-bit numerator of vertical factor.
[23:16]	**PKTSVML**	**Packet Scaling Vertical Factor m (Lower 8-bit)** Specify the lower 8-bit of denominator part (M) of the vertical scaling factor. The lower 8-bit will be cascaded with higher 8-bit (PKDSVMH) to form a 16-bit denominator (M) of vertical factor. The output image width will be equal to the image height * N/M. **Note:** The value of N must be equal to or less than M.
[15:8]	**PKTSHNL**	**Packet Scaling Horizontal Factor n (Lower 8-bit)** Specify the lower 8-bit of numerator part (N) of the horizontal scaling factor. The lower 8-bit will be cascaded with higher 8-bit (PKDSHNH) to form a 16-bit numerator of horizontal factor.
[7:0]	**PKTSHML**	**Packet Scaling Horizontal Factor m (Lower 8-bit)** Specifies the lower 8-bit of denominator part (M) of the horizontal scaling factor. The lower 8-bit will be cascaded with higher 8-bit (PKDSHMH) to form a 16-bit denominator (M) of vertical factor. The output image width will be equal to the image width * N/M. **Note:** The value of N must be equal to or less than M.

Planar Scaling Vertical/Horizontal Factor Register (LSB) (CAP_PLNSL)

Register	Offset	R/W	Description	Reset Value
CAP_PLNSL x=0,1	CAPx_BA+0x2C	R/W	Planar Scaling Vertical/Horizontal Factor Register (LSB)	0x0000_0000

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

Bits	Description	
[31:24]	**PLNSVNL**	**Planar Scaling Vertical Factor n (Lower 8-bit)** Specify the lower 8-bit of numerator part (N) of the vertical scaling factor. The lower 8-bit will be cascaded with higher 8-bit (PNDSVNH) to form a 16-bit numerator of vertical factor.
[23:16]	**PLNSVML**	**Planar Scaling Vertical Factor m (Lower 8-bit)** Specify the lower 8-bit of denominator part (M) of the vertical scaling factor. The lower 8-bit will be cascaded with higher 8-bit (PNDSVMH) to form a 16-bit denominator (M) of vertical factor. The output image width will be equal to the image height * N/M. **Note:** The value of N must be equal to or less than M.
[15:8]	**PLNSHNL**	**Planar Scaling Horizontal Factor n (Lower 8-bit)** Specify the lower 8-bit of numerator part (N) of the horizontal scaling factor. The lower 8-bit will be cascaded with higher 8-bit (PNDSHNH) to form a 16-bit numerator of horizontal factor.
[7:0]	**PLNSHML**	**Planar Scaling Horizontal Factor m (Lower 8-bit)** Specify the lower 8-bit of denominator part (M) of the horizontal scaling factor. The lower 8-bit will be cascaded with higher 8-bit (PNDSHMH) to form a 16-bit denominator (M) of vertical factor. The output image width will be equal to the image width * N/M. **Note:** The value of N must be equal to or less than M.

Packet Scaling Vertical/Horizontal Factor Register (MSB) (CAP_PKTSM)

Register	Offset	R/W	Description	Reset Value
CAP_PKTSM **x=0,1**	CAPx_BA+0x48	R/W	Packet Scaling Vertical/Horizontal Factor Register (MSB)	0x0000_0000

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

Bits	Description	
[31:24]	**PKTSVNH**	**Packet Scaling Vertical Factor n (Higher 8-bit)** Specify the higher 8-bit of numerator part (N) of the vertical scaling factor. Please refer to the register “CAP_PKTSL” to check the cooperation between these two registers.
[23:16]	**PKTSVMH**	**Packet Scaling Vertical Factor m (Higher 8-bit)** Specify the lower 8-bit of denominator part (M) of the vertical scaling factor. Please refer to the register “CAP_PKTSL” to check the cooperation between these two registers.
[15:8]	**PKTSHNH**	**Packet Scaling Horizontal Factor n (Higher 8-bit)** Specify the lower 8-bit of numerator part (N) of the horizontal scaling factor. Please refer to the register “CAP_PKTSL” for the detailed operation.
[7:0]	**PKTSHMH**	**Packet Scaling Horizontal Factor m (Higher 8-bit)** Specify the lower 8-bit of denominator part (M) of the horizontal scaling factor. Please refer to the register “CAP_PKTSL” for the detailed operation.

Planar Scaling Vertical/Horizontal Factor Register (MSB) (CAP_PLNSM)

Register	Offset	R/W	Description	Reset Value
CAP_PLNSM **x=0,1**	CAPx_BA+0x4C	R/W	Planar Scaling Vertical/Horizontal Factor Register (MSB)	0x0000_0000

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

Bits	Description	
[31:24]	**PLNSVNH**	**Planar Scaling Vertical Factor n (Higher 8-bit)** Specifies the higher 8-bit of numerator part (N) of the vertical scaling factor. For detailed programming, please refer to the register “CAP_PLNSL”.
[23:16]	**PLNSVMH**	**Planar Scaling Vertical Factor m (Higher 8-bit)** Specifies the lower 8-bit of denominator part (M) of the vertical scaling factor. For detailed programming, please refer to the register “CAP_PLNSL”.
[15:8]	**PLNSHNH**	**Planar Scaling Horizontal Factor n (Higher 8-bit)** Specifies the higher 8-bit of numerator part (N) of the horizontal scaling factor. For detailed programming, please refer to the register “CAP_PLNSL”.
[7:0]	**PLNSHMH**	**Planar Scaling Horizontal Factor m (Higher 8-bit)** Specifies the higher 8-bit of denominator part (M) of the horizontal scaling factor For detailed programming, please refer to the register “CAP_PLNSL”.

Scaling Frame Rate Factor Register (CAP_FRCTL)

Register	Offset	R/W	Description	Reset Value
CAP_FRCTL **x=0,1**	CAPx_BA+0x30	R/W	Scaling Frame Rate Factor Register	0x0000_0000

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

Bits	Description	
[31:14]	**Reserved**	Reserved.
[13:8]	**FRN**	**Scaling Frame Rate Factor n** Specify the denominator part (N) of the frame rate scaling factor.
[7:6]	**Reserved**	Reserved.
[5:0]	**FRM**	**Scaling Frame Rate Factor M** Specify the denominator part (M) of the frame rate scaling factor. The output image frame rate will be equal to input image frame rate * (N/M). **Note:** The value of N must be equal to or less than M.

Output Frame Pixel Stride Width (CAP_STRIDE)

Register	Offset	R/W	Description	Reset Value
CAP_STRIDE **x=0,1**	CAPx_BA+0x34	R/W	Frame Output Pixel Stride Width Register	0x0000_0000

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

Bits	Description	
[31:28]	**Reserved**	Reserved.
[29:16]	**PLNSTRIDE**	**Planar Frame Output Pixel Stride Width** The output pixel stride size of the planar pipe.
[15:12]	**Reserved**	Reserved.
[13:0]	**PKTSTRIDE**	**Packet Frame Output Pixel Stride Width** The output pixel stride size of the packet pipe.

FIFO Threshold Register (CAP_FIFOTH)

Register	Offset	R/W	Description	Reset Value
CAP_FIFOTH **x=0,1**	CAPx_BA+0x3C	R/W	FIFO Threshold Register	0x070D_0507

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

Bits	Description	
[31]	**OVF**	**FIFO Overflow Flag** Indicate the FIFO overflow flag.
[30:29]	**Reserved**	Reserved.
[28:24]	**PKTFTH**	**Packet FIFO Threshold** Specify the 5-bit value of the packet FIFO threshold.
[23:21]	**Reserved**	Reserved.
[20:16]	**PLNYFTH**	**Planar Y FIFO Threshold** Specify the 5-bit value of the planar Y FIFO threshold.
[15:12]	**Reserved**	Reserved.
[11:8]	**PLNUFTH**	**Planar U FIFO Threshold** Specify the 4-bit value of the planar U FIFO threshold.
[7:4]	**Reserved**	Reserved.
[3:0]	**PLNVFTH**	**Planar V FIFO Threshold** Specify the 4-bit value of the planar V FIFO threshold.

Compare Memory Address Register (CAP_CMPADDR)

Register	Offset	R/W	Description	Reset Value
CAP_CMPADDR **x=0,1**	CAPx_BA+0x40	R/W	Compare Memory Base Address Register	0xFFFF_FFFC

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

Bits	Description	
[31:0]	**CMPADDR**	**Compare Memory Base Address** It is a word alignment address, that is, the address is aligned by ignoring the 2 LSB bits [1:0].

Current Packet System Memory Address Register (CAP_CURADDRP)

Register	Offset	R/W	Description	Reset Value
CAP_CURADDRP x=0,1	CAPx_BA+0x50	R	Current Packet System Memory Address Register	0x0000_0000

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**	**Current Packet Output Memory Address** Speficy the 32-bit value of the current packet output memory address.

Current Planar Y System Memory Address Register (CAP_CURADDRY)

Register	Offset	R/W	Description	Reset Value
CAP_CURADDRY **x=0,1**	CAPx_BA+0x54	R	Current Planar Y System Memory Address Register	0x0000_0000

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**	**Current Planar Y Output Memory Address** Speficy the 32-bit value of the current planar Y output memory address.

Current Planar U System Memory Address Register (CAP_CURADDRU)

Register	Offset	R/W	Description	Reset Value
CAP_CURADDRU **x=0,1**	CAPx_BA+0x58	R	Current Planar U System Memory Address Register	0x0000_0000

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**	**Current Planar U Output Memory Address** Speficy the 32-bit value of the current planar U output memory address.

Current Planar V System Memory Address Register (CAP_CURADDRV)

Register	Offset	R/W	Description	Reset Value
CAP_CURADDRV **x=0,1**	CAPx_BA+0x5C	R	Current Planar V System Memory Address Register	0x0000_0000

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**	**Current Planar V Output Memory Address** Speficy the 32-bit value of the current planar V output memory address.

System Memory Packet Base Address 0 Register (CAP_PKTBA0)

Register	Offset	R/W	Description	Reset Value
CAP_PKTBA0 **x=0,1**	CAPx_BA+0x60	R/W	System Memory Packet Base Address 0 Register	0x0000_0000

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

Bits	Description	
[31:0]	**BASEADDR**	**System Memory Packet Base Address 0** It is a word alignment address, that is, the address is aligned by ignoring the 2 LSB bits [1:0].

System Memory Planar Y Base Address Register (CAP_YBA)

Register	Offset	R/W	Description	Reset Value
CAP_YBA **x=0,1**	CAPx_BA+0x80	R/W	System Memory Planar Y Base Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**BASEADDR**	**System Memory Planar Y Base Address** It is a word alignment address, that is, the address is aligned by ignoring the 2 LSB bits [1:0].

System Memory Planar U Base Address Register (CAP_UBA)

Register	Offset	R/W	Description	Reset Value
CAP_UBA **x=0,1**	CAPx_BA+0x84	R/W	System Memory Planar U Base Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**BASEADDR**	**System Memory Planar U Base Address** It is a word alignment address, that is, the address is aligned by ignoring the 2 LSB bits [1:0].

System Memory Planar V Base Address Register (CAP_VBA)

Register	Offset	R/W	Description	Reset Value
CAP_VBA **x=0,1**	CAPx_BA+0x88	R/W	System Memory Planar V Base Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**BASEADDR**	**System Memory Planar V Base Address** It is a word alignment address, that is, the address is aligned by ignoring the 2 LSB bits [1:0].

6.28 Analog to Digital Converter (ADC)

6.28.1 Overview

The NUC980 series contains one 12-bit Successive Approximation Register analog-to-digital converter (SAR A/D converter) with 9 input channels.

6.28.2 Features

- Resolution: 12-bit resolution
- DNL: +/-1.5 LSB, INL: +/-3 LSB
- Data Rate up to 200kSPS
- Analog Input Range: V_{REF} to AGND, can be rail-to-rail
- Analog Supply: 2.7-3.6V
- Digital Supply: 1.2V
- 9 Single-Ended analog inputs
- Auto Power Down
- Low Power Consumption: 2170uW (at 200k SPS), < 1uA

6.28.3 Block Diagram

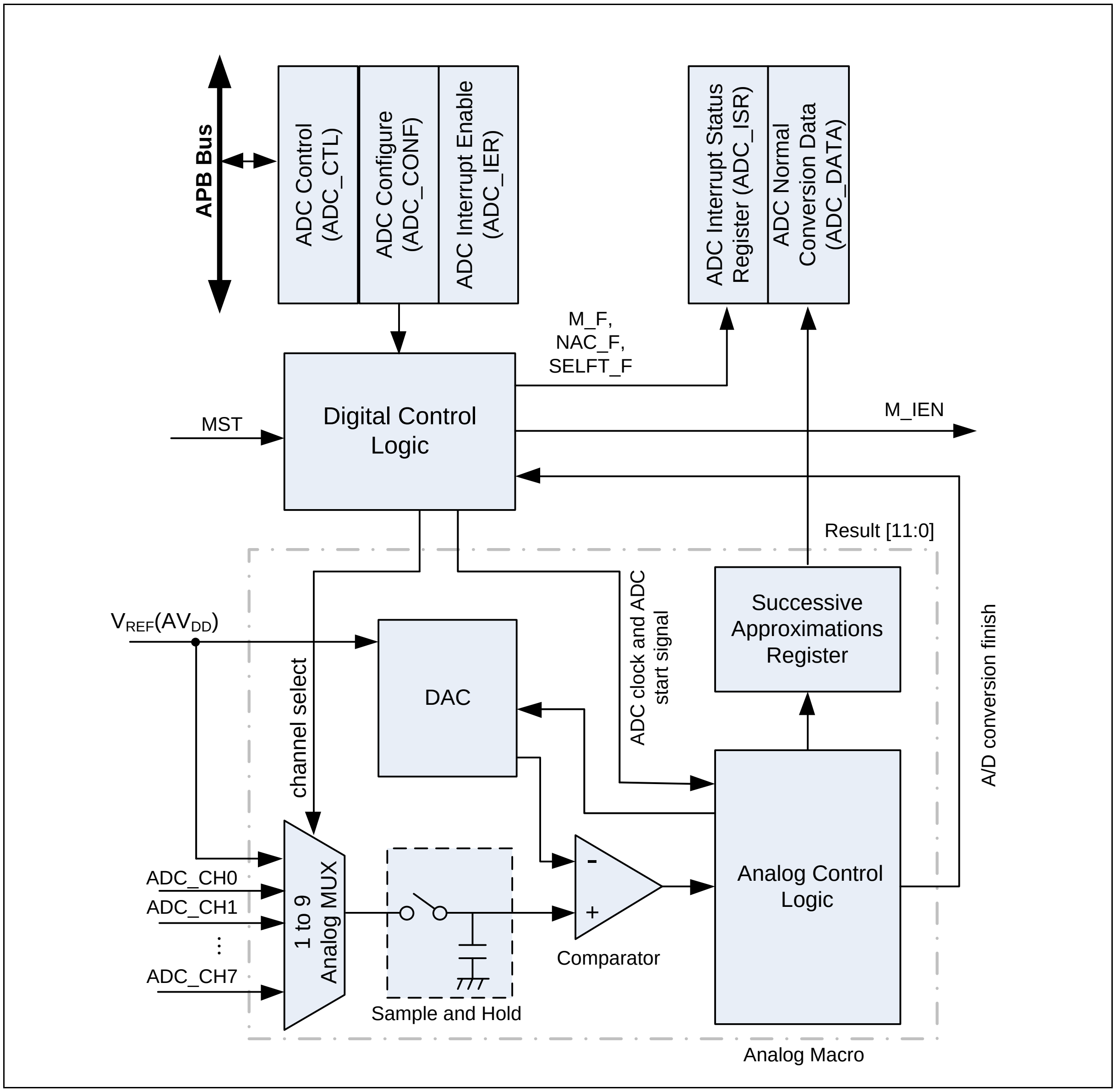


Figure 6.28-1 ADC Functional Block Diagram

6.28.4 Basic Configuration

- Clock source Configuration
  - Select the engine clock source on ADC_S (CLK_DIVCTL7[20:19])
  - Select the clock divider number on ADC_N (CLK_DIVCTL7[31:24])
  - Enable ADC peripheral clock on ADCCKEN (CLK_PCLKEN1[24]).
- Reset Configuration
  - Reset ADC controller on ADCRST(SYS_APBIPRST1[24])

- Pin Configuration

Group	Pin Name	GPIO	MFP
ADC	ADC_AIN0	PB.0	MFP8
	ADC_AIN1	PB.1	MFP8
	ADC_AIN2	PB.2	MFP8
	ADC_AIN3	PB.3	MFP8
	ADC_AIN4	PB.4	MFP8
	ADC_AIN5	PB.5	MFP8
	ADC_AIN6	PB.6	MFP8
	ADC_AIN7	PB.7	MFP8

6.28.5 Functional Description

6.28.5.1 ADC Transfer Function

The ADC output coding is offset in binary, 1LSB=VREF/4096, the transfer characteristic is shown in Figure 6.28-2:

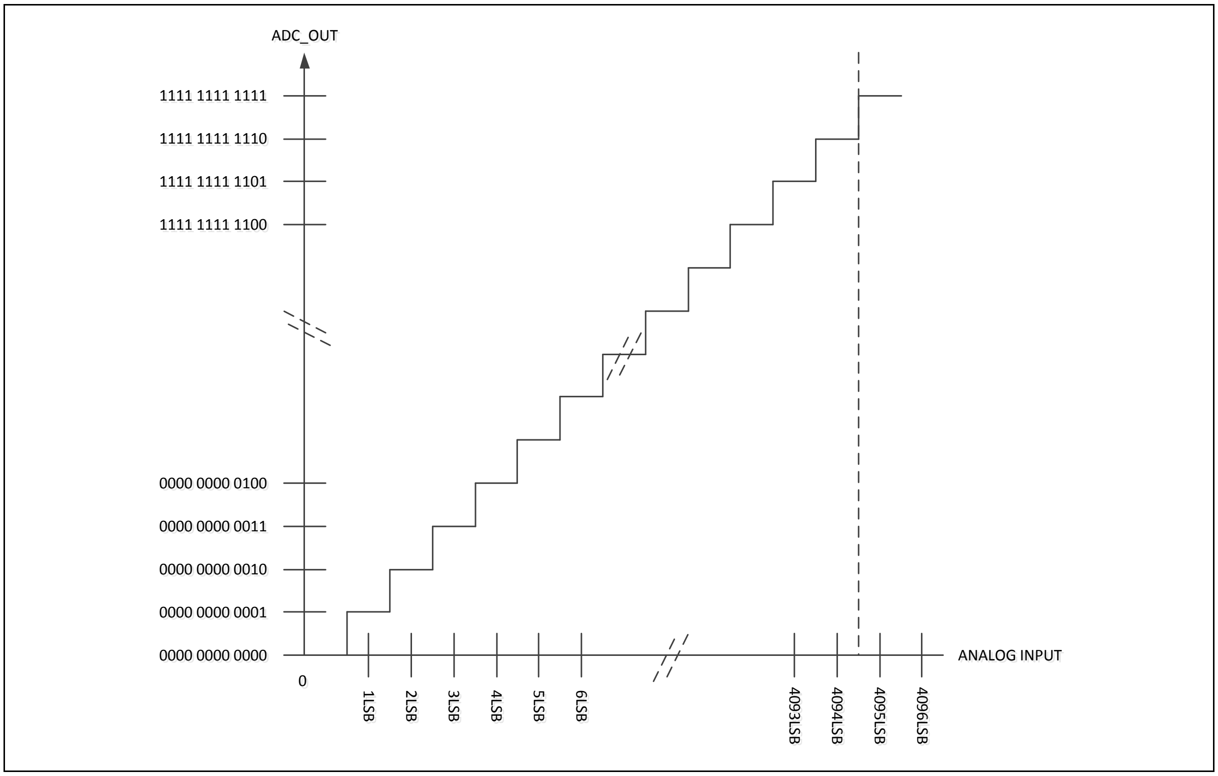


Figure 6.28-2 ADC Transfer Function

6.28.5.2 Selection of Input Signal

CHSEL[15:12]	Select ADC Analog Input Signal	Description
0000	ADC_CH0	ADC input, support 200KS/S

0001	ADC_CH1	ADC input, support 200KS/S
0010	ADC_CH2	ADC input, support 200KS/S
0011	ADC_CH3	ADC input, support 200KS/S
0100	ADC_CH4	ADC input, support 200KS/S
0101	ADC_CH5	ADC input, support 200KS/S
0110	ADC_CH6	ADC input, support 200KS/S
0111	ADC_CH7	ADC input, support 200KS/S
1000	V_{REF}	ADC input, support 200KS/S

The input current on analog input pins depends on analog input voltage and sampling rate. In the sampling mode, the input current charges the internal sampling capacitor. After the capacitor is fully charged, the current will be off. The internal sampling capacitor must be charged to 12-bit settling level at sampling time, under afore mentioned condition.

The signal source impedance connected to analog input needs to satisfy the following expression and Figure 6.28-3:

$$0.69 \times (12 + 1) \times C_{samp} \times 1.1 \times (R_{src} + R_{on} + R_{ref}) < \frac{1}{f_{clk}} \times 3$$

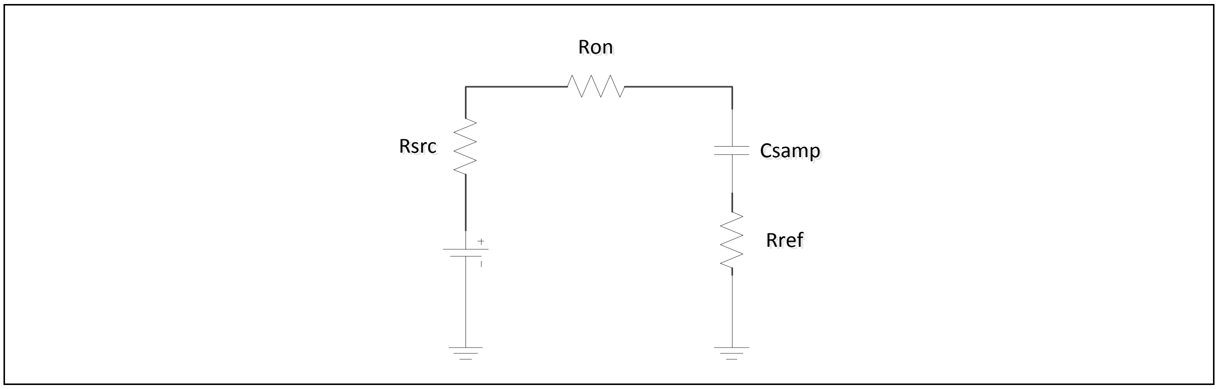


Figure 6.28-3 Simplified Sampling Diagram

6.28.5.3 Conversion rate at 200ks/s

$$C_{samp} = 25.6p$$

$$f_{clk} = 4M, \text{frequency of the clock}$$

$$R_{ref} = 1500$$

$$R_{on} = 100$$

6.28.5.4 Selection of Reference Voltage

REF_SEL[2:0]	**ADC Analog Reference Pair Selection Signals**
00	AGND33 vs. V_{REF} input.
11	AGND33 Vs $AV_{DD}33$

Table 6.28-1 Relation between Reference Voltage and Reference Selection

Reference voltage is selected very flexibly, and could be selected according to the application. When REF_SEL is set to 00, reference voltage of ADC is switched to V_{REF} pin. When REF_SEL is set to 11, reference voltage of ADC is switched to $AV_{DD}33$.

6.28.5.5 ADC Timing Diagram

When A/D conversion for each enabled function is completed, the result is transferred to the A/D data register corresponding to each functional. By the way, set ADC reference voltage stable counter REFCNT (ADC_CONF[19:16]) to add additional wait cycle (n), use sampling counter registers SAMPCNT (ADC_CONF[19:16]) to add additional sample and hold cycle (n).

Please note that these graphics refer to the additional sampling and hold clock cycles (i.e. in the situations where n=0, the sampling period is 1 × ADC_CLK).

The ADC conversion timing diagram is shown in Figure 6.28-4.

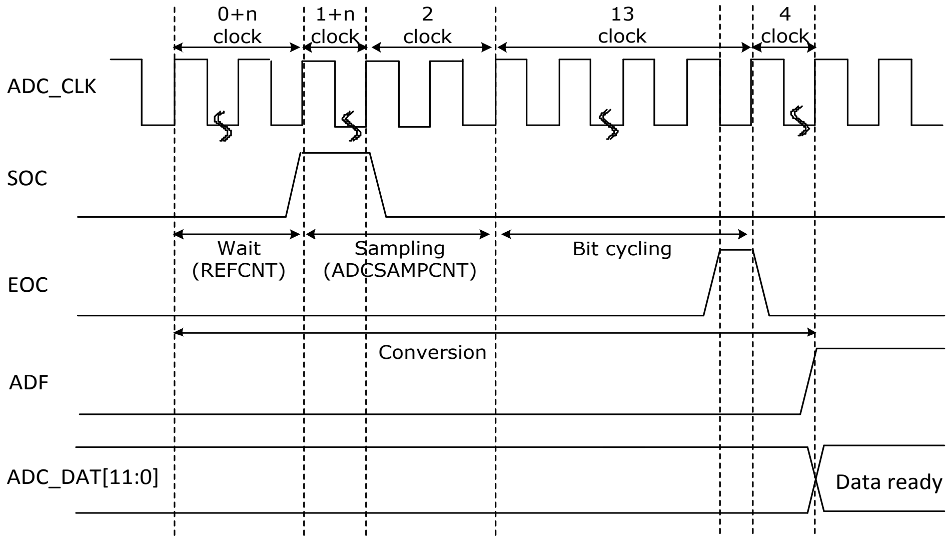


Figure 6.28-4 ADC Timing Diagram

6.28.6 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 = 0xB004_3000**				
ADC_CTL	ADC_BA+0x00	R/W	ADC Control	0x0000_0000
ADC_CONF	ADC_BA+0x04	R/W	ADC Configure	0x0000_0000
ADC_IER	ADC_BA+0x08	R/W	ADC Interrupt Enable Register	0x0000_0000
ADC_ISR	ADC_BA+0x0C	R/W	ADC Interrupt Status Register	0x0000_0000
ADC_DATA	ADC_BA+0x28	R	ADC Normal Conversion Data	0x0000_0000

6.28.7 Register Description

ADC Control (ADC_CTL)

Register	**Offset**	**R/W**	**Description**	**Reset Value**
ADC_CTL	ADC_BA+0x00	R/W	ADC Control	0x0000_0000

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

Bits	**Description**	
[31:9]	**Reserved**	Reserved.
[8]	**MST**	**Menu Start Conversion** 0 = Functional menu not start. 1 = Start functional menu conversion. **Note:** This bit is set by software and cleared by hardware when all the tasks listed in ADC_CONF are done.
[2]	**Reserved**	Reserved.
[1]	**VBG_EN**	**ADC Internal Bandgap Power Control** 0 = Power down internal bandgap. 1 = Power on internal bandgap.
[0]	**AD_EN**	**ADC Power Control** 0 = Power down ADC. 1 = Power on ADC.

ADC Configure (ADC_CONF)

Register	Offset	R/W	Description	Reset Value
ADC_CONF	ADC_BA+0x04	R/W	ADC Configure	0x0000_0000

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

Bits	Description	
[31:24]	**SAMPCNT**	**ADC Sample Counter** Set the counter value to extend the ADC start signal period to get more sampling time for precise conversion.
[23:20]	**Reserved**	Reserved.
[19:16]	**REFCNT**	**ADC Reference Counter** ADC reference voltage stable counter count value. 0 = 1 ADC clock. 1 = 2 ADC clock. 2 = 4 ADC clock. 3 = 8 ADC clock. 4 = 16 ADC clock. 5 = 32 ADC clock. 6 = 64 ADC clock. 7 = 128 ADC clock. 8 = 256 ADC clock. 9 = 512 ADC clock. 10 = 1024 ADC clock. 11 = 1024 ADC clock. 12 = 1024 ADC clock. 13 = 1024 ADC clock. 14 = 1024 ADC clock. 15 = 1024 ADC clock.

[15:12]	**CHSEL**	**Channel Selection** ADC input channel selection. 0000 = Channel 0. 0001 = Channel 1. 0010 = Channel 2. 0011 = Channel 3. 0100 = Channel 4. 0101 = Channel 5. 0110 = Channel 6. 0111 = Channel 7. 1000 = V_{REF}.
[11:8]	**Reserved**	Reserved.
[7:6]	**REFSEL**	**ADC Reference Select** ADC reference voltage select when ADC operate in normal conversion. 00 = AGND33 vs V_{REF} input. 11 = AGND33 vs $AV_{DD}33$.
[5:3]	**Reserved**	Reserved.
[2]	**NAC_EN**	**Normal A/D Conversion Enable Bit** ADC normal conversion function enable 0 = Normal A/D Conversion Disabled. 1 = Normal A/D Conversion Enable.
[1:0]	**Reserved**	Reserved.

ADC Interrupt Enable Register (ADC_IER)

Register	Offset	R/W	Description	Reset Value
ADC_IER	ADC_BA+0x08	R/W	ADC 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							**M_IEN**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**M_IEN**	**Menu Interrupt Enable Bit** 0 = Menu Interrupt Disabled. 1 = Menu Interrupt Enabled.

ADC Interrupt Status Register (ADC_ISR)

Register	Offset	R/W	Description	Reset Value
ADC_ISR	ADC_BA+0x0C	R/W	ADC 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					**NAC_F**	**Reserved**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**M_F**

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10]	**NAC_F**	**Normal AD Conversion Finish** Functional menu normal AD conversion finish. **Note:** set by hardware and write 1 to clear this bit.
[9:1]	**Reserved**	Reserved.
[0]	**M_F**	**Menu Complete Flag** Function menu complete status indicator. **Note:** set by hardware and write 1 to clear this bit.

ADC Normal Conversion Data (ADC_DATA)

Register	Offset	R/W	Description	Reset Value
ADC_DATA	ADC_BA+0x28	R	ADC Normal Conversion Data	0x0000_0000

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

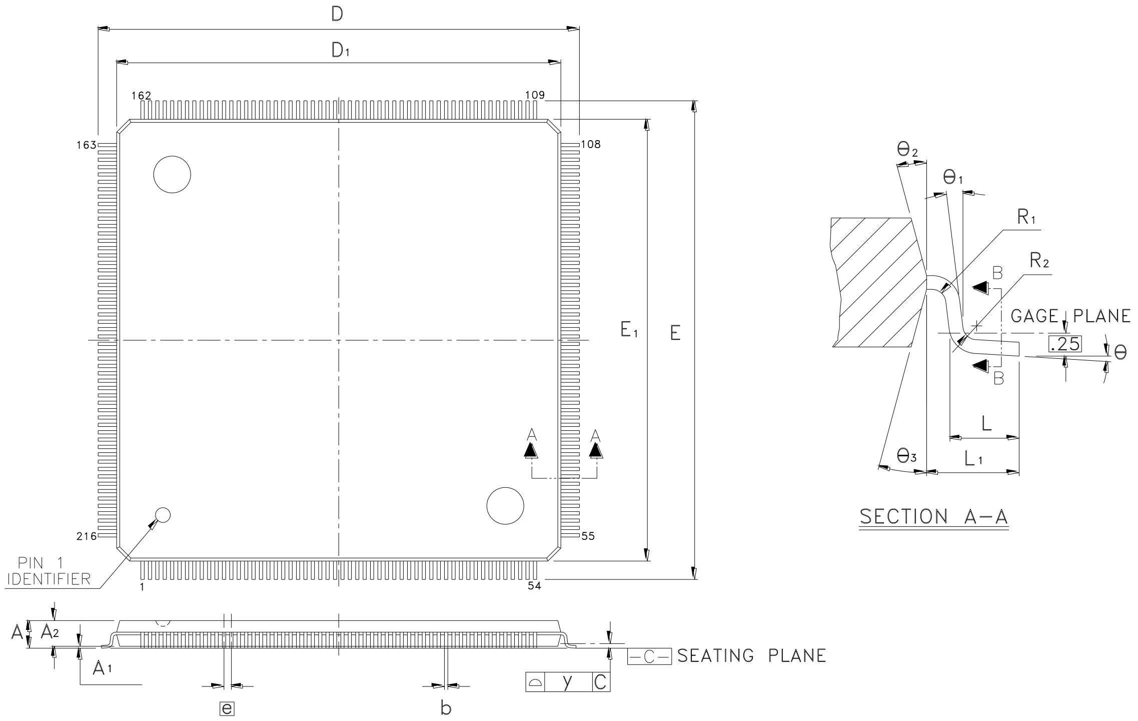
Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:0]	**ADC_DATA**	**ADC Data** When NAC_EN (ADC_CONF[2]) is enabled, the AD converting result with corresponding channel is stored in this register.

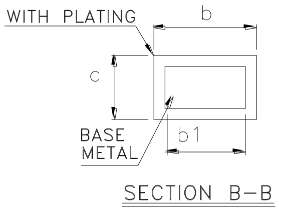
7 ELECTRICAL CHARACTERISTICS

For information on the NUC980 electrical characteristics, please refer to NUC980 Series Datasheet.

8 PACKAGE DIMENSIONS

8.1 LQFP 216L (24x24x1.4mm footprint 2.0mm)

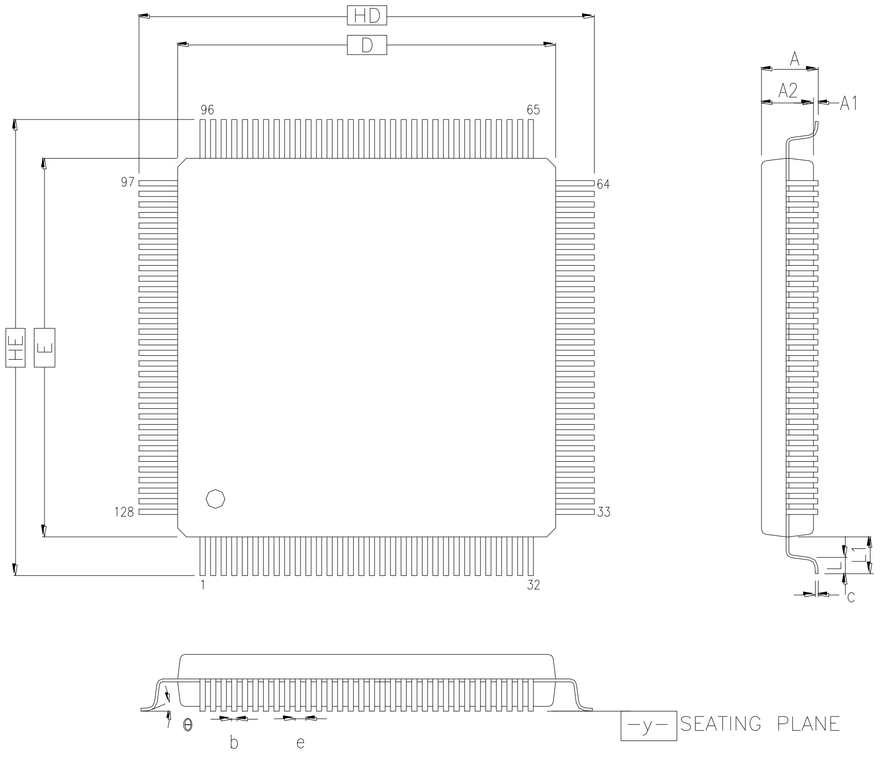




Controlling Dimension :Millimeters

Symbol	Dimension in mm			Dimension in inch		
	Min	Nom	Max	Min	Nom	Max
A	—	—	1.60	—	—	0.063
A_1	0.05	—	0.15	0.002	—	0.006
A_2	1.35	1.40	1.45	0.053	0.055	0.057
b	0.13	0.18	0.23	0.005	0.007	0.009
b_1	0.13	0.16	0.19	0.005	0.006	0.007
c	0.09	0.14	0.20	0.004	0.006	0.008
D	25.85	26.00	26.15	1.018	1.024	1.030
D_1	23.90	24.00	24.10	0.941	0.945	0.949
E	25.85	26.00	26.15	1.018	1.024	1.030
E_1	23.90	24.00	24.10	0.941	0.945	0.949
$\boxed{e}$	0.40 BSC			0.0157 BSC		
L	0.45	0.60	0.75	0.018	0.024	0.030
L_1	1.00 REF			0.039 REF		
R_1	0.08	—	—	0.003	—	—
R_2	0.08	—	—	0.003	—	—
θ	0°	3.5°	7°	0°	3.5°	7°
$θ_1$	0°	—	—	0°	—	—
$θ_2$	11°	12°	13°	11°	12°	13°
$θ_3$	11°	12°	13°	11°	12°	13°
y	0.08			0.00314		

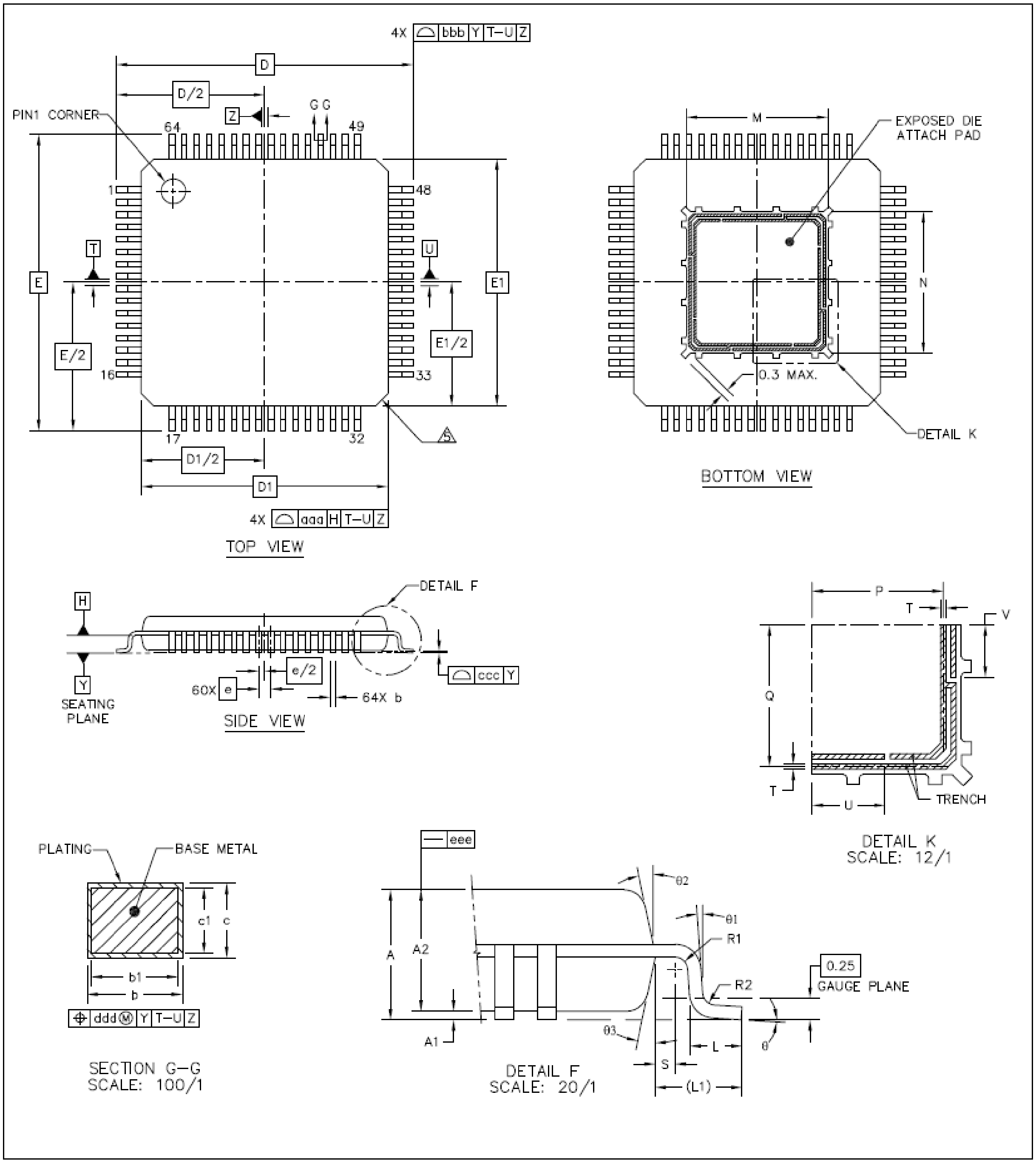
8.2 LQFP 128L (14x14x1.4mm footprint 2.0mm)

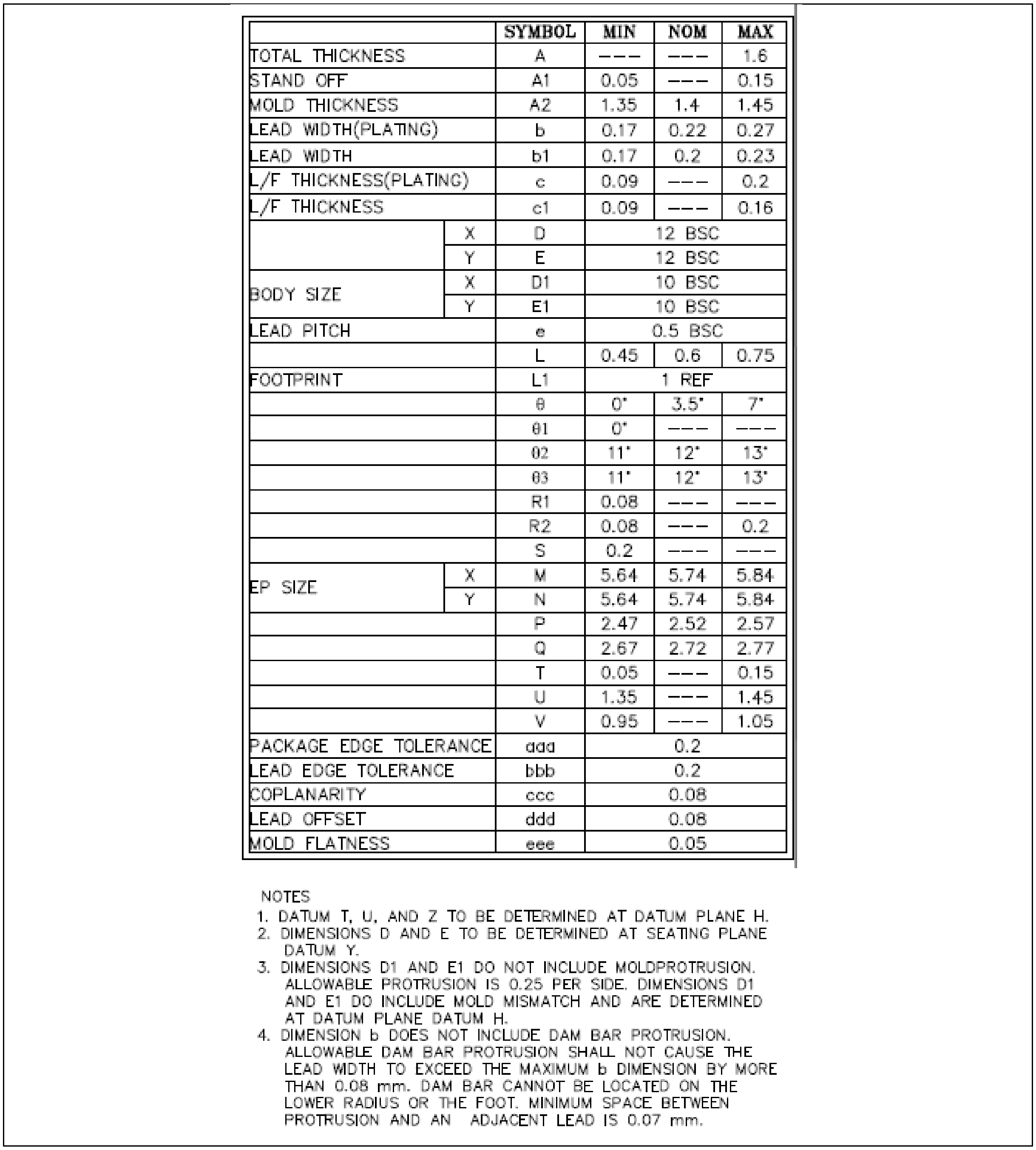


COTROL DIMENSIONS ARE IN MILLIMETERS.

SYMBOL	MILLIMETER			INCH		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
A	——	——	1.60	——	——	0.063
A1	0.05	——	0.15	0.002	——	0.006
A2	1.35	1.40	1.45	0.053	0.055	0.057
HD	16.00 BSC.			0.630 BSC.		
D	14.00 BSC.			0.551 BSC.		
HE	16.00 BSC.			0.630 BSC.		
E	14.00 BSC.			0.551 BSC.		
b	0.13	0.16	0.23	0.005	0.006	0.009
e	0.40 BSC.			0.016 BSC.		
θ	0°	3.5°	7°	0°	3.5°	7°
c	0.09	——	0.20	0.004	——	0.008
L	0.45	0.60	0.75	0.018	0.024	0.030
L 1	1.00 REF			0.039 REF		
y	–	–	0.1	–	–	0.004

8.3 LQFP 64L EX-PAD (10x10x1.4mm, footprint 2.0mm)





		SYMBOL	MIN	NOM	MAX
TOTAL THICKNESS		A	---	---	1.6
STAND OFF		A1	0.05	---	0.15
MOLD THICKNESS		A2	1.35	1.4	1.45
LEAD WIDTH(PLATING)		b	0.17	0.22	0.27
LEAD WIDTH		b1	0.17	0.2	0.23
L/F THICKNESS(PLATING)		c	0.09	---	0.2
L/F THICKNESS		c1	0.09	---	0.16
BODY SIZE	X	D		12 BSC	
	Y	E		12 BSC	
	X	D1		10 BSC	
	Y	E1		10 BSC	
LEAD PITCH		e		0.5 BSC	
FOOTPRINT		L	0.45	0.6	0.75
		L1		1 REF	
		θ	0°	3.5°	7°
		θ1	0°	---	---
		θ2	11°	12°	13°
		θ3	11°	12°	13°
		R1	0.08	---	---
		R2	0.08	---	0.2
		S	0.2	---	---
EP SIZE	X	M	5.64	5.74	5.84
	Y	N	5.64	5.74	5.84
		P	2.47	2.52	2.57
		Q	2.67	2.72	2.77
		T	0.05	---	0.15
		U	1.35	---	1.45
		V	0.95	---	1.05
PACKAGE EDGE TOLERANCE		aaa		0.2	
LEAD EDGE TOLERANCE		bbb		0.2	
COPLANARITY		ccc		0.08	
LEAD OFFSET		ddd		0.08	
MOLD FLATNESS		eee		0.05	

NOTES

1. DATUM T, U, AND Z TO BE DETERMINED AT DATUM PLANE H.
2. DIMENSIONS D AND E TO BE DETERMINED AT SEATING PLANE DATUM Y.
3. DIMENSIONS D1 AND E1 DO NOT INCLUDE MOLDPROTRUSION. ALLOWABLE PROTRUSION IS 0.25 PER SIDE. DIMENSIONS D1 AND E1 DO INCLUDE MOLD MISMATCH AND ARE DETERMINED AT DATUM PLANE DATUM H.
4. DIMENSION b DOES NOT INCLUDE DAM BAR PROTRUSION. ALLOWABLE DAM BAR PROTRUSION SHALL NOT CAUSE THE LEAD WIDTH TO EXCEED THE MAXIMUM b DIMENSION BY MORE THAN 0.08 mm. DAM BAR CANNOT BE LOCATED ON THE LOWER RADIUS OR THE FOOT. MINIMUM SPACE BETWEEN PROTRUSION AND AN ADJACENT LEAD IS 0.07 mm.

9 ABBREVIATIONS

9.1 Abbreviations

Acronym	Description
ADC	Analog-to-Digital Converter
AES	Advanced Encryption Standard
AIC	Advanced Interrupt Controller
APB	Advanced Peripheral Bus
AHB	Advanced High-Performance Bus
AMBA	Advanced Microcontroller Bus Architecture
BCH	Bose–Chaudhuri–Hocquenghem
BPS	Bit Per Second
CAN	Controller Area Network
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
DDR	Double Data Rate
DDR2	Double Data Rate 2
DMA	Direct Memory Access
EBI	External Bus Interface
ECC	Elliptic Curve Cryptography
ECC	Error Correcting code
EHCI	Enhance Host Controller Interface
EINT	External Interrupt pin
EMAC	Ethernet MAC Controller
eMMC	Embedded Multimedia Card
ETU	Elementary time unit
FIFO	First In, First Out
FIQ	Fast Interrupt
FMI	Flash Memory Interface
GPIO	General-Purpose Input/Output
HCLK	The Clock of Advanced High-Performance Bus
HMAC	keyed-Hash Message Authentication Code
HSUSBD	High Speed USB 2.0 Device Controller
HXT	12 MHz External High Speed Crystal Oscillator
I^{2}C	Inter-Integrated Circuit
I^{2}S	Inter-IC Sound

LIN	Local Interconnect Network
LPDDR	Low Power DDR
LSB	Least Significant Bit
LVD	Low Voltage Detect
LVR	Low Voltage Reset
LXT	32.748 kHz External Low Speed Crystal Oscillator
MLC	Multi-Level Cell NAND Flash
MMU	Memory Management Unit
MSB	Most Significant Bit
OHCI	Open Host Controller Interface
PCLK	The Clock of Advanced Peripheral Bus
PCM	Pulse Code Modulation
PDMA	Peripheral Direct Memory Access
PLL	Phase-Locked Loop
PMBus	Power Management Bus
PRNG	Pseudo Random Number Generator
PWM	Pulse Width Modulation
RMII	Reduced Media Independent Interface
RSA	Rivest、Shamir and Adleman Cryptography
RTC	Real Time Clock
SC	Smart Card
SD	Secure Digital
SDHC	Secure Digital High Capacity
SDIC	SDRAM Interface Controller
SDIO	Secure Digital Input Output
SDR	Single Data Rate
SHA	Secure Hash Algorithm
SLC	Single Level Cell NAND Flash
SMBus	System Management Bus
SPI	Serial Peripheral Interface
SPS	Samples per Second
TMR	Timer Controller
UART	Universal Asynchronous Receiver/Transmitter
USB	Universal Serial Bus

WDT	Watchdog Timer
WWDT	Window Watchdog Timer

10 REVISION HISTORY

Date	Revision	Description
2018.10.9	1.00	Initial version.
2020.01.7	1.01	1. Updated CAP, Crypto, I^{2}C, RTC, PDMA, PWM, and WWDT register description. 2. Updated product selection table in section 3.2.

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.